Ontology Engineering Group
Laboratorio de Inteligencia Artificial
Dept. Inteligencia Artificial, Facultad de Informática – UPM

OEG Publication

Corcho O, Gómez-Pérez A, González-Cabero R, Suárez-Figueroa MC

ODEVAL: A TOOL FOR EVALUATING RDF(S), DAML+OIL, AND OWL CONCEPT TAXONOMIES

1st IFIP Conference on Artificial Intelligence Applications and Innovations
(AIAI 2004)
Artificial Intelligence Applications and Innovations
August 2004.
Toulouse, France.
Pages: 369 to 382

ODEVAL: A TOOL FOR EVALUATING RDF(S), DAML+OIL, AND OWL CONCEPT TAXONOMIES

Óscar Corcho, Asunción Gómez-Pérez, Rafael González-Cabero and M. Carmen Suárez-Figueroa
Laboratorio de Inteligencia Artificial. Facultad de Informática. Universidad Politécnica de Madrid. Campus de Montegancedo sn. Boadilla del Monte, 28660. Madrid, Spain

Abstract: Ontologies implemented in RDF(S), DAML+OIL, and OWL should be evaluated from the point of view of knowledge representation before using them in Semantic Web applications. Several language-dependent ontology validation tools and ontology platforms, such as OilEd with FaCT, can be used in order to evaluate RDF(S), DAML+OIL and OWL ontologies. This paper offers two main contributions. The first of these exams whether previous ontology tools detect knowledge representation problems in RDF(S), DAML+OIL, and OWL concept taxonomies. Indeed, such tools do not focus on detecting inconsistencies and redundancies in concept taxonomies. The second contribution is ODEval, a language-dependent tool for evaluating, from the point of view of knowledge representation, concept taxonomies in ontologies implemented in such languages. ODEval complements previous ontology tools when we want to evaluate RDF(S), DAML+OIL, and OWL concept taxonomies.

Key words: Ontology Evaluation; Ontology Tools; Concept Taxonomies

1. INTRODUCTION

Like any other resources used in software applications, ontology content needs to be evaluated before (re)using it in other ontologies or applications. Moreover, content evaluation as well as evaluation of the software environments used to build ontologies are critical processes before ontologies can be integrated in final applications.

Ontology evaluation is a crucial activity, which needs to be carried out during the whole ontology life-cycle. The goal of this evaluation is to determine what the ontology correctly defines, does not define at all, or even incorrectly defines. Few domain-independent methodological approaches (Fernández-López et al., 1999; Gómez-Pérez et al., 2003) have been reported for building ontologies, nevertheless all of them identify the need for ontology evaluation, but the evaluation is performed differently in each one.

The first work on ontology content evaluation started in 1994 and in the past three years the interest on this topic has grown. The principal efforts were made by Gómez-Pérez (2001) and by Guarino and Welty (2000).

Along with the increasing number of ontologies implemented in the ontology languages RDF(S)[1, 2], DAML+OIL[3], and OWL[4], certain specialized ontology parsers, namely Validating RDF Parser[5], RDF Validation Service[6], DAML Validator[7], DAML+OIL Ontology Checker[8], OWL Ontology Validator[9], and OWL Validator[10], and import services within ontology platforms, namely OilEd (Bechhofer et al., 2001), OntoEdit[11], Protégé-2000[12], and WebODE[13] have been built. These ontology tools must be studied in order to analyze whether they detect, from the point of view of knowledge representation, possible taxonomic problems, like inconsistencies and redundancies, in ontologies implemented in such languages.

In (Gómez-Pérez and Suárez-Figueroa, 2003, 2004) we describe how several ontology tools evaluate RDF(S) and DAML+OIL concept taxonomies. In this paper we detail our study with 41 ontologies, which are well-built from a syntactic point of view, but have inconsistencies and redundancies in their concept taxonomies. We have parsed these ontologies with the previous ontology parsers and have imported them into OilEd using its import service and connecting it to the reasoning engine FaCT (Horrocks et al., 1999). It has been discovered that, in the majority of the experiments, these ontology tools do not detect the taxonomic problems identified in (Gómez-Pérez, 2001). For this reason, we have built ODEval[14] as a

[1] http://www.w3.org/TR/PR-rdf-schema
[2] http://www.w3.org/TR/REC-rdf-syntax/
[3] http://www.daml.org/2001/03/daml+oil-walkthru.html
[4] http://www.w3.org/TR/owl-ref/
[5] http://139.91.183.30:9090/RDF/VRP/
[6] http://www.w3.org/RDF/Validator/
[7] http://www.daml.org/validator/
[8] http://potato.cs.man.ac.uk/oil/Checker
[9] http://phoebus.cs.man.ac.uk:9999/OWL/Validator
[10] http://owl.bbn.com/validator/
[11] http://www.ontoprise.de/products/ontoedit_en
[12] http://protege.stanford.edu/
[13] http://babage.dia.fi.upm.es/webode/
[14] http://minsky.dia.fi.upm.es/odeval

complement to the previous ontology tools. ODEval performs syntactic evaluation of RDF(S), DAML+OIL, and OWL ontologies, and evaluates their concept taxonomies from the point of view of knowledge representation using the ideas proposed in (Gómez-Pérez, 2001).

This paper is organized as follows: section two briefly presents possible anomalies that can appear in taxonomic knowledge; section three presents a brief description of the RDF(S), DAML+OIL, and OWL tools used in our experiments; section four includes our comparative study; section 5 describes ODEval; and finally, we conclude with further work on evaluation.

2. EVALUATING TAXONOMIC KNOWLEDGE IN ONTOLOGIES

Figure 1 presents a set of possible problems that can appear when ontologists model taxonomic knowledge in ontologies (Gómez-Pérez, 2001).

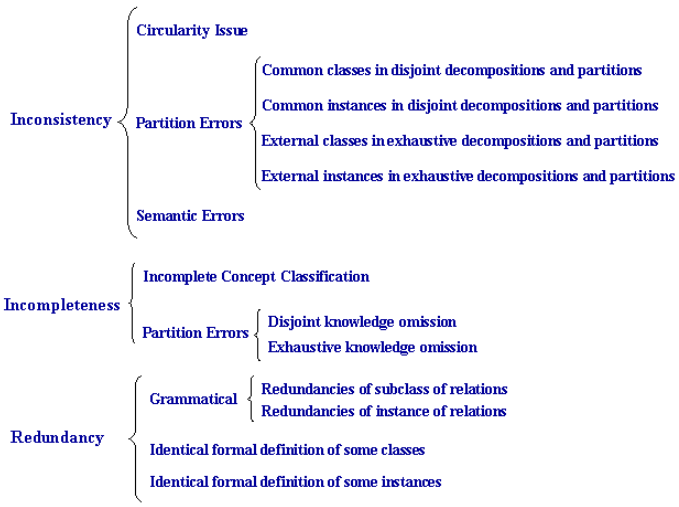

Figure 1. Potential problems that might appear in taxonomies

In this paper we focus on the automatic detection of inconsistencies (circularity issue and partition errors) and redundancy grammatical problems. We postpone the analysis of the others for further works.

3. RDF(S), DAML+OIL, AND OWL ONTOLOGY TOOLS

Currently, there are several ontology parsers and ontology platforms which can be used in order to evaluate RDF(S), DAML+OIL, and OWL ontologies. In this paper, we focus on the following ontology tools:

The ICS-FORTH **Validating RDF Parser** (VRP v2.5)[15] analyzes, validates and processes RDF schemas and resource descriptions. This parser offers s*yntactic validation* for checking if the input namespace conforms to the updated RDF/XML syntax proposed by W3C, and s*emantic validation* for verifying constraints derived from RDF Schema Specification (RDFS).

The W3C **RDF Validation Service**[16] is based on HP-Labs Another RDF Parser (ARP[17]), which currenlty uses the version 2-alpha-1. This service supports the Last Call Working Draft specifications issued by the RDF Core Working Group, including datatypes. It offers s*yntactic validation* for checking if the input namespace conforms to the updated RDF/XML syntax proposed by W3C. However, this service does not do any RDFS validation.

The **DAML Validator**[18] uses the ARP parser from the Jena (1.6.1) toolkit to create an RDF triple model from the input code being validated. This validator offers s*yntactic validation* for checking for namespace problems during model creation, and also tests RDF resources for existence. This tool performs s*emantic validation* for verifying the global domain and range constraints of the predicate.

The **DAML+OIL Ontology Checker**[19] is a servlet that uses the OilEd codebase to check the syntax of DAML+OIL ontologies. This checker is a web interface to check DAML+OIL ontologies and content using Jena. It offers s*yntactic validation* for checking missing definitions, and s*emantic validation* for verifying class hierarchy loops.

The **OWL Ontology Validator**[20] can be used to check if an ontology conforms to a specific OWL species, since it validates an OWL ontology and reports as a result the OWL language species to which the ontology belongs: OWL Lite, OWL DL, and OWL Full. Besides, if requested, the validator returns a description of the classes, properties and individuals in the ontology in terms of the OWL Abstract Syntax.

---

[15] http://139.91.183.30:9090/RDF/VRP/index.html
[16] http://www.w3.org/RDF/Validator/
[17] ARP was created and is maintained by Jeremy Carroll at HP-Labs in Bristol
[18] http://www.daml.org/validator/
[19] http://potato.cs.man.ac.uk/oil/Checker
[20] http://phoebus.cs.man.ac.uk:9999/OWL/Validator

The **OWL Validator**[21] is based on the DAML Validator[22] (it uses a modified version of the Jena Toolkit). This tool is not a simple parser in the sense that it checks OWL ontologies not only for problems related to simple syntax errors, but also for other potential errors. The OWL Validator does not aim at perfoming full reasoning or inferencing, but only at checking these kinds of problems.

OilEd[23] (Bechhofer et al., 2001) was initially developed as an ontology editor for OIL ontologies, in the context of the European IST OntoKnowledge Project. However, OilEd has evolved and now is an editor of DAML+OIL and OWL ontologies. The current version is 3.5.

OilEd can connect to any DL reasoner that uses the interface described in the DL Implementation Group (DIG) reasoner interface (Bechhofer, 2002). Currently, it can connect to reasoning engines such as FaCT (Horrocks et al., 1999) to detect inconsistencies in class taxonomies.

OilEd can import ontologies implemented in RDF(S), OIL, DAML+OIL, OWL, and the SHIQ XML format.

4. COMPARATIVE STUDY OF RDF(S), DAML+OIL, AND OWL ONTOLOGY TOOLS

As we said before, the first goal of this paper is to analyse whether tools presented in section 3 detect problems presented in section 2.

We have built a testbed of 41 ontologies (7 in RDF(S), 17 in DAML+OIL, and 17 in OWL), each of which implements one of the problems presented in section 2. In the case of RDF(S) we have only 7 ontologies because partition knowledge cannot be defined in this language. These ontologies and the results of their evaluation can be found at http://minsky.dia.fi.upm.es/odeval. Any user can also evaluate its ontologies in that URL, since the ODEval[24] tool is publicly available there.

We have parsed our 41 ontologies using the ontology tools presented in section 3. All these tools recognised the code as well formed code, but the majority had problems detecting most of the knowledge representation problems that these ontologies contained.

The results of analysing and comparing these ontology tools are shown in table 1, and the symbols used in this table are the following:

✓: The ontology tool detects the problem in this language

×: The ontology tool does not detect the problem in this language

[21] http://owl.bbn.com/validator/
[22] http://www.daml.org/validator/
[23] http://oiled.man.ac.uk
[24] http://minsky.dia.fi.upm.es/odeval

--: The problem cannot be represented in this language

☑: The problem can be inserted in the ontology tool, which detects it when the ontology is verified

As we can see in table 1[25]:

- *Circularity problems* are detected by some of the ontology tools studied in this experiment. In particular, VRP is able to detect circularity at any distance in RDF(S) ontologies, indicating that there is a semantic error with the following message "loop detected". The DAML+OIL Ontology Checker identifies circularity at any distance in DAML+OIL ontologies, throwing the following warning "cycles in class hierarchy". And OilEd warns in its log that there are loops (circularity problems at any distance) in RDF(S) and DAML+OIL ontologies using this message "Cycles in class hierarchy!". However, OilEd imports the ontology and shows the inconsistency.
- Regarding *partition errors*, they have only been studied for DAML+OIL and for OWL, since they cannot be represented in RDF(S). None of the ontology parsers have detected partition errors with the DAML+OIL and OWL ontologies. Common classes in disjoint decompositions and partitions can be inserted in OilEd, but when we use FaCT, OilEd marks in red the wrong class and shows this message "$Class_i$ is an unsatisfiable class".
- As for *grammatical redundancy problems*, they are not detected by any of the ontology tools studied.

5. ODEVAL: A FRAMEWORK TO EVALUATE CONCEPT TAXONOMIES

As we can see in section 4, only a few ontology tools are able to detect loops in concept taxonomies in RDF(S), DAML+OIL, and OWL ontologies. Furthermore, only OilEd (connecting to the reasoning engine FaCT) is able to identify a few of partition errors in concept taxonomies. And regarding grammatical redundancy problems, none of the ontology tools detect them.

[25] In table 1 we use the following abbreviation:
VRP: Validating RDF Parser; RDF V. S.: RDF Validation Service; DAML V.: DAML Validator; DAML+OIL O. C.: DAML+OIL Ontology Checker; OWL O. V.: OWL Ontology Validator; OWL V.: OWL Validator;
R: RDF(S); D+O: DAML+OIL; O: OWL;
d. d.: disjoint decompositions; e. d.: exhaustive decompositions; p.: partitions; s-o: subclass-of; i-o: instance-of; Dir.: Direct; Ind.: Indirect

Table 1. Results of the analysis of the RDF(S), DAML+OIL, and OWL ontology tools

			VRP	RDF V. S.	DAML V.	DAML+OIL O. C.	OWL O. V.	OWL V.	OilEd + FaCT		
			R	R	D+O	D+O	O	O	R	D+O	O
Inconsistency: Circularity Problems	At distance zero		✓	×	×	✓	×	×	✓	✓	×
	At distance one		✓	×	×	✓	×	×	✓	✓	×
	At distance N		✓	×	×	✓	×	×	✓	✓	×
Inconsistency: Partition Errors	Common classes in d. d.	Dir.	--	--	×	×	×	×	--	☑	☑
		Ind.	--	--	×	×	×	×	--	☑	☑
	Common classes in p.		--	--	×	×	×	×	--	☑	☑
	Common instances in d. d.	Dir.	--	--	×	×	×	×	--	×	×
		Ind.	--	--	×	×	×	×	--	×	×
	Common instances in p.		--	--	×	×	×	×	--	×	×
	External classes in e. d.		--	--	×	×	×	×	--	×	×
	External classes in p.		--	--	×	×	×	×	--	×	×
	External instances in e. d.		--	--	×	×	×	×	--	×	×
	External instances in p.		--	--	×	×	×	×	--	×	×
Redundancy: Grammatical Problems	Redundancies of ‘s-o’ relations	Dir.	×	×	×	×	×	×	×	×	×
		Ind.	×	×	×	×	×	×	×	×	×
	Redundancies of ‘i -o’ relations	Dir.	×	×	×	×	×	×	×	×	×
		Ind.	×	×	×	×	×	×	×	×	×

Consequently, we have decided to build ODEval[26], a tool for evaluating RDF(S), DAML+OIL, and OWL ontologies from a knowledge representation point of view using the ideas proposed in (Gómez-Pérez, 2001). ODEval is a complement to the previous ontology tools when we want to evaluate RDF(S), DAML+OIL, and OWL concept taxonomies, from a knowledge representation point of view.

In this section, we describe the algorithms, based on graph theory, used in ODEval to detect possible problems in ontology concept taxonomies. The concept taxonomy will be considered as a directed graph *G(V,A)*, where *V* is a set of nodes (vertex) and *A* is a set of directed arcs. The elements included in the sets *V* and *A* will be different depending on each language and on each type of problem that we want to detect.

5.1 RDF(S) Evaluation in ODEval

In RDF(S), the only primitive that can be used to express specialization/generalization between classes is *rdfs:subClassOf*. We cannot define disjoint nor exhaustive knowledge with any of the primitives of the language. Consequently, the only problems that can exist in RDF(S) ontologies are circularity and redundancy.

5.1.1 Circularity Problems

In order to detect circularity problems, the graph *G(V,A)* will contain in *V* the set of named and anonymous classes of the ontology, and in *A* the set of all the *rdfs:subClassOf* relations between classes in the ontology. To detect these problems, ODEval looks for cycles in the graph *G*.

5.1.2 Redundancy Problems

In this case, the graph *G(V,A)* will contain in *V* the set of named and anonymous classes and instances of the ontology, and in *A* the set of all the *rdfs:subClassOf* relations and "instance-of" relations. To detect grammatical redundancy problems in concept taxonomies, we define the predicate *reachablesFrom(G,v,R)* as the adjacent elements of the vertex *v* in the transitive closure of the graph *G* using the arcs that belong to the relation types set *R*. In other words, all the vertices v_i for which we are able to find a path of arches of a type that belongs to R that begins in the vertex *v* and ends in each vertex v_i. For each class *class_A* in the set *V*, and for each arc r_i in the set *A* whose origin is *class_A*, we take r_i out of the set *A* and check

[26] http://minsky.dia.fi.upm.es/odeval

whether this change affects the set of elements that are reachable from the *class_A*. If there is no change, this means at least one of the r_i is dispensable.

Formally, this can be defined as:

$$\forall r_i \in \{r_1 ... r_N\} \left| \begin{array}{l} reachablesFrom(G, class\_A, R) \supset reachablesFrom(\widetilde{G}_i, \\ class\_A, R) \end{array} \right.$$

where $\widetilde{G}_i$ is the graph *G* without the arc of the relation we want to check,

$$\widetilde{G}_i \equiv G_i(\widetilde{V}_i, \widetilde{A}_i) = \begin{cases} \widetilde{A}_i \equiv A - \{r_i\} \\ \widetilde{V}_i \equiv V \end{cases}$$

5.2 DAML+OIL Evaluation in ODEval

In DAML+OIL, the set *V* of the graph *G(V,A)* will contain both named and anonymous classes, and instances (in the case of partition errors and redundancy problems). Regarding the arcs to be included in the set *A*, we will distinguish between the specialization/generalization primitives (*rdfs:subClassOf*, *daml:disjointUnionOf*, *daml:intersectionOf*, and *daml:unionOf*), class equivalence primitives (*daml:sameClassAs* and *daml:equivalentTo*) and the "instance-of" relations between instances and classes. We have not considered the primitive *daml:complementOf*.

5.2.1 Circularity Problems

To detect circularity problems, the set *A* of the graph *G(V,A)* will contain specialization/generalization and class equivalence primitives. The direction of each arc is the same as the direction of the relation, except for *daml:unionOf* and *daml:disjointUnionOf*, where it is the opposite.

ODEval looks for the following types of cycles in the graph *G*:

- *Mixed cycles*. The arcs of these cycles contain class equivalence relations and specialization/generalization relations. In this case, we have found a circularity.
- *Equivalence cycles*. All the arcs of the cycle are class equivalence relations. In this case, we have found a redundancy problem, that is, one of the class equivalence relations could be removed.

5.2.2 Partition Errors

5.2.2.1 Common classes and common instances in disjoint decompositions and partitions

ODEval uses the same evaluation algorithm for these partition errors. Detecting these errors is not as straightforward as only searching common direct classes or instances of the concepts that form the partition, but also recursively checking it in their subclasses.

The set *V* of the graph *G* will contain the classes (named and anonymous) and instances defined in the ontology. The set *A* will not only contain the specialization/generalization and class equivalence relations, but also arcs that connect each instance with the class to which it belongs.

We define the predicate *reachablesTo(G,v,R)*, which is the reverse of *reachablesFrom,* that is, it changes the directions of the arcs of the directed graph and looks for paths in the opposite direction. To know whether an element *element* (class or instance) of the ontology belongs to more than one path of a disjoint decomposition or a partition, composed by $\{Class\_P_1^i, Class\_P_2^i, ..., Class\_P_N^i\}$, we must check that we can only reach one of the classes of the decomposition through the set of relations *A*. This is formally defined as follows:

$$card\left\{x \left| \begin{array}{l} x \in reachablesFrom(G, element, R) \wedge x \in \\ \{Class\_P_1^i, ..., Class\_P_N^i\} \end{array} \right.\right\} = 1$$

If we are interested in checking whether this error occurs in a disjoint decomposition or a partition, formed by the classes $\{Class\_P_1, Class\_P_2, ..., Class\_P_n\}$, we must check that there are no common elements in two or more branches of the partition. This can be expressed as follows:

$$\forall class1, class2 \in \{Class\_P_1, ..., Class\_P_n\}$$
$$\left| (class1 \neq class2) \rightarrow \begin{pmatrix} reachablesTo(G, class1, R) \\ \cap (reachablesTo(G, class2, R)) = \phi \end{pmatrix} \right.$$

5.2.2.2 External classes and external instances in exhaustive decompositions and partitions

As in the previous case, ODEval uses the same algorithm for both types of errors and the same information is included in the sets *V* and *A* of *G*.

The problem that arises here, is the existence in DAML+OIL of class equivalence relations. If the class *Class_A* to be checked has equivalent classes, then each of them must be checked separately. *E* contains the set of classes equivalent to the base class:

$$E = reachablesTo(G, Class\_A, \{equivalentTo, sameClassAs\})$$

We call *Class_A'*$_i$ to each of the classes that belong to this set

$$E = \{Class\_A'_1, Class\_A'_2, ..., Class\_A'_M\}$$

We also define *A'* as *A* minus the class equivalence relations.
We can conclude that a base class *Class_A* has not external elements if:

$$\bigcup_{i=1}^{M} reachables\ To(G, Class\_A'_i, R') \subseteq \bigcup_{i=1}^{N} reachables\ To(G, Class\_P_i, R)$$

The following two errors can be found with this algorithm:

- *Partition error*. If the element is only reachable from the base class (or its equivalents) and it is not reachable from the classes of the decomposition. Depending on the type of the element, class or instance, we have an external class or external instance.
- *Redundancy problem*. If the element is reachable both from the base class (or its equivalents) and from one of the classes of the decomposition.

5.2.3 Redundancy Problems

Apart from the redundancy problems described in the previous sections, we can find redundancy in the same way described with RDF(S) (section 5.1.2), using the predicate *reachablesFrom(G,v,R)* and detecting whether removing an arc from the set *A* implies any change in the set of elements reachable from a vertex.

5.3 OWL Evaluation in ODEval

In OWL, the set *V* of the graph *G(V,A)* will contain both named and anonymous classes, and instances (in the case of partition errors and redundancy problems). Regarding the arcs to be included in the set *A*, we will consider the following primitives: *rdfs:subClassOf*, *owl:intersectionOf* ,

owl:disjointWith, *owl:unionOf*, and the "instance-of" relations between instances and classes.

5.3.1 Circularity Problems

To detect circularity problems, the set *A* of the graph *G(V,A)* will contain specialization/generalization primitives. The direction of each arc is the same as the direction of the relation, except for *owl:unionOf*, where it is the opposite. ODEval looks for cycles in the graph *G*.

5.3.2 Partition Errors

5.3.2.1 Common classes and common instances in disjoint decompositions and partitions

In these cases, ODEval uses the evaluation algorithm explained in section 5.2.2.1. The set *V* of the graph *G* will contain the classes (named and anonymous) and instances defined in the ontology. The set *A* will not only contain the specialization/generalization, but also arcs that connect each instance with the class to which it belongs.

5.3.3 Redundancy Problems

We can find redundancy in the same way described with RDF(S) (section 5.1.2), using the predicate *reachablesFrom(G,v,R)* and detecting whether removing an arc from the set *A* implies any change in the set of elements reachable from a vertex.

6. CONCLUSIONS AND FURTHER WORK

In this paper we have shown that, in general, current ontology tools are unable to detect possible anomalies, from a knowledge representation point of view, in concept taxonomies in RDF(S), DAML+OIL, and OWL.

Taking into account that: (a) only a few ontology tools are able to detect loops in concept taxonomies in RDF(S), DAML+OIL, and OWL ontologies, (b) only OilEd (connecting to FaCT) is able to identify a few of partition errors in concept taxonomies, and (c) none of the ontology tools detect redundancy problems; we considered that it was necessary to create more advanced evaluators in order to be complement to the ontology tools studied.

Consequently, we have developed ODEval, which is a tool that evaluates RDF(S), DAML+OIL, and OWL concept taxonomies from a knowledge

representation point of view. This tool is meant to help ontology developers in designing ontologies, without anomalies, in such ontology languages.

We will go on working in ontology evaluation from the knowledge representation point of view. We will extend ODEval so as to capture more problems in concept taxonomies (such as checking that the "subclass-of" relationships are defined between classes), relation taxonomies, etc.

ACKNOWLEDGEMENTS

This work has been supported by the Esperonto project (IST-2001-34373), by several research grants from UPM ("Becas asociadas a proyectos modalidad A y B") and by a research grant from MEC (AP-2002-3828).

REFERENCES

Bechhofer S (2002) *The DIG Description Logic Interface: DIG/1.0*. Technical Report. http://potato.cs.man.ac.uk/dig/interface1.0.pdf

Bechhofer S, Horrocks I, Goble C, Stevens R (2001) *OilEd: a reason-able ontology editor for the Semantic Web*. In: Baader F, Brewka G, Eiter T (eds) Joint German/Austrian conference on Artificial Intelligence (KI'01). Vienna, Austria. (LNAI 2174) Springer-Verlag, Berlin, Germany, pp 396–408.

Fernández-López M, Gómez-Pérez A, Pazos-Sierra A, Pazos-Sierra J (1999) *Building a Chemical Ontology Using METHONTOLOGY and the Ontology Design Environment*. IEEE Intelligent Systems & their applications 4(1) (1999) 37-46.

Gómez-Pérez A (2001) *Evaluating ontologies: Cases of Study*. IEEE Intelligent Systems and their Applications. Special Issue on Verification and Validation of ontologies. March 2001, Vol 16, Nº 3. Pag. 391 – 409.

Gómez-Pérez A, Suárez-Figueroa MC (2004) *Evaluation of RDF(S) and DAML+OIL Import/Export Services within Ontology Platforms*. 3rd Mexican International Conference on Artificial Intelligence (MICAI 2004) Mexico City, Mexico. PP: 109-118.

Gómez-Pérez A, Suárez-Figueroa MC (2003) *Results of Taxonomic Evaluation of RDF(S) and DAML+OIL Ontologies using RDF(S) and DAML+OIL Validation Tools and Ontology Platforms Import Services*. Evaluation of Ontology-based Tools (EON2003) 2nd International Workshop located at the 2nd International Semantic Web Conference (ISWC 2003) Sundial Resort, Sanibel Island, Florida, USA. PP: 13-26.

Gómez-Pérez A, Fernández-López M, Corcho, O (2003) *Ontological Engineering*. November 2003. Springer Verlag.

Guarino N, Welty C (2000) *A Formal Ontology of Properties* In R. Dieng and O. Corby (eds.), Knowledge Engineering and Knowledge Management: Methods, Models and Tools. 12th International Conference, EKAW2000, LNAI 1937. Springer Verlag: 97-112.

Horrocks I, Sattler U, Tobies S (1999) *Practical reasoning for expressive description logics*. In: Ganzinger H, McAllester D, Voronkov A (eds) 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99). Tbilisi, Georgia. (Lecture Notes in Artificial Intelligence LNAI 1705) Springer-Verlag, Berlin, Germany, pp 161–180.