A PROTÉGÉ PLUG-IN-BASED SYSTEM TO MANAGE AND QUERY LARGE DOMAIN ONTOLOGIES

E. Jiménez-Ruiz[1], V. Nebot[1], R. Berlanga[1], I. Sanz[2], A. Rios[3]

[1]Departamento de Lenguajes y Sistemas Informáticos
[2]Departamento de Ingeniería y Ciencias de la Computación
(Temporal Knowledge Bases Group, http://krono.act.uji.es/)
Universitat Jaume I, Castellón, Spain
e-mail: {ejimenez, victoria.nebot, berlanga, isanz}@uji.es
[3]Maat Gknowledge, Valencia, Spain
e-mail: arios@maat-g.com

1. INTRODUCTION: MOTIVATION ISSUES

Nowadays, several efforts are focused on building very large ontologies that are continuously growing as new knowledge is added to them by the respective communities. This happens mainly in the biomedical domain (e.g. GO, GALEN, FMA, NCI, Tambis, etc.). However, the very large size of these ontologies as well their variety makes it difficult to deploy them in particular applications. A first problem is scalability. Most of these ontologies are expressed in OWL-DL, with different degrees of expressivity. The classification of new concepts, queries and assertions requires the use of reasoners (e.g. Fact, Pellet, etc.), but they are not able to handle even medium-size ontologies [1]. A second problem is the use of these ontologies in concrete applications. Such applications usually do not require comprehensive descriptions of the domains but rather a handful subset of concepts and properties from them (i.e. a local view of the domain [2]). Another important issue is their visualization in ontology editors, in which large ontologies have difficulties to be loaded and properly displayed. In this paper, a tool[i] extending Protégé-OWL has been developed with the aim of allowing the management of a collection of related ontologies and the extraction of personalized modules (views) by means of a query language named OntoPath [3].

2. ONTOLOGY MANAGEMENT SYSTEM ARCHITECTURE

The architecture of the Ontology management System is presented in Figure 1, in which we can identify four main components:

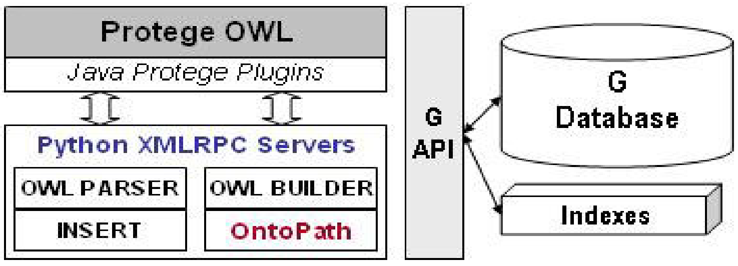


Fig. 1. Architecture of the Ontology Management System

- *OWL parser and constructor*: in order to provide our application with a greater flexibility and storage capabilities (e.g. indexes), a SAX-based parser and a constructor have been implemented. The OWL parser creates from the OWL file a set of structures for classes, properties, nominal and individuals, which will be stored in the graph-based database G.
- *G database:* the database G [4] has been used as a backend to store, index and retrieve the OWL ontologies as graphs. To store the ontologies we use four database object types, namely: *ontology*, *property*, *concept*, and *enumeration*; the latter for nominal lists. Figure 2 summarizes the four object types and their relationships through references. All the OWL constructors are regarded in these object types so that ontologies can be loaded and retrieved from the database without semantic lost.

[i] Ontology Management System: http://krono.act.uji.es/people/Ernesto/G_Protege_Plugin

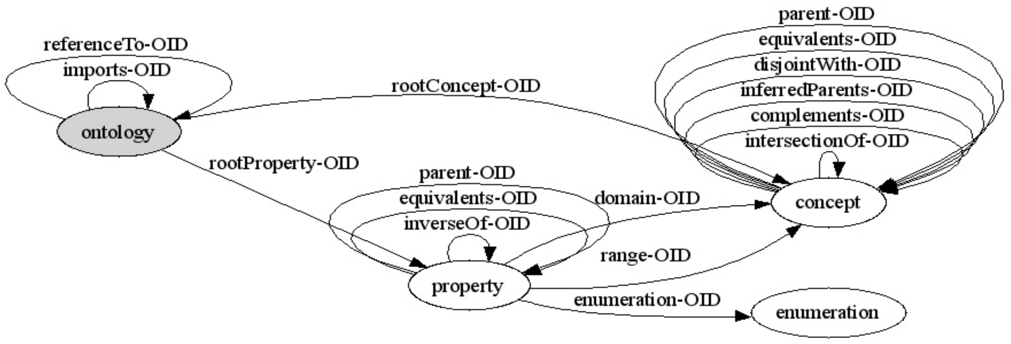


Fig. 2. Meta-schema for storing OWL Ontologies into G.

- *OntoPath* is a query language for retrieving consistent fragments (personalized modules or views) from domain ontologies. By using OntoPath, users can specify the desired detail level in the concept taxonomies as well as the properties between concepts that are required by the target applications. The syntax and aims of OntoPath resemble XPath's in that they are simple enough to be handled by non-expert users and they are designed to be included in other XML-based applications (e.g. transformations sheets, semantic annotation of web services, etc.). Figure 3 shows the interpretation of a simple OntoPath query like **Disease / related_to / Rheumatoid_Factor**, where an ontology fragment is extracted with *diseases* (if exist) directly related by means of the *related_to* property to the *Rheumatoid Factor* concept. For further information about the syntax and semantics of OntoPath see [3].

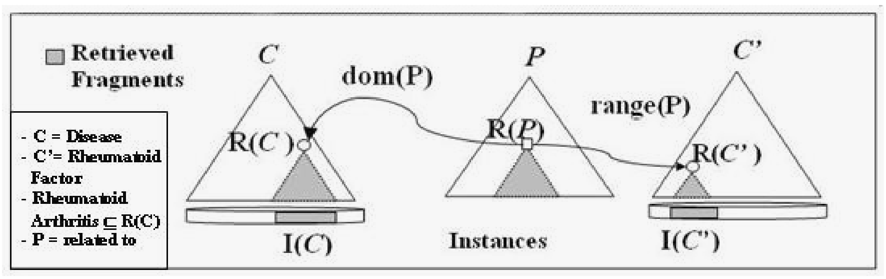


Fig. 3. Graphical interpretation of the OntoPath query C/P/C'.

- *Protégé-OWL* has been selected as the front-end to visualize and manipulate ontologies due to its flexible interface for developing plug-ins, making easy the extension of all its functionalities. The main characteristics of the extension and the developed interfaces are commented in the next section.

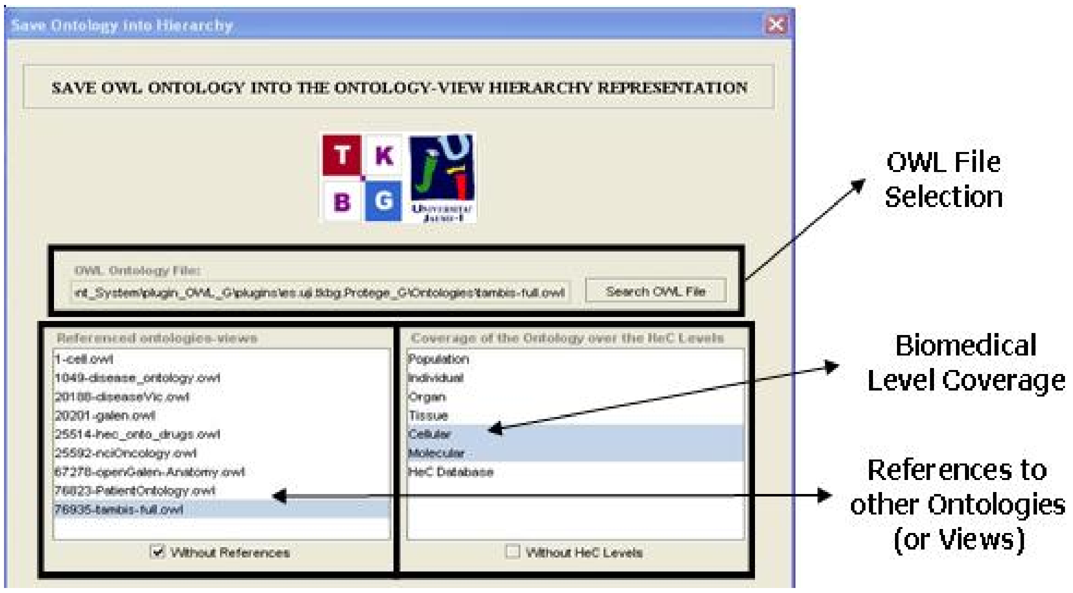
3. PROTÉGÉ-OWL EXTENSIONS

Protégé-OWL has been extended in order to provide a couple of interfaces to allow the storage and retrieval of entire OWL ontologies (or fragments) in/from G databases. A new menu component has been added to the Protégé interface for these purposes.

3.1. Storing Ontologies

OWL ontologies are parsed in order to create the necessary structures which will be inserted in the database. The developed system provides an interface (see Figure 4) to indicate the OWL file to be inserted (or the current ontology loaded in Protégé-OWL). In this interface we can also indicate some meta-information about the ontology like the coverage over a set of pre-defined modules (in this case the biomedical vertical levels[ii]) and the references to other ontologies. Let us emphasize that all fragments defined with OntoPath will have also a reference to the source ontologies.

[ii] Example taken from the Health-e-Child project (http://www.health-e-child.org/).


Save Ontology into Hierarchy
SAVE OWL ONTOLOGY INTO THE ONTOLOGY-VIEW HIERARCHY REPRESENTATION
OWL Ontology File:
Search OWL File
Referenced ontologies-views
Coverage of the Ontology over the HeC Levels
Without References
Without HeC Levels
OWL File Selection
Biomedical Level Coverage
References to other Ontologies (or Views)

nt_System\plugin_OWL_G\plugins\es.uji.tkbg.Protege_G\Ontologies\tambis-full.owl

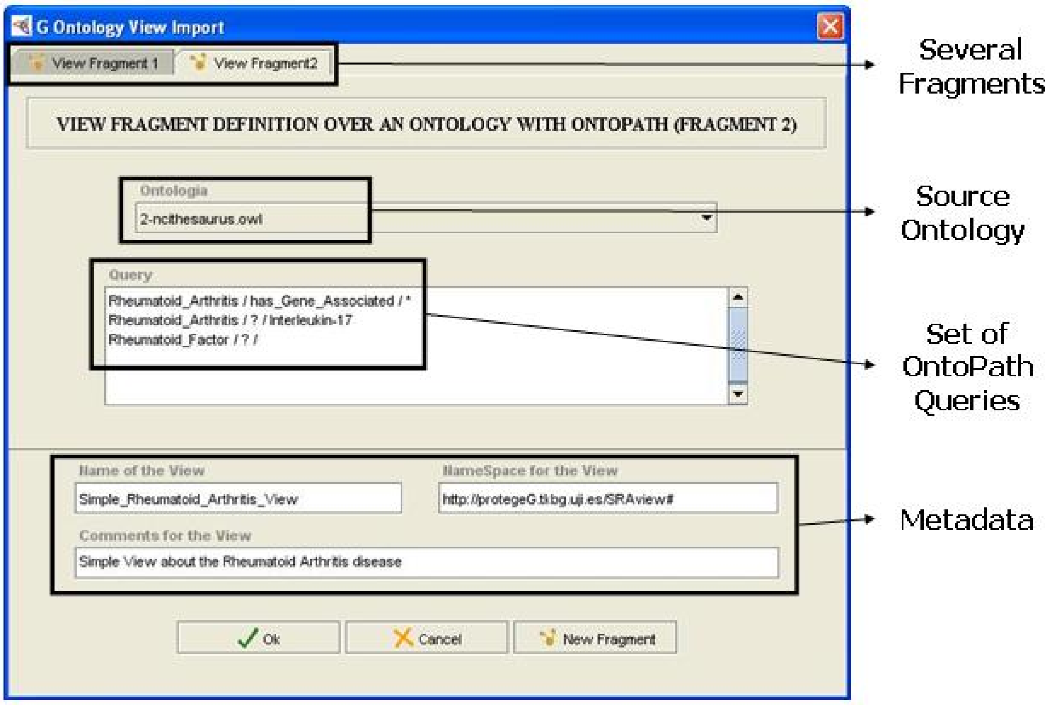
Referenced ontologies-views	Coverage of the Ontology over the HeC Levels
1-cell.owl	Population
1049-disease_ontology.owl	Individual
20188-diseaseVic.owl	Organ
20201-galen.owl	Tissue
25514-hec_onto_drugs.owl	Cellular
25592-nciOncology.owl	Molecular
67278-openGalen-Anatomy.owl	HeC Database
76823-PatientOntology.owl	
76935-tambis-full.owl	

[✓] Without References [ ] Without HeC Levels

Fig. 4. Interface to store OWL ontologies in G databases.

3.2 Retrieving full ontologies and defining fragments

The developed plug-in has also the capability of retrieving ontologies stored in G databases, extracting all their objects and reconstructing the OWL file by means of the OWL builder module. Moreover, as commented before, the system provides a framework to define fragments by means the query language OntoPath. With this language we can extract only the desired objects of the ontologies which are arranged in a new personalized OWL file. Figure 5 shows the Protégé interface to define the OntoPath queries over one or more ontologies.



Ontologia: 2-ncithesaurus.owl

Query:

```
Rheumatoid_Arthritis / has_Gene_Associated / *
Rheumatoid_Arthritis / ? / Interleukin-17
Rheumatoid_Factor / ? /
```

Name of the View: Simple_Rheumatoid_Arthritis_View

NameSpace for the View: http://protegeG.tkbg.uji.es/SRAview#

Comments for the View: Simple View about the Rheumatoid Arthritis disease

Fig. 5. Interface to define OntoPath queries.

It is worth mentioning that the proposed tool also provides view mechanisms for frame-based ontologies. In this case, the frame-based version of Ontopath, OntopathView [5], is used instead.

3.3 Representation of stored ontologies and fragments

Another facility provided by the system is the organization of ontologies and views (i.e. fragments) in a definition hierarchy and the classification into biomedical levels, as it can be seen in Figure 6. This provides the user with a global vision of the ontologies and fragments stored in the databases making easy their retrieval.

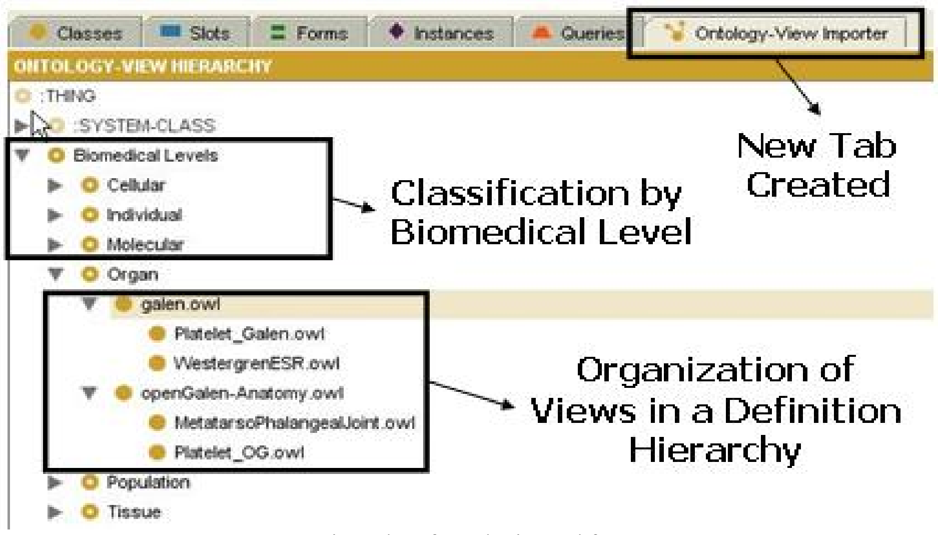


Fig. 6. Hierarchy of ontologies and fragments.

ACKNOWLEDGEMENTS

This work has been partially funded by the PhD Fellowship Program of the Generalitat Valenciana, the CICYT Project TIN2005-09098-C05-04 from the Spanish Ministry of Education and Science, and the Health-e-Child European Project.

REFERENCES

[1] J. Seidenberg, A. Rector "Web Ontology Segmentation: Analysis, Classification and Use", Proceedings of the 15th international conference on World Wide Web, WWW 2006

[2] P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt, "COWL: Contextualizing Ontologies", ISWC-2003, LNCS vol. 2870, pp. 164-179, Springer Verlag.

[3] E. Jiménez-Ruiz, V. Nebot, R. Berlanga, I. Sanz: "OntoPath: A Language for Retrieving Ontology Fragments". Submitted to OTM-ODBASE 2007. http://krono.act.uji.es/publications/techrep

[4] A. Rios "G Platform (Helide S.A)": http://www.maat-g.com

[5] E. Jiménez, R. Berlanga, I. Sanz, M. J. Aramburu, R. D.: OntoPathView: A Simple View Definition Language for the Collaborative Development of Ontologies. En B. López et al. (Eds.): Artificial Intelligence Research and Development, pages 429-436. IOS Press, 2005.