Construction of an Owl Ontology with the Protégé Development Environment

Vanesa Tenopala Zavala[1], José Crispín Hernández Hernández[1], Perfecto Malaquias Quintero Flores[1], Oscar Atriano Ponce[2]

[1]Tecnológico Nacional de México. Instituto Tecnológico de Apizaco
División de Estudios de Posgrado e Investigación, Avenida Instituto Tecnológico s/n C.P. 90300, Apizaco, Tlaxcala, México.

[2]Smartsoft America BA, Chiautempan 90802, Mexico; oatriano[at]smartsoftamerica.com.mx

Abstract: ***Today the internet has changed the way we communicate at all levels, including the professional who at the same time goes hand in hand with the staff. Every day it is more difficult to carry out some activity without the help of Internet search engines, so organizations must be at the forefront of technology and be able to meet the needs of users. The current web has some disadvantages in terms of the information that is published day by day, so the Internet intends to evolve giving way to the semantic web or web 3.0. The semantic web has as its main elements the "Ontologies". This article refers to a description for the development of OWL ontologies using the Protégé development environment as a tool.***

Keywords: Internet, Semantic web, OWL Ontologies, Protégé

1. Introduction

The web content is designed only for human understanding and not for machines. However, as a result of the growth of web content, the efficiency of current search engines is greatly reduced and becomes a major problem on the web. The idea of the Semantic Web was presented by the inventor of the World Wide Web, Tim Berners-Lee. The Semantic Web is an extension of the current web where information receives well-defined meanings that improved interoperability between machines and humans [1].

The semantic web is a way in which the user could find answers to their questions more quickly and easily. The objective of W3C (World Wide Web Consortium) is to make semantics the main protagonist in the linguistic part for the study of lexical structures and mental processes since by means of these human beings we are able to make sense of linguistic expressions, in addition to being the main way of Human-Computer communication.

The ontologies are the main elements of the Semantic Web that act as semantic repositories between machines.

2. Ontologies

Ontology is a term taken from philosophy, but in computer science an ontology is the representation of a domain of knowledge, where a set of objects and their relationships are described by a vocabulary [2].

An ontology is a hierarchy of concepts with attributes and relationships that have an agreed terminology to define semantic networks of interrelated units of information. An ontology provides a vocabulary of classes and relationships to describe a domain. Ontologies are used to capture knowledge about some domain of interest. An ontology describes the concepts of the domain, as well as the relationships that exist between them. There are different languages for ontologies that provide various facilities; The most recent standard given by the W3C is the OWL (Web Ontology Language) [3].

An ontology must be able to provide a good understanding of the domain it represents. This includes relationships of terms and concepts, their definitions or meanings, their relationships with each of them and the characteristics of the domain. This helps in solving the differences between the models and the domain used by the available sources, which is done by mapping the available data of all the resources for all the domains from the local model used by the source for the model specified by of the ontology [4].

2.1 OWL ontology language

The OWL is designed to be used when the information contained in the documents needs to be processed by programs or applications, as opposed to situations where the content only needs to be presented to human beings. OWL can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. This representation of the terms and their relationships is called ontology. The web ontology language (OWL) is an extension of the Resource Description Framework (RDF) language. OWL has functions that can express the meaning and semantics that XML, RDF, and RDFS cannot, goes beyond what these languages offer, since OWL has the possibility to represent the content of the web so that it can be understood and interpreted by machine.

If the machines are capable of performing reasoning tasks on the documents in which a semantics are used that go further than the basic semantics of RDF Schema, the semantic web will be on track.

OWL has the following levels:

- OWL Lite: It has simple restrictions (cardinality only 0 or 1), it facilitates compatibility with other models/paradigms in addition to facilitating the development of authoring tools. It has an efficient reasoning, facilitates the development of tools, compatibility with other models. It is the least expressive and is used in situations where there is only a simple hierarchy of classes and where few restrictions are needed.

- OWL DL (Description Logics): It is medium term in terms of expressiveness; it is based on descriptive logic, which is a logic based on first order; This is favorable for automatic reasoning, for ranking hierarchies and for detecting inconsistencies in ontologies. Use restrictions to ensure the existence of a reasonable reasoning procedure. There can be no cardinality restrictions (local or global) in transitive properties, neither its inverse nor its super properties. There can be missing definitions. The axioms of inequality can only be applied to named instances.
- OWL Full: It is the most expressive; it is used in situations where extreme expressiveness is more important than the ability to reason. Performs OWL and RDF syntax union (no restrictions).

OWL levels based on RDF explain that OWL Lite is contained in OWL DL and this is contained in OWL Full. OWL is expressed with RDF primitives, so that any valid OWL document is a valid RDF document. RDF is contained in OWL Full (the latter is an extension of RDF) so that a valid RDF document is a valid document in OWL Full.

3. Protégé

Protégé is an open source tool developed at Stanford Medical Informatics. Like many other modeling tools, Protégé is separated between a model part and a view part. The Protégé model is the internal representation mechanism of ontologies and knowledge bases. The components of the Protégé view provide the user with an interface to visualize and manipulate the model [5].

The Protégé model is based on a metamodeling comparable to object-oriented systems and frame-based systems. Protégé can represent ontologies that consist of classes, properties, characteristics of properties and instances, its advantage is that its metamodeling is a Protégé ontology. The structure of the Protégé metamodeling easily enables extension and adaptation to other representations.

Protégé is capable of generating user interfaces that support the creation of instances. In each class of the ontology the system creates a form with editing components for each property of the class [6].

Protégé can currently be used to load, edit and save ontologies in various formats such as CLIPS, RDF, XML, UML, OWL. Ontologies play an important role in semantic web applications. Protégé extensions can benefit from the general services provided by the core platform, to an event mechanism, undo capabilities and plugin mechanisms.

With the Protégé OWL plugin, you can allow many people to edit the same ontology at the same time. Protégé also provides a highly scalable database, which allows users to create ontologies with many classes [7].

The OWL plugin is a complex extension that can be used to edit OWL files and databases. The plugin includes a collection of features for components called widgets and provides access to services based on classification, consistent checking and testing.

Protégé OWL is a tool that makes possible the description of concepts and provides facilities such as: a rich set of operators (and, or and denial); it is based on a logical model that guarantees that the concepts are well described; complex concepts can be described, and the logical model allows the use of a reasoner that checks the consistency described in the ontology ensuring that the hierarchy is correct, known as the Racer reasoner.

3.1 OWL ontology construction

An OWL ontology is composed of individuals, properties and classes. Individuals represent domain objects of interest and are also known as instances. Properties are binary relationships on individuals and can be inverse, transitive or symmetric. OWL classes are understood as sets that contain individuals and can be organized within a hierarchy of classes and subclasses known as taxonomy. Classes are also known as concepts, as they are a concrete representation of them [8].

3.1.1 Class

When starting an OWL ontology there is an initial class called "owl: Thing" that will contain the ontology. To create a class, you must use the corresponding icon in the class browser window as shown in figure 1.

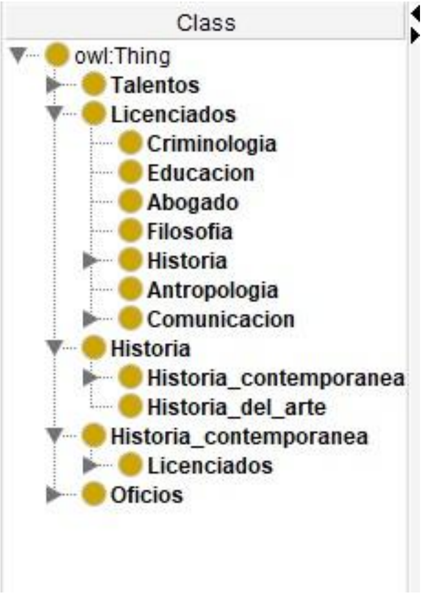


Figure 1: Class creation

3.1.2 OWL Properties

OWL properties represent relationships between two objects or individuals There are two types of properties in OWL: "ObjectProperties", which allows one individual to relate to another, and "DatatypeProperties", which relates an individual to an XML Schema Datatype value or an RDF literal [8]. The figure 2 shows a window that shows how to add properties of the two types mentioned. The elements that an ObjectProperty must have are name, domain and range, where the domain refers to the initial class or classes and the range to the final class or classes that relates that property.

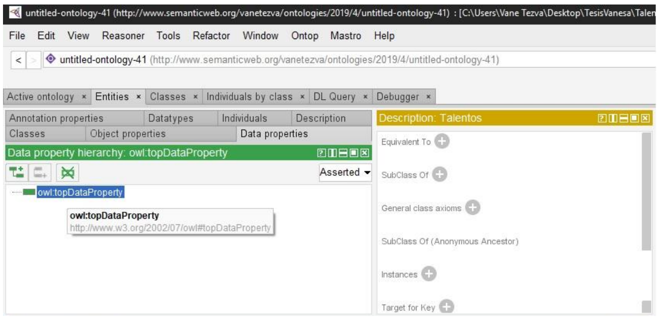

untitled-ontology-41 (http://www.semanticweb.org/vanetezva/ontologies/2019/4/untitled-ontology-41) : [C:\Users\Vane Tezva\Desktop\TesisVanesa\Talen
File Edit View Reasoner Tools Refactor Window Ontop Mastro Help
untitled-ontology-41 (http://www.semanticweb.org/vanetezva/ontologies/2019/4/untitled-ontology-41)
Active ontology × Entities × Classes × Individuals by class × DL Query × Debugger ×
Annotation properties Datatypes Individuals Description
Classes Object properties Data properties
Data property hierarchy: owl:topDataProperty
Asserted
owl:topDataProperty
owl:topDataProperty
http://www.w3.org/2002/07/owl#topDataProperty
Description: Talentos
Equivalent To
SubClass Of
General class axioms
SubClass Of (Anonymous Ancestor)
Instances
Target for Key

Figure 2: OWL properties.

3.1.3 Inverse properties

Any object property or "ObjectProperty" must have a specified domain and range; properties link domain objects to objects in the range, where the domain and range are classes defined in the ontology.

Each ObjectProperty must have its corresponding inverse property. If a property links an object A with another B, then the inverse property links object B with A.

3.1.4 OWL property features

OWL allows the meaning of the properties to be enriched with the characteristics of the properties, which are:

- **Functional properties:** If a property is functional given an object, an object related to another can be at most by this property. For example, if you have three objects that are A, B and C and you have a functional property with Father, then you could associate objects A and B through the property and it would result in A having Father B. You could also associate the objects A and C by means of the property and would result in A Having Father C. Since Father has functional property, it is concluded that B and C are the same object.
- **Inverse functional properties:** If a property is inverse functional, it indicates that an object related to another can be at most by this property inversely to the original functional property. For example, if you have three objects that are A, B and C and you have a functional property isFatherOf, then you could associate the object B and A through the property and it would result in B isFatherOf A. You could also associate the objects C and A by means of the property and would result in C is Father A. As isFatherOf is inverse functional property, it is concluded that B and C are the same object.
- **Transitive properties:** If a property is transitive and relates two objects A and B, and there is also a property that relates object B to another C, then it can be inferred that object A is related to object C through the transitive property.
- **Symmetric properties:** If a property P is symmetric and the property relates to objects A and B, then object B is related by means of property P with object A.

3.2 Definition and description of CLASSES

Once several properties have been created, they can be used to define and describe the behavior of classes [5].

3.2.1 Property restrictions

Properties are used to create restrictions on classes in an OWL ontology. Usually the name of the property should suggest the restrictions imposed on the objects of the class. OWL restrictions are presented in the following three categories:

- Quantification restrictions.
- Cardinality restrictions.
- Value restrictions.

The quantification restrictions consist of the following elements:

- Existential quantifier (ɜ), which allows to indicate the existence of at least one object.
- Universal quantifier (Ɐ), which allows to indicate the existence of all objects.

Given the ontology shown in figure 3, which represents the conceptualization of the university model, different restrictions can be made using previously defined properties and applying existential quantifiers.

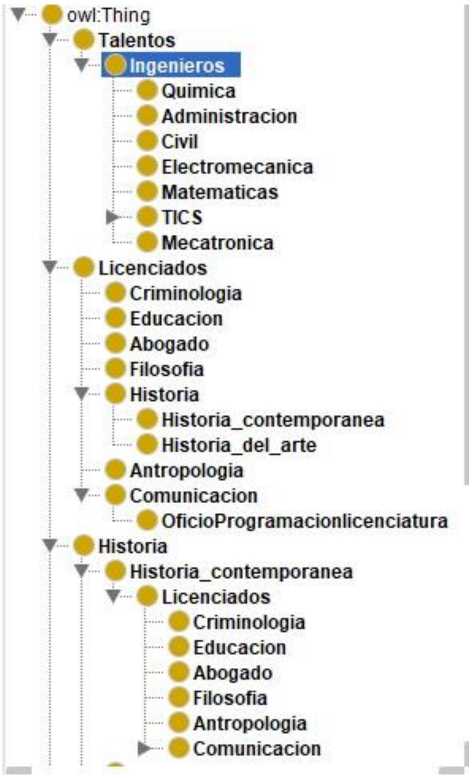
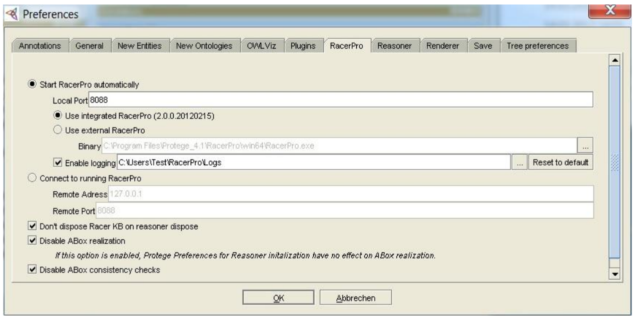


Figure 3: Ontology model

3.3 Reasoning of the ontology

To perform the reasoning of an ontology it is necessary to acquire a tool that acts as a reason for Protégé. There is a tool called Racer that can be connected with Protégé for the analysis and reasoning of ontologies.

To start Racer, run the RacerPro program. A window will appear as shown in the figure 4 indicating that RacerPro is running on the IP address: localhost and on the port: 8088 previously configured.



● Start RacerPro automatically
Local Port 8088
● Use integrated RacerPro (2.0.0.20120215)
○ Use external RacerPro
Binary C:\Program Files\Protege_4.1\RacerPro\win64\RacerPro.exe
☑ Enable logging C:\Users\Test\RacerPro\Logs
○ Connect to running RacerPro
Remote Adress 127.0.0.1
Remote Port 8088
☑ Don't dispose Racer KB on reasoner dispose
☑ Disable ABox realization
If this option is enabled, Protege Preferences for Reasoner initalization have no effect on ABox realization.
☑ Disable ABox consistency checks

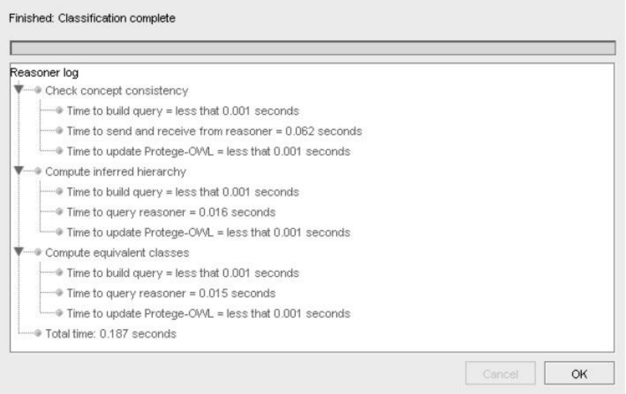
Figure 4: RacerPro configuration

It is necessary to configure the address where the RacerPro reasoner is running in Protégé. To do this, you enter the OWL Preferences window and set the address set in RacerPro.

Racer is used for three main activities that are:

- **Consistency check:** It allows checking that there is consistency between the properties of objects and restrictions assigned to the classes.
- **Taxonomic classification:** Find the ontology hierarchy of the ontology based on the assertion hierarchy, which is the classification obtained in the construction of the ontology.
- **Inferred type computing:** Allows computing the types of objects within the ontology

By executing the taxonomic classification that is Racer's most important operation, Protégé generates a hierarchy of inference in the ontology as shown in the figure 5.



```
Finished: Classification complete

Reasoner log
▼ Check concept consistency
    Time to build query = less that 0.001 seconds
    Time to send and receive from reasoner = 0.062 seconds
    Time to update Protege-OWL = less that 0.001 seconds
▼ Compute inferred hierarchy
    Time to build query = less that 0.001 seconds
    Time to query reasoner = 0.016 seconds
    Time to update Protege-OWL = less that 0.001 seconds
▼ Compute equivalent classes
    Time to build query = less that 0.001 seconds
    Time to query reasoner = 0.015 seconds
    Time to update Protege-OWL = less that 0.001 seconds
  Total time: 0.187 seconds
```

Figure 5: Execution of the taxonomic classification

3.4 OWLviz

OWLViz is a Protégé tool that is installed using a plugin. To make use of this plugin it is necessary to install a tool that allows graphic components of the ontology called Graphviz. Once OWLViz and Graphviz are installed, a tab is generated in the Protégé environment called OWLViz. There is a toolbar that provides different features to the plugin. In this bar there is an icon called options, which you need to access to indicate the path of the Graphviz application [9].

Once all the installed elements are found, the ontology can be generated graphically with the assertion hierarchy or inference hierarchy as shown in the figure 6.

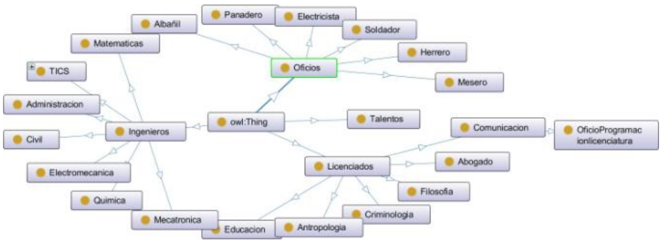


Figure 6: Ontology in the assertion model seen in OWLViz.

4 Results

Based on the research and experiments performed for the preparation of this article, it has been possible to observe the great importance of ontologies in the different applications of solutions based on the semantic web. In addition, if the use of web services is considered for the implementation of the semantics described by the designed ontology, it is necessary to apply the concepts of orchestration and choreography in order to achieve in a synchronized way the correct flow of information between services and in this way obtain a product with a high level of abstraction from the problem domain.

Once the ontology has been prepared using the Protégé tool, the taxonomy that this ontology represents is generated using the RacerPro tool. This taxonomy is based on OWL properties raised in the designed ontology, allowing this process to show the results in terms of inferences generated by the model that describes the problem domain. Based on these results, we proceed to implement the model described with the help of a technological platform that allows the construction of web services, which, when built under a semantic model, become semantic web services.

However, the process of implementing this model implies the need to make use of existing technologies and paradigms, with a lower level of abstraction than ontologies. By performing this step, which becomes necessary in the process of executing a project, one can think that the power that semantic web models offer is simplified.

For the applications to implement all the semantics proposed by an ontology, it may be necessary to build a semantic interpretation platform for said ontology, that is, it is necessary to generate a very high level language that allows to directly encode the provided semantics of the domain represented. Considering this point of view, one might think that the features offered by OWL are not sufficient for the implementation of semantic web systems that provide the complete abstraction of the business logic of the model using the software development languages currently available.

5 Conclusions

The semantic web can be considered an extension of the web of today that allows to manage new technologies to improve the use and access of the resources of the web making use of concepts such as OWL and WSDL (Web Service Definition Language), which support an ontological language to define and classify concepts in a domain. These can be implemented through web services.

- An ontology is a hierarchy of concepts with attributes and relationships that creates a classification to define the semantics of interrelated units of information. Ontologies provide an understanding of a domain that can be communicated between people and heterogeneous systems.
- Protégé is an ontology development tool that allows the construction of OWL ontologies taking into account all the characteristics of the domain. Through Protégé the development of properties and restrictions allows to generate inference about the domain, generating conceptualization and knowledge of it.
- Racer allows you to quickly verify an ontology taking into account elements of consistency, in addition to performing the taxonomic classification called the inference hierarchy, derived from the model built in the ontology called the assertion hierarchy. Based on these elements, the importance of ontology within the concept of the semantic web becomes understandable.
- OWLViz is a Protégé tool that allows graphic representations of the ontology either in assertion or inference models.

References

[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The Semantic web," Scientific American, May 2001.

[2] K. K. Breitman, M. A. Casanova, and W. Truszkowski, Semantic Web: Concepts, Technologies and Applications: Springer, 2007.

[3] Guarino, N. (1998). Formal Ontology and Information Systems. In the Proceedings of Formal Ontology in Information Systems. Taken from World Wide Web: www.loa-cnr.it/Papers/FOIS98.pdf

[4] Klein, M., et al. (2002). Ontologies and Schema Languages on the Web, in Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential. MIT Press, pp. 95 - 135.

[5] Getting Started with Protégé (2003). Taken from World Wide Web: http://protege.stanford.edu/doc/tutorial/index.html

[6] Héctor Arturo Flórez Fernández. (Diciembre, 2007). Construcción de ontologías OWL. Vinculos, 1, 19-34.

[7] Knublauch, H. (2005). The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications. Stanford University. Taken from World Wide Web: www.scf.usc.edu/~csci586/ppt-2005/bhavin.ppt.

[8] Horridge, M. (2004). A Practical Guide to Build OWL Ontologies.

[9] Knublauch, H. (2005). The Protégé OWL Plugin: An Open Development Environment for Semantic Web Applications. Stanford University. Taken from World World Wide Web: wwwscf.usc.edu/~csci586/ppt-2005/bhavin.ppt.

Author Profile

Vanesa Tenopala Zavala: Graduated as a computer technician from the Colleges of Scientific and Technological Studies (CECyTE) Plantel 07 Ahuashuatepec., Obtained her career as a computer engineer at the Autonomous University of Tlaxcala and currently studies the master's degree in computer systems at the Technological Institute of Apizaco.

José Crispín Hernández Hernández He obtained the title of Lic. In Computing by the Technological Institute of Apizaco (ITA), Apizaco, Tlax., Mexico, in 1993. The degree of M. in C. in Computational Sciences by the ITA, Apizaco, Tlax., Mexico, in 1998. And the degree of Doctorate in Informatique by the Université d'Angers, France in 2008. His areas of research include: metaheuristics optimization, bioinformatics, biomedicine, fuzzy logic, computer vision.

Perfecto Malaquías Quintero Flores. Industrial engineer in production (manufacture) by the Instituto Tecnológico de Apizaco (ITA), Apizaco, Tlaxcala, México. Master's degree in computer science by the centro de graduados e investigación del Instituto Tecnológico de Toluca, Metepec, Edo. De México. PhD in computational sciences in l'Université Montpellier 2 (UM2) in the laboratory for computational sciences, robotics and microelectronic of l'Université Montpellier 2 and the national center for scientific research of France (LIRMM-CNRS) in Languedoc Roussillon, France. He obtained the certification in PSP, by Software Engineering Institute – Carnegie Mellon University, on June 2014.