How to Find Suitable Ontologies Using an Ontology-Based WWW Broker

Julio César Arpírez Vega[1], Asunción Gómez-Pérez[1],
Adolfo Lozano Tello[2] and Helena Sofia Andrade N. P. Pinto[3]†.
{jarpirez, asun, alozano}@delicias.dia.fi.upm.es, sofia@gia.ist.utl.pt

Abstract. Knowledge reuse by means of ontologies now faces three important problems: (1) there are no standardized identifying features that characterize ontologies from the user point of view; (2) there are no web sites using the same logical organization, presenting relevant information about ontologies; and (3) the search for appropriate ontologies is hard, time-consuming and usually fruitless. To solve the above problems, we present: (1) a living set of features that allow us to characterize ontologies from the user point of view and have the same logical organization; (2) a living domain ontology about ontologies (called *Reference Ontology*) that gathers, describes and has links to existing ontologies; and (3) (ONTO)^{2}Agent, the ontology-based www broker about ontologies that uses the Reference Ontology as a source of its knowledge and retrieves descriptions of ontologies that satisfy a given set of constraints. (ONTO)^{2}Agent is available at http://delicias.dia.fi.upm.es/REFERENCE_ONTOLOGY/

1 INTRODUCTION AND MOTIVATION

Nowadays, it is easy to get information from organizations that have ontologies using the WWW. There are even specific points that gather information about ontologies and have links to other web pages containing more explicit information about such ontologies (see The Ontology Page[4], also known as TOP) and there are also ontology servers like The Ontology Server[5] [8, 9], Cycorp's Upper CYC Ontology Server[6] [29] or Ontosaurus[7] [36] that collect a huge number of very well-known ontologies.

When developers search for candidate ontologies for their application, they face a complex multi-criteria choice problem. Apart from the dispersion of ontologies over several servers; (a) ontology content formalization differs depending on the server at which it is stored; (b) ontologies on the same server are usually described with different detail levels; and (c) there is no common format for presenting relevant information about the ontologies so that users can decide which ontology best suits their purpose.

Choosing an ontology that does not match the system needs properly or whose usage is expensive (people, hardware and software resources, time) may force future users to stop reusing the ontology already built and oblige them to formalize the same knowledge again. It would be very useful for the knowledge reuse market to prepare a kind of *yellow pages of ontologies* that provides classified and up-dated information about ontologies. These living yellow pages would help future users to locate candidate ontologies for a given application. A broker specialized in the ontology field can help in this search,

[1] Grupo de reutilización. Laboratorio de Inteligencia Artificial. Facultad de Informática. Universidad Politécnica de Madrid. España
[2] Área de Lenguajes y Sistemas Informáticos. Departamento de Informática. Universidad de Extremadura. España
[3] Grupo de Inteligência Artificial.. Departamento de Engenharia Informática. Instituto Superior Técnico. Lisboa. Portugal
† This work was partially supported by JNICT grant PRAXIS XXI/BD/11202/97 (Sub-Programa Ciência e Tecnologia do Segundo Quadro Comunitário de Apoio).
[4] http://www.medg.lcs.mit.edu/doyle/top
[5] http://www-ksl.standford.edu:5915
[6] http://www.cyc.com
[7] http://indra.isi.edu:8000/Loom

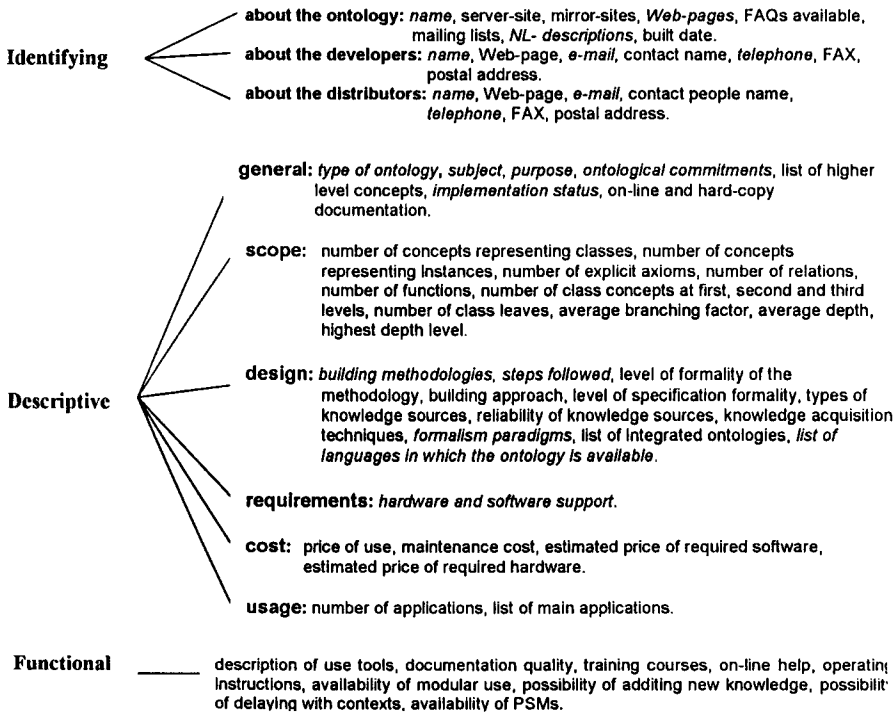

Figure 1. Feature taxonomy.

speeding up the search and selection process, by supplying the engineer with a set of ontologies that totally/partially meet the identified requirements. As a first step to solving the problem of searching for candidate ontologies, we present (ONTO)^{2}Agent, an ontology-based WWW broker on the field of ontologies that spreads information about existing ontologies, helps to search appropriate ontologies, and reduces the search time for the desired ontology. (ONTO)^{2}Agent uses as a source of its knowledge an ontology about ontologies (called *Reference Ontology*) that plays the role of a yellow pages of ontologies.

In this paper, we will firstly present an initial set of features that allow us to characterize, evaluate and assess ontologies from the user point of view. Secondly, we will show how we have built the Reference Ontology at the knowledge level [32] using the METHONTOLOGY framework [5, 11, 16] and the Ontology Design Environment (ODE) [5], and how we have incorparated the Reference Ontology into the (KA)^2 initiative [4]. Finally, we will present the technology we have used to build ontology-based WWW brokers and how it has been instantiated in (ONTO)^{2}Agent. (ONTO)^{2}Agent is capable of answering questions like: give me all the ontologies in the domain D that are implemented in languages L1 and L2.

2 FEATURES FOR COMPARING ONTOLOGIES

The goal of this section is to provide an initial set of features that allows us to characterize the ontologies from the user point of view by identifying the main attributes and their values. The kind of questions we are trying to answer are, for example: Which are the languages in which an ontology is available? Which are the mechanisms for interacting with the ontology? Is the knowledge represented in a frame-based formalism? What is the cost of the hardware and software infrastructure needed to use the ontology? What is the cost of the ontology? Is the ontology well documented? Was it evaluated [17] from a technical point of view?

Although Software Engineering and Knowledge Engineering provide detailed features for evaluating and assessing Software Engineering and Knowledge Engineering products [26, 33, 34], the literature reviewed in the field of ontologies shows that there are few papers about identifying features for describing, comparing and assessing ontologies. The taxonomy presented by Hovy [23] for comparing ontologies for natural language processing (divided into form, content and use) is insufficient for comparing ontologies in other domains. Fridman and Hafner [14] studied a small set of features for comparing well-known and representative ontologies.

To be able to answer the above questions, we have made a detailed study of the ontologies available at ontology servers on the web (Ontology Server, Cyc Server, Ontosaurus) and also other ontologies found in the literature (PhysSys [6], EngMath [18]). Our aim is twofold: first, to identify the more representative features of these ontologies (developers, ontology-server, type, purpose,...); second, to define a shared domain ontology about ontologies (the Reference Ontology) and relate each individual ontology to that shared ontology. This Reference Ontology could help future users to select the most adequate and suitable ontology for the application they have in mind.

To ease and speed up the process of searching for the features of the ontology, they are grouped in the following categories: identifying, descriptive and functional features, as shown in Figure 1. A preliminary set of features is proposed for each category. Since not all the features are equally important, the essential features, i.e., features which are indispensable in order to distinguish each ontology, are given in italics. It is compulsory to fill in these features. We also stress that: (1) some features cannot be used to characterize certain ontologies; (2) the ontology builder may not know the values of some features; and (3) this is a living list of features to be improved and completed with new features if as required.

2.1 Identifying features

They provide information about the ontology itself, its developers and distributors. We consider it important to specify:

- *About the ontology*: Its name, server-site, mirror-sites, Web pages, FAQs available, mailing lists, natural language description and built date.
- *About the main developers and distributors*: their names, Web pages, e-mails, contact names, telephone and fax numbers and postal addresses.

2.2 Descriptive features

They provide information about the content and form of the ontology. They have been divided into six categories: general, scope, design, requirements, cost and usage.

General features describe basic content issues. Users will frequently consult this kind of information, since these features are crucial for looking up other features. We considered the following properties: type of ontology [22], subject of the ontology,

purpose [37], ontological commitments [19], list of higher level concepts, implementation status, and on-line and hard-copy documentation.

Scope features describe measurable attributes proper to the ontology. They give an idea of the content and depth of the ontology. Properties to be taken into account are: number of concepts representing classes, number of concepts representing instances, number of explicit axioms, number of relations and functions, number of class concepts at first, second and third levels, number of class leaves, average branching factor, average depth, highest depth level.

Design features describe the method followed to build the ontology, the activities carried out during the whole process and how knowledge is organized and distributed in the ontology [8].

1. It is important to mention the methodology used, the steps [5, 11, 16] taken to build the ontology (mainly planning, specification, knowledge acquisition, conceptualization, implementation, evaluation, documentation and maintenance) according to the selected methodology, its level of formality [37], and the construction approach [37].
2. Depending on the methodology, the specification may be formal, informal or semi-formal.
3. With regard to knowledge acquisition, it is important to state the types of knowledge sources, how reliable such knowledge sources are and the techniques used in the process.
4. With respect to formalism paradigms, a frame-based formalism, a first order logic approach, a semantic network, like conceptual graphs, or even a hybrid knowledge representation paradigm can be selected. It is important to state here that the chosen formalism places constraints on the knowledge representation ontology in which the current ontology is going to be implemented. For example, if we select a frame-based formalism paradigm, one major candidate would be the frame-ontology at the Ontology Server. The formalism paradigm also plays a major role in ontology integration. For example, if you want to integrate an ontology built using a first order language into a frame-based paradigm a lot of knowledge will be lost due to the weaker expressive power of the latter.
5. As far as integration is concerned, a list of the integrated ontologies should be given.
6. Finally, we need to know from the implementation point of view, the source languages in which the ontology is supplied and the list of formal KR languages supported by available translators

Requirement features identify the minimal hardware (swap and hard disk space, RAM, processor, operating system) and software support requirements (knowledge representation languages and implementation language underneath the KR language) for using the ontology. All these features will greatly influence costs.

Cost features help to assess the estimated cost of using the ontology in a given organization. Since the hardware and software costs vary widely and depend on the existing computer infrastructure, the total cost should be calculated by adding the cost of use and maintenance to the features identified above (estimated prices of the hardware and software required).

The usage feature refers to the applications that use this ontology as a source of their knowledge. The number of known applications and their names are the features to be filled in by the informant.

[8] The ontology can be divided into several ontologies.

2.3 Functional features

These properties give clues on how the ontology can be used in applications. We have identified the following features: description of use tools (taxonomical browsers, editors, evaluators, translators, remote access modules, ...), quality of documentation, training courses available, on-line help available, how to use the ontology (including the steps followed to access, manipulate, display and update knowledge from remote and on-site applications), availability of modular use, possibility of addition of new knowledge, possibility of dealing with contexts, availability of integrating PSMs, etc.

3 DESIGN OF AN ONTOLOGY ABOUT ONTOLOGIES: THE REFERENCE ONTOLOGY

Having presented a living set of features that describe each ontology and differentiate one ontology from another, the goal of this section is to present how we have built the Reference Ontology using the features identified in section 2. As stated above, the Reference Ontology is a domain ontology about ontologies that plays the role of a kind of yellow pages of ontologies. Its aims are to gather, describe and have links to existing ontologies, using a common logical organization.

The development of this Reference Ontology was divided into two phases. The first phase is concerned with the development of its conceptual structure, and the identification of its main concepts, taxonomies, relations, functions and axioms. This phase was carried out using the METHONTOLOGY framework and the Ontology Design Environment. As one of the research topics of the KA community is ontologies, we decided to incorporate the Reference Ontology into the Product ontology of the $(KA)^2$ initiative that is currently being developed by the KA community. The second phase corresponds to the addition of knowledge about specific ontologies that act as instances in this Reference Ontology. Ontology developers will enter such knowledge using a WWW form also based on the features previously presented in section 2. So, the effort made to collect information about specific ontologies is distributed among ontology developers. It should be stressed that this is a first attempt at building a living ontology in the domain of ontologies. In this section we only present issues related to the first phase.

3.1 METHONTOLOGY

The METHONTOLOGY framework enables the construction of ontologies at the knowledge level. It includes: the identification of the ontology development process, a proposed life cycle and the methodology itself. The ontology development process identifies which tasks should be performed when building ontologies (planning, control, quality assurance, specification, knowledge acquisition, conceptualization, integration, formalization, implementation, evaluation, maintenance, documentation and configuration management). The life cycle (based on evolving prototypes) identifies the stages through which the ontology passes during its lifetime. Finally, the methodology itself, specifies the steps to be taken to perform each activity, the techniques used, the products to be outputted and how they are to be evaluated. The main phase in the ontology development process using the METHONTOLOGY approach is the conceptualization phase. Its aims are: to organize and structure the acquired knowledge in a complete and consistent knowledge model, using external representations (glossary of terms, concept classification trees, "ad hoc" binary relation diagrams, concept dictionary, table of "ad-hoc" binary relations, instance attribute table, class attribute table, logical axiom table, constant table, formula table, attribute classification trees and an instance table) that are independent of implementation languages and environments. As a

result of this activity, the domain vocabulary is identified and defined. For detailed information on building ontologies using this approach, see [16].

3.2 (KA)2 Ontological Reengineering Process

$(KA)^2$ is an initiative that models the Knowledge Acquisition Community (its researchers, research topics, products, events, publications, etc.) in an ontology that it is called the $(KA)^2$ Ontology. Initially, the $(KA)^2$ ontology was formalized in Flogic [28]. A WWW broker called Ontobroker [10] uses this Flogic ontology to infer new information that is not explicitly stored on the ontology.

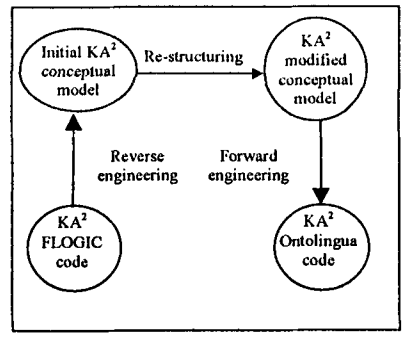

Figure 2. Ontological Reengineering Process of the $(KA)^2$ Ontology.

To make this ontology accessible to the entire community, it was decided to translate this Flogic ontology to Ontolingua [20] and to make it accessible through the Ontology Server. Since all the knowledge had been represented in a single ontology, the option of directly translating from Flogic to Ontolingua was ruled out (since it transgressed the modularity criterion), and it was decided to carry out an *ontological reengineering process* of the $(KA)^2$ ontology as shown in Figure 2. First, we obtained a $(KA)^2$ conceptual model, attached to the Flogic ontology manually by a reverse engineering process. Second, we restructured it using ODE conceptualization modules. After this, we got a new $(KA)^2$ conceptual model, composed of eight smaller ontologies: People, Publications, Events, Organizations, Research-Topics, Projects, Research-Products and Research-Groups. Finally, we converted the restructured $(KA)^2$ conceptual model into Ontolingua using forward ODE translators.

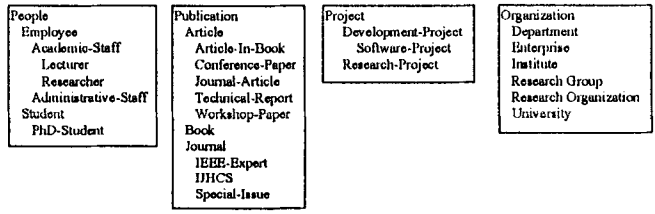

Figure 3. Concept Classification Tree in $(KA)^2$.

Figure 3 shows the main concepts identified in the domain grouped in Concept Classification Trees[9]. Figure 4 shows the most representative "ad hoc" binary relationships described in the Diagram of Binary Relations[10] of the new $(KA)^2$ ontology conceptual model; for instance, the relation Affiliation, between an Employee and an Organization; its inverse, the relation Employs; and the relation Cooperates-with, between two Researchers. It should be noted that multiple inheritance among concepts represented in the ontology is allowed, since for example a PhD Student is both a Student

[9] These trees identify the main taxonomies of a domain. Each tree will produce an independent ontology.
[10] The goal of this diagram is to establish relationships between concepts from the same or different ontologies.

and a Researcher. For a detailed explanation of the new $(KA)^2$ ontology conceptual model built after restructuring the Flogic $(KA)^2$ ontology, see [5]

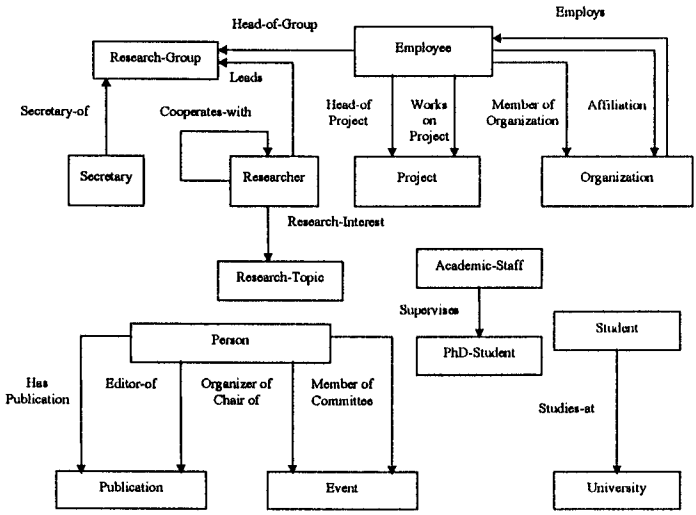

Figure 4. Diagram of Binary "Ad-hoc" Relations in $(KA)^2$.

3.3 Incorporating the Reference Ontology into (KA)2

As starting points for developing our Reference Ontology, we took three sources of knowledge. The first source was the set of features presented earlier in section 2. The second source was the restructured $(KA)^2$ conceptual model. The third source was the set of properties identified for the Research-Topic ontology, which were established during the KEML workshop held at Karlsruhe, on January 23, 1998 and distributed by R. Benjamins to the KA-coordinators-list. The properties identified were: Name, Description, Approaches, Research-groups, Researchers, Related-topics, Sub-topics, Events, Journals, Projects, Application-areas, Products, Bibliographies, Mailing-lists, Webpages, International-funding-agencies and National-funding-agencies. All these properties describe the field of ontologies and differentiate it from other fields of research. However, the properties we presented in section 2 characterize each ontology and differentiate one ontology from another. Some of the features presented in section 2 lead to some minor changes and extensions to the $(KA)^2$ ontology. For instance, information concerning distributors and developers was associated to Product and not exclusively to Ontology.

The design criteria used to incorporate the Reference Ontology into the $(KA)^2$ ontology were:

- Modularity: we sought to build a module-based ontology to allow more flexibility and varied uses.
- Specialize: we identified general concepts that were specialized into more specific concepts until domain instances were reached. Our goal was to classify concepts by similiar features and to guarantee inheritance of such features.

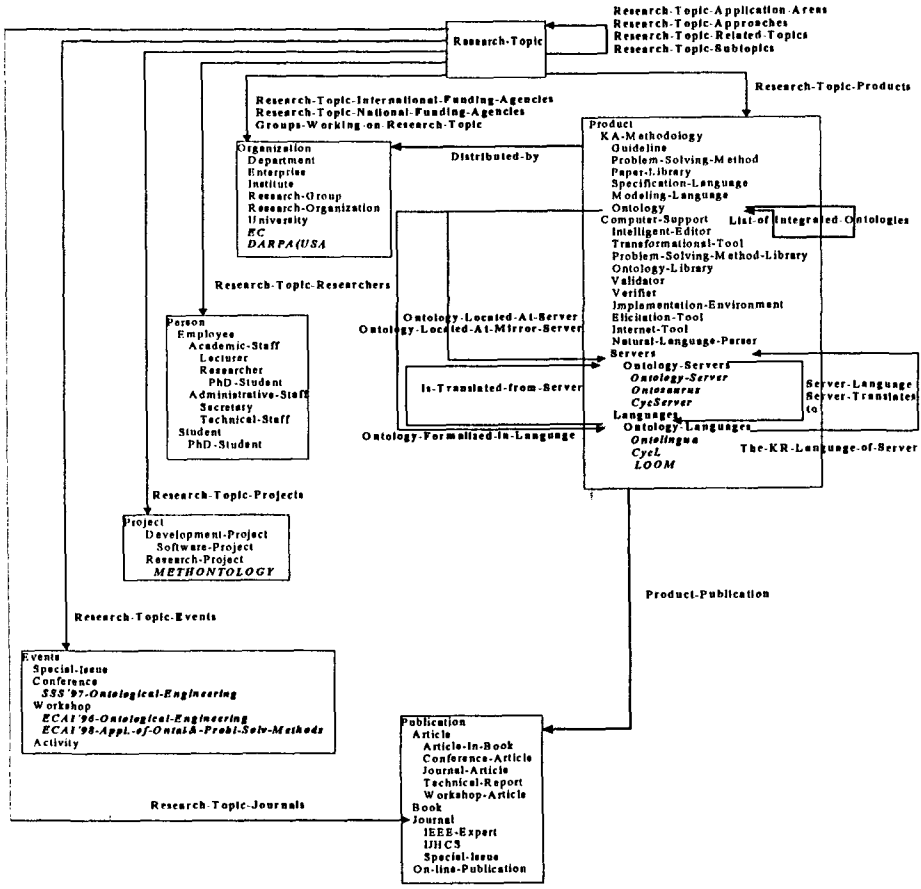

Figure 5. Some of the relations and concepts added to the $(KA)^2$ ontology.

- Diversify each hierarchy to increase the power provided by multiple inheritance mechanisms. By representing enough knowledge in the ontology and using as many different classification criteria as possible, it is easier to enter new concepts (since they can be easily specified from the pre-existing concepts and classification criteria).
- Minimize the semantic distance between sibling concepts: similar concepts are grouped and represented as subclasses of one class and should be defined using the same primitives, whereas concepts which are less similar are represented further apart in the hierarchy.
- Maximize relationships between taxonomies: in this sense, "ad hoc" relations and slots were filled in as concepts in the ontology.
- We have not taken into account ontology server, ontologies and language releases to build our ontology. For instance, in our ontology, the Ontology Server is an instance of servers and we do not keep records of its latest and future releases.
- Standardize names: whenever possible we specified that a relation should be named by concatenating the name of the ontology (or the concept representing the first

element of the relation), the name of the relation and the name of the target concept; for instance, the relation Ontology-Formalized-in-Language between the class of ontologies and one Language.

Based on the previous criteria, our analysis of the conceptual model of the $(KA)^2$ ontology showed that:

- about the classes: from the viewpoint of the Reference Ontology, some important classes were missing; for instance, the classes Servers and Languages, subclasses of Computer-Support at the Product ontology. The subclass of the class Servers is the class Ontology-Servers, whose intances are the Ontology-Server, the Ontosaurus and the CycServer. The subclass of the class Languages is the class Ontology-Languages, whose instances are Ontolingua, CycL [29] and LOOM [30].
- about the relations: from the viewpoint of the Reference Ontology, some important relations were missing; for instance, the relation Research-Topic-Products between a research topic and a product, or the relation Distributed-by between a product and an organization or the relation Ontology-Located-at-Server that relates an ontology to a server.
- about the properties: from the viewpoint of the Reference Ontology, some important properties were missing; for instance, Research-Topic-Webpages, Developers-Web-Pages, Type-of-Ontology or Product-Name.

So, we introduced the classes, relations and properties needed. The most representative appear highlighted in bold lettering in Figure 5.

All the changes, the entry of new relations and properties and the entry of new concepts were guided by the features that were presented in section 2. Essentially, the $(KA)^2$ ontology was extended using new concepts and some knowledge previously represented in the $(KA)^2$ ontology was specialized in order to represent the information that we found was of use and of interest for comparing different ontologies with a view to reuse or use as a basis for further applications.

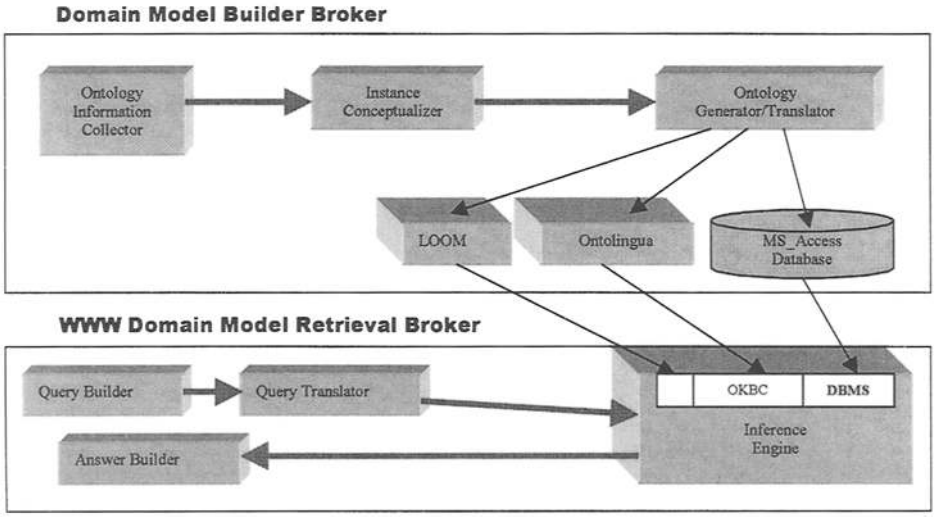

Figure 6. OntoAgent architecture.

4 ONTOAGENT ARCHITECTURE

Having identified the relevant features of ontologies and built the conceptual structure of the Reference Ontology using the Ontology Design Environment, the problem of entering, accessing and updating the information about each individual ontology arises. Ontology developers will enter such knowledge using a WWW form based on the features identified in section 2. A broker specialized in the ontology field, called (ONTO)^{2}Agent, can help in this search. In this section, we describe domain-independent technology for building and maintaining ontology-based WWW brokers. The broker uses ontologies as a source of its knowledge and interactive WWW user interfaces to collect information that is distributed among ontology developers.

The approach taken to build ontology-based WWW brokers is based on the architecture presented in Figure 6. It consists of different modules, each of which carries out a major function within the system. These modules are:

A. *A world-wide web domain model builder broker*, whose main capability is to instantiate the conceptual structure of an ontology about the broker domain expertise. This domain model builder needs:

A.1. *Ontology Information Collector:* an easy-to-use interactive WWW user interface that eases data input by distributed agents (both programs and humans);

A.2. *An Instance Conceptualizer:* for transforming the data from the WWW user interface into instances of the ontology specified at the knowledge level;

A.3. *Ontology Generator/Translators:* For generating or translating the ontology specified at the knowledge level into several target languages used to formalize ontologies and thus allow access from heterogeneous applications.

B. *A world wide web domain model retrieval broker*, whose aim is to provide help in accessing the information in an ontology warehouse and show it nicely. It is divided into:

B.1. A *query builder* to help to build queries using the broker vocabulary, as well as to reformulate and refine a query given by the user; the queries will be formulated upon a set of ontologies previously selected from the ontology pool available in the architecture;

B.2. A *query translator* that transforms the user query into a query representation compatible with the language in which the ontology is implemented;

B.3. An *inference engine* that searches for the answer to the query; as shown in Figure 6, knowledge sources can be represented in several formats;

B.4. An *answer builder* that presents to the client the answers to the query obtained by the inference engine module in an easy and human readable manner. The answers are presented for each ontology that has been searched. Thus, one query may be answered in several domains, depending on domains of the ontologies.

This technology has already been instantiated in two applications: (ONTO)^{2}Agent and Chemical OntoAgent.

4.1 (ONTO)2Agent

In the ontological engineering context, using the *Reference Ontology* as a source of its knowledge, the broker locates and retrieves descriptions of ontologies that satisfy a given set of constraints. For example, when a knowledge engineer is looking for ontologies written in a given language applicable to a particular domain, (ONTO)^{2}Agent can help in the search, supplying the engineer with a set of ontologies that totally/partially comply with the requirements identified.

The above abstract architecture has been instantiated as follows:

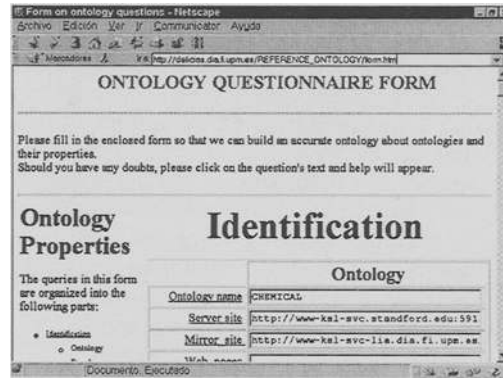

Figure 7. HTML ontology questionnaire form.

A. The *WWW-based domain model builder broker* uses:

A.1. A world-wide web form based on the identified ontology features previously discussed in this paper. Its main aim is to gather information about ontologies and thus distribute the effort made in collecting this data from ontology developers. Part of this form (http://delicias.dia.fi.upm.es/REFERENCE_ONTOLOGY) is shown in Figure 7. Note that the different categories are divided into groups. There are compulsory options that ontology developers must fill in -e.g., the ontology name, the language of the ontology-, while others are optional and offer a more detailed view of the ontology -e.g., number of nodes at the first level. The form contains proper questions to get the values of the features of an ontology. Besides, it contains help to guide the ontology developers filling in the form. A set of possible values are also identified for some questions, so the user merely has to click on a radio button or check box.

A.2. The data are used to fill in the instances of the concepts identified in the ontology described in section 3, which was built using ODE, thus ensuring full compatibility with this tool. Furthermore, we prefer to store the ontologies in a relational database rather than as implementations of other knowledge representation languages.

A.3. This database representation of the ontology specification is generated automatically using ODE forward translation modules. Knowledge can also be represented using other formats. Indeed, a number of translation languages we support, includes Ontolingua and SFK [13]. In the future, other languages such as Flogic or LOOM will be supported.

B. With regard to the *WWW-based domain model retrieval broker*:

B.1. Two query builders have been implemented, both similar in their conception but not implemented in the same manner. The first is a Java applet and the second, a Java standalone application. The main goal of the former is to get a fast applet download time to a web browser, limited by the Internet current transfer speed. Its functionality is smaller than the standalone application. This however, is due to the strict security restrictions applied to Java applets [25] and the above-mentioned speed limitation. Both elements seek to provide easy and quick access to ontologies. They possess a graphical user interface from which the user can build queries to any ODE ontology stored in the relational database. The query system is in fact domain-independent, although it has actually only been tested with two ontologies: Reference and CHEMICALS.

Both query builders allow users to formulate simple and complex queries. Simple queries can be made using the predefined queries present in the agent. They are based on ODE intermediate representations and include: definition of a concept, instances of a concept, comparison between two concepts, etc. They are used to get answers, loaded with information, easily and quickly. The query procedure is similar to the one used by Yahoo[11] or Alta Vista[12], so anyone used to working with these Internet search tools is unlikely to have any problems using the interface. Complex queries can be formulated by using a query builder wizard that works with AND/OR trees and the vocabulary obtained from the ontologies we are querying. It will allow us to build a more restrictive and detailed query than is shown in Figure 8, where we are looking for all the ontologies in the engineering domain, with standard units as a defined term and whose language is either Ontolingua, LOOM or SFK; before the query is translated to the proper query language, it is checked semantically for inconsistencies - syntactic correctness is implicit-, thanks to the query building method. If it is all right, it is refined, eliminating any redundancies.

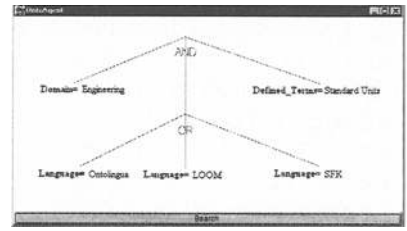

Figure 8. (ONTO)'Agent is asked to provide all the ontologies in the engineering domain, written in Ontolingua, LOOM or SFK, with Standard Units as a defined term, using a query expressed by means of an AND/OR tree.

B.2. The resulting query is then translated into the SQL language in order to match the ontology specification at the knowledge level, using the implementation of the ontology stored in a database. For the ontolingua implementation of a similar agent, an OKBC-capable [39] builder would be required.

B.3. The SQL query is sent to the server by means of a OntoAgent-specific protocol built on top of the TCP/IP stack. Therefore, the applications will be able to contact the server by means of this protocol. The inference engine used is the search engine equipped with MS-Access and some add-ins.

B.3. Once the query is sent to the server, the results will be returned and will be graphically visualized by the system. This representation will be different depending on whether or not natural language generation was requested. These results can be saved in HTML format for later consult using a common web browser.

Appart from this querying capability, we can also download or upload ontologies from the server or to the server. So, we can work on the ontology of our own workstation so as to work with it employing ODE, and modify and/or enlarge it as desired.

4.2 Chemical OntoAgent

Chemical OntoAgent is the other broker to which this technology has been applied. It is a chemistry teaching broker that allows students to learn chemistry in a very straightforward manner, providing the necessary domain knowledge and helping students to test their skills. To make the answers more understandable to students, this technology is able to interact with a system called OntoGeneration [1]. OntoGeneration is a system that uses a domain ontology (CHEMICALS [12]) and a linguistic ontology (GUM [2]) to

[11] http://www.yahoo.com
[12] http://www.altavista.com

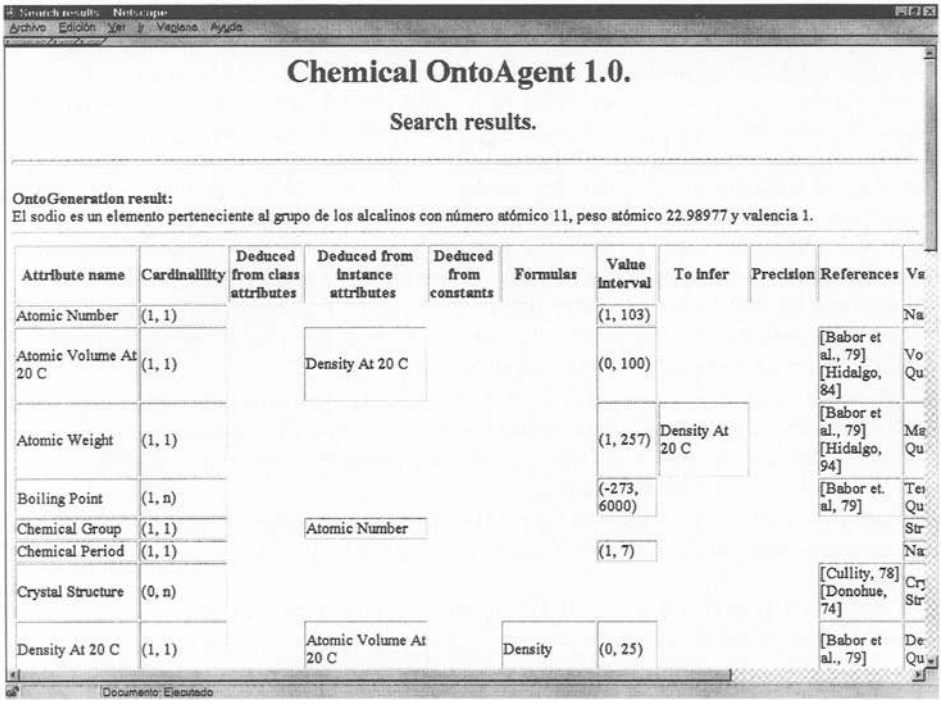

Figure 9. Search results in natural language and in tabular form. Sodium definition: Sodium is an element that belongs to the alkalymetal group and has an atomic number of 11,an atomic weight of 22.98977 and a valency of 1. The table also shows the Chemicals instance attributes table.

generate Spanish text descriptions in response to the queries in the domain of chemistry. This is shown in Figure 9, where we queried the definition of sodium and the instance attributes table of the Chemicals ontology using a predefined query.

Chemical OntoAgent does not have the modules described for the world-wide web domain model builder broker, since the Chemicals ontology was built entirely using ODE, and needed no further dynamic updating after its completion.

5 CONCLUSIONS

In this paper we presented (ONTO)^{2}Agent, an ontology-based WWW broker to select ontologies for a given application. This application seeks to solve some important problems:

1. To solve the problem of the absence of standardized features for describing ontologies, we have presented a living and domain-independent taxonomy of 70 features to compare ontologies using the same logical organization. This framework differs from Hovy's approach, which was built exclusively for comparing natural language processing ontologies. This framework also extends the limited number of features proposed by Fridman and Hafner for comparing well-known and representative ontologies, like: CYC [29], Wordnet [31], GUM [3], Sowa's Ontology [35], Dahlgren's Ontology [7], UMLS [24], TOVE [21], GENSIM [27], Plinius [38] and KIF [15].

2. To solve the problem of the dispersion of ontologies over several servers, and the absence of common formats for representing relevant information about ontologies using the same logical organization, we built a living *Reference Ontology* (a domain ontology

about ontologies) that gathers, describes using the same logical organization and has links to existing ontologies. We built this ontology at the knowledge level using the METHONTOLOGY framework and the Ontology Design Environment. We also presented the design choices we made to incorporate the *Reference Ontology* into the $(KA)^2$ initiative ontology after carrying out an Ontological Reengineering Process.

3. To solve the problem of searching for and locating candidate ontologies over several servers, we built $(ONTO)^2$Agent, an ontology-based WWW broker that retrieves the ontologies that satisfy a given set of constraints using the knowledge formalized in the Reference Ontology. $(ONTO)^2$Agent is an instantiation of the OntoAgent Architecture. OntoAgent and Ontobroker have several key points in common. Both are distributive, joint-efforts by the community, they use an ontology as the source of their knowledge, they use the web to collect information, and they have a query language for formulating queries. However, the main differences between them are:

- OntoAgent architecture uses: (1) a SQL database to formalize the ontology, (2) a WWW form and an ontology generator to store the captured knowledge, and (3) simple and complex queries based on ODE intermediate representations and AND/OR trees to retrieve information from the ontology.
- Ontobroker uses: (1) a Flogic ontology, (2) Ontocrowler for searching WWW annotated documents with ontological information, and (3) a Flogic-based syntax to formulate queries.

We hope that $(ONTO)^2$Agent and the Reference Ontology will ease the search of ontologies to be used in other applications.

6 ACKNOWLEDGEMENTS

We would like to thank Mariano Fernandez and Juanma Garcia for their help in using ODE.

7 REFERENCES

1. Aguado G., Bateman J., Bañón A., Bernardos S., Fernández M., Gómez-Pérez A., Nieto. E, Olalla A., Plaza R., Sanchez A. *ONTOGENERATION: Reusing domain and linguistic ontologies for* Spanish. Workshop on Applications of Ontologies and PSMs. Brighton. England. August 1998. 29
2. Bateman, J.A.; B. Magnini, G. Fabris. The Generalized Upper Model Knowledge Bare: Organization and Use. In Towards Very Large Knowledge Bases. Pages 60-72. IOS Press. 1995.
3. Bateman J. A., Magnini B., Rinaldi F., The Generalized Italian, German, English Upper Model, Proceedings of ECAI94's Workshop on Comparison of Implemented Ontologies, Amsterdam, 1994.
4. Benjamins R., Fensel D., *Community is Knowledge! in (KA)*^{2}, Knowledge Acquisition Workshop, KAW98. Presented in FOIS 1998.
5. Blázquez M., Fernández M., García-Pinar J. M., Gómez-Pérez A., *Building Ontologies at the Knowledge Level using the Ontology Design Environment*, Knowledge Acquisition Workshop, KAW98, Banff, 1998.
6. Borst P., Benjamins J., Wielinga B., Akkermans H., An Application of Ontology Construction, Workshop on Ontological Engineering, ECAI'96, Budapest, PP. 5-16, 1996.
7. Dahlgren K., Naive Semantics for Natural Language Understanding, Boston:MA, Kluwer Academic, 1988.
8. Farquhar A., Fikes R., Rice J., *The Ontolingua Server: A Tool for Collaborative Ontology Construction*, Proceedings of the 10th Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Alberta, Canada, PP. 44.1-44.19, 1996.
9. Farquhar A., Fikes R., Pratt W., Rice J., *Collaborative Ontology Construction for Information Integration*, Technical Report KSL-95-10, Knowledge Systems Laboratory, Stanford University, CA, 1995.
10. Fensel, D, Decker, S. Erdman M. Studer, R. *Ontobroker: The Very High Idea.* In Proceedings of the 11th International Flairs Conference (FLAIRS-98), Sanibal Island, Florida, May 1998.
11. Fernández, M., Gómez-Pérez, A. Juristo, N. *METHONTOLOGY: From Ontological Art Toward Ontological Engineering.* Spring Symposium Series on Ontological Engineering. AAAI97. Stanford. USA. March 1997.
12. Fernández M. *CHEMICALS: ontología de elementos químicos.* Proyecto fin de carrera. Facultad de Informática. Universidad Politécnica de Madrid. December 1996.
13. Fisher, D.; K. Rosle SFK: A Smalltalk Frame Kit. Technical Report, GMD/IPSI, Darmstadt, Germany, 1993
14. Fridman N., Hafner C., *The State of the Art in Ontology Design*, AI MAGAZINE, Fall 1997, PP. 53-74, 1997.
15. Genesereth M., Fikes R., Knowledge Interchange Format, Technical Report, Computer Science Department, Stanford University, Logic-92-1, 1992
16. Gómez-Pérez A., *Knowledge Sharing and Reuse*, The Handbook of Applied Expert Systems, Edited by J. Liebowitz, CRC Press, 1998.
17. Gómez-Pérez A., *Towards a Framework to Verify Knowledge Sharing Technology*, Expert Systems with Applications, Vol 11, N° 4, PP. 519-529, 1996.

18. Gruber, T. and Olsen, R. An Ontology for Engineering Mathematics, Technical Report KSL-94-18, Knowledge Systems Laboratory, Stanford University, CA, 1994.
19. Gruber T., *Toward Principles for the Design of Ontologies Used for Knowledge Sharing*, Technical Report KSL-93-04, Knowledge Systems Laboratory, Stanford University, CA, 1993.
20. Gruber T., ONTOLINGUA: A Mechanism to Support Portable Ontologies, KSL-91-66, Knowledge Systems Laboratory, Stanford University, 1992.
21. Gruninger M., Fox M., Methodology for the Design and Evaluation of Ontologies, Proceedings of IJCAI95's Workshop on Basic Ontological Issues in Knowledge Sharing, 1995.
22. van Heist G., Schreiber A. Th., Wielinga B. J., *Using explicit ontologies in KBS development*, International Journal of Human-Computer Studies, 45, PP. 183-292, 1997.
23. Hovy E., *What Would it Mean to Measure an Ontology?*, unpublished, 1997.
24. Humphreys B. L., Lindberg D. A. B., UMLS project: making the conceptual connection between users and the information they need, Bulletin of the Medical Library Association, 81(2), 1993.
25. JavaSoft. *Java Security FAQ*. http://java.sun.com/sfaq. October 1997.
26. Kan S. K., Metrics and Models in Software Quality Engineering. Ed. Addison-Wesley Publishing Company, MA, USA 1995.
27. Karp P.D., A Qualitative Biochemistry and its Application to the Regulation of the Trypophan Operon, Artificial Intelligence abd Molecular Biology, L. Hunter (ed.), 289-325, AAAI Press/MIT Press, 1993.
28. Kifer M., Lausen G., Wu J., *Logical Foundations of Object-Oriented and Frame-Based Languages*, Journal of the ACM, 1995.
29. Lenat D.B., *CYC: Toward Programs with Common Sense*, Communications of the ACM, 33(8), PP. 30-49, 1990.
30. Loom Users Guide Version 1.4. ISI Corporation. 1991.
31. Miller G. A., WordNet: An On-line Lexical Database, International Journal of Lexicography 3, 4: 235- 312, 1990.
32. Newell A., *The Knowledge Level*, Artificial Intelligence (18), PP. 87-127, 1982.
33. Pressman R., *Software Engineering: A practitioner's approach*, McGraw-Hill, 1997.
34. Slagle J., Wick M., *A Method for Evaluating Candidate*, AI MAGAZINE, WINTER 88, PP. 44-53, 1988.
35. Sowa J. F., Knowledge Representation: Logical, Philosophical, and Computational Foundations, Boston:MA, PWS Publishing Company, Forthcoming, 1997.
36. Swartout B., Patil R., Knight K., Russ T., *Towards Distributed Use of Large-Scale Ontologies*, AAAI97 Spring Symposium Series on Ontological Engineering, 1997.
37. Uschold M., Grüninger M., *ONTOLOGIES: Principles, Methods and 16Applications*, Knowledge Engineering Review, Vol. 11, N. 2, June 1996.
38. Van der Vet P.E., Speel P.-H., Mars N. J. I., The Plinius ontology of ceramic materials. Proceedings of ECAI94's Workshop on Comparison of Implemented Ontologies, Amsterdam, 1994.
39. Chaudhri Vinay K., Farquhar Adam, Fikes Richard, Karp Peter D., Rice James P. *The Generic Frame Protocol 2.0* , July 21, 1997.