The State of the Art of Ontology-based Query Systems: A Comparison of Existing Approaches

Hanh Huu Hoang A Min Tjoa
Institute of Software Technology and Interactive Systems
Vienna University of Technology
Favoritenstrasse 9-11/188
A-1040 Vienna, Austria
{hanh, tjoa}@ifs.tuwien.ac.at

***Abstract*-Based on an in-depth analysis of existing approaches in building ontology-based query systems we discuss and compare the methods, approaches to be used in current query systems using Ontology or the Semantic Web techniques. This paper identifies various relevant research directions in ontology-based querying research. Based on the results of our investigation we summarise the state of the art ontology-based query/search and name areas of further research activities.**

I. INTRODUCTION

This paper is aiming at a survey of current approaches in ontology-based search/query systems and researches. This survey based on an investigation of approximately 30 important publications on *ontology-based search/query systems*. The material used for this paper originates from various important publications databases and citation indexes. Furthermore, the approaches are analysed in this paper are very recently high-profiled. In this paper, we are not covering the topic of 'semantic search engines' but we focus on query systems or query modules in current systems or frameworks which are based on ontology and the Semantic Web technology.

From the data gathered from this survey, relevant research directions in ontology-based querying were identified based on similarities of research goals. While the classifications sometimes do not differ much in methodology, they seem sufficiently separate and logical with regards to research goals. Besides research directions, this paper also analyses the literature for the common methodologies and gives a short discussion about mentioned issues.

The main issues of our investigation to be presented in this paper are described according to the following criteria:

- *Search strategies with support of ontology*

We analyse three basic approaches of ontology-based search/query systems on making queries: the keyword search with augmenting of ontology, multi-facet (so-called view-based) search, and native ontology-based search/query.

This work has been generously supported by ASEA-UNINET and the Austrian National Bank within the framework of the project Application of Semantic-Web-Concepts for Business Intelligence Information Systems - Project No. 11284.

- *Query Formulation*

We survey the approaches of dealing with the query formulation problem, including the issues of complex constraint queries, proposing a new query language, and query user interface as well.

- *Query refinement*

We investigate how the ontology-based query systems deal with the problems of ambiguity solving, ranking during query refinement process. We also distinguish approaches in the current query refinement techniques.

- *User's interaction*

The issue is discussed along two aspects: query user interface in query formulation, and user's interaction during query refinement and answering processes. These issues are going to be analysed inside discussions on above criteria.

The remainder of this paper is organised as follows: an analysis of search strategies used in ontology-based query/search systems is presented in Section II. Section III describes the issues of query formulation. Details on the issues of query refinement are analysed in sections IV. And in section V, we present a sketch on the common methodology of current approaches. The paper is concluded with a summary of the discussed issues.

II. SEARCH STRATEGIES WITH SUPPORT OF ONTOLOGY

A. Augmenting Keyword Search with Ontology

Many query expansion implementations used in keyword search make use of 'thesaurus ontology navigation' as a step in query expansion such as [1], [2], and [3]. Particularly, a well-known usage of this technique is the large WordNet ontology (http://wordnet.priceton.edu). This kind of systems function along the same basic scheme: first, the keywords are located in the ontology, then, various other concepts are located through graph traversal, after which the terms related to those concepts are used to either broaden or constrain the search.

An algorithm is described in [5] for finding additional relevant information to a query given a starting set obtained via text search. Firstly, traditional text search is applied into a document collection.

Then, a process of RDF graph traversal is started from the annotations of those documents. The aim is to find related concepts such as the writer of the document, the project the document refers to, etc. The traversal is done by a spread activation algorithm, for the use of which the arcs in the ontology are weighed according to general interest. This is calculated by combining a specificity measure favouring unique connections in the knowledge base, and a cluster measure, which favours links between similar concepts.

Meanwhile, the CIRI system [7] provides an ontological front-end text search. The search is taken out through an ontology browser that visualises the ontologies as subsumption trees, from which concepts can be selected to constrain the search. The actual search is done through keywords annotated to these concepts and subconcepts, using a traditional search engine. The actual search algorithm is similar to the query expansion algorithms discussed before. The main difference is in the user interface being based on directly ontological browsing, leaving out the first step of mapping a search keyword to the ontology.

B. Ontology-based Multi-facet Search

A powerful search paradigm is that of multi-facet search [10] and additionally combined with ontology [12]. This is the search method of the Ontogator [12], and OntoViews [11]-based portals. In multi-facet search, multiple distinct views are provided into the data. These views are created via ontology projection, using also the various other hierarchical relationship trees and leaf relations usually inherent in ontologies besides class subsumption and membership.

In Ontogator [12], the underlying domain ontologies are mapped into facets and facilitate a multi-facet search. After finding information of interest by multi-facet search, Ontogator uses the domain ontology together with annotation data to recommend the user other related results. These relevant results are not available in the multi-facet search phase, which can be delivered by the recommendation system of Ontogator by semantic navigations. The idea behind this concept is that the user can start constraining the search from the view that is most natural to him/her.

In some versions of OntoViews, a concept called semantic auto-completion is used, which makes use of keyword search as a prelude to ontological navigation. The idea is that main interface of the portal opens with a keyword field. The keywords, however, are not linked directly to information items, but to ontological classes in the different views, from which semantic disambiguation can be made. The search then proceeds as a multi-facet search query. Once the search has proceeded to the point where at least a single interesting instance is found, additional information can be retrieved via browsing. The process is similar to the browsing of WWW, however, the shown items are resources and the links between them are defined by their relations.

C. Native Ontology-based Search

An approach with enhanced services of reasoning, catalogue is taken in the GeoShare project [22]. GeoShare uses ontologies for describing vocabularies and catalogues as well as search mechanisms for keywords to capture more meaning. During the search process, the user narrows his/her search space’s size by selecting specific domain; then he/she picks the appropriate concepts from these models and application ontologies, covering all available concepts, to define the concrete query. After that, he/she can parameterise the query to concretise the retrieval process. In the sequel, the system processes the query and transforms it into the ontology language for the terminological part, where the system looks for equivalent concepts and subconcepts.

Typically, the semantic web data is divided into two classes: ontological and instance data. The actual data of interest are entities belonging to a class, but the domain knowledge and relationships are described primarily as class relationships in the ontology. This very natural organization of data is embodied in SHOE system [8]. In SHOE, the user is provided first with a visualization of the ontology, and he can choose the class of instances he is looking for. Then, the possible relationships or properties associated with the class are discovered, and that allows the user to constrain the set of instances by applying keyword filters to the various instance properties. A similar system is Ontobroker [21], although there are some differences from SHOE in Ontologies usage. In SHOE, information providers can introduce arbitrary extensions to a given ontology. In contrast, Ontobroker relies on the notion of an *Ontogroup* [21] defining a group of users that agree on an ontology for a given subject.

With help of ontology, OntoLoger [14] builds a query mechanism by recording the user’s behaviours in an ontology and recall it. OntoLoger is a query system based on usage analysis in the ontology-based information portals. The structure of the ontology reflects the users’ needs. By using this, OntoLoger supports the user in fine-tuning of his initial query. Knowledge Sifter [20] is another approach using ontologies and agent technique. In this system, a user query is refined by consulting the Ontology Agent which provides a conceptual model for the domain.

In a further effort of above approaches, the authors of the Haystack [18] based their user interface paradigm almost completely on the browsing from resource to resource [19]. The principle of this approach is that users themselves usually do not actually know or remember the specific qualities of what they are looking for, but have some idea of other things related to the wanted items. The searching process is then a browsing experience in which the user looks for information resources that he/she already knows and which are somehow related to the target, and from there locates additional information on the target resource until it can be found.

III. Query Formulation

Many kinds of complex queries can be formulated as a problem of finding a group of objects of certain types which are connected by certain relationships. In the Semantic Web, this translates to graph patterns with constrained object node and property arc types. An example would be "Find all papers published in IEEE proceedings from 2000 to 2003 about 'ontology-based query', cited by recent publications in 2005," where "publications", "IEEE proceedings", years are ontological class restrictions on nodes and "published in", "cited by", and "time restriction" are the required connecting arcs in the pattern. While such patterns are easy to formalise to query in the context of the semantic web, they remain problematic because they are not easy to formulate for the users. Therefore, a number of approaches of the research into complex queries have been developed on the level of user interfaces for creating such query patterns as intuitively as possible.

In an effort of this approach, [20] presents GRQL, a graphical user interface for building graph pattern queries that is based on navigating ontology. Firstly, a class in the ontology is selected as a starting point. All properties defined as applicable to the class in the ontology are then given for expansion. Clicking on a property expands the graph pattern to contain that property, and moves selection to the range class defined for that property, e.g. clicking the "creates property" in an Artist class creates the pattern "`Artist→creates→Artifact`", and moves the focus to the `Artifact` class, showing the properties for that class for further path expansion. In addition to lengthening the path, other operations can be performed on the query pattern. The pattern can be tightened to concern only some subclasses of a class, as by tightening Artifact to "Painting of Sculptures" in the previous example to "`Artist→creates→ Painting or Sculpture`". In a similar way, property restriction definitions can be tightened into subproperties. More complex queries can be formulated by visiting a node created earlier and branching the expression there, creating patterns such as the one visually depicted in Fig. 1.

In a further effort of reducing the complexity of query formulation, the approach of "Semantic Search" interface, namely GetData [4], expresses the need of a much lighter weight interface for constructing complex queries. The reason is that the current query languages for RDF, DAML, and more generally for

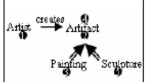
Navigation	RQL query	Path
	`select X2 from {X1}creates{X2;Painting }, Sculpture{X2}`	`Artist→creates→Painting Sculpture`

Fig. 1. A phase of GRDL

semi-structured data provide very expressive mechanisms that are aimed at making it easy to express complex queries; however, the queries require a lot of computational resources to process. The idea of GetData is to design a simple query interface which enables to network accessible data presented as directed labelled graph. This approach provides a system which is very easy to build, support both type of users, data providers and data consumers.

The multi-facet search portals mentioned earlier can also be regarded of as user interfaces for creating a very constrained subset of complex graph patterns. While in the simple case the query is formulated as searching for an information with particular properties, in a wider sense the definitions of how the objects map to the views can be arbitrarily complex and involve graph navigation, as for example where items are not directly annotated to particular event types, but the link is drawn from a combination of item type and material, for example.

IV. Query Refinement

A. Query Ambiguity Discovering

In the early approach, word sense disambiguation of the terms in the input query and words in the document has shown to be useful for improving both precision and recall of an information retrieval system. In the approaches of [1], [2] and [3], lexical relations from WordNet are used for query expansion, but without treating the query ambiguity.

Meanwhile, the approach in [15] examines the query ambiguity in two factors: the structure of the query and the content of the knowledge repository. Regarding ambiguities in the query structure, there are two issues are defined: structural ambiguity in which the structure of a user's query is analysed regarding the underlying ontology; and semantic ambiguity. The second factor is the content of the knowledge repository. The ambiguity of a query posted in a knowledge repository is repository-dependent. To overcome, [15] introduces a 'response factor' for taking the specificities of knowledge repository content in determining the ambiguity of a query. This factor of a query is the measure to know how the terms from that query cluster the resources in the underlying knowledge repository.

In recent activities, another approach for dealing with query ambiguity is presented in [17]. In this process, firstly, potential ambiguities of the initial query are discovered and assessed (Ambiguity-

Discovery). Next, these ambiguities are interpreted regarding the user's information need, in order to estimate the effects of an ambiguity on the fulfilment of the user's goals (Ambiguity-Interpretation).

B. *Query Refinement*

The approach of [15] for query refinement tries to simulate reflect the refinement model which a human librarian uses in her daily work. It means that we use three sources of information in query refinement: (1) the structure of the underlying ontology, (2) the content of the knowledge repository and (3) the users' behaviour (how users refine their queries on their own). Since the first two sources are used for measuring the ambiguity of a query, the query refinements based on them are treated cooperatively as the *ambiguity-driven* query refinement. In this query refinement approach, the ambiguity parameters presented in the previous section are combined and presented to the user in case she wants to make a refinement of the initial query. Each of ambiguity parameters has its role in quantifying ambiguity. For each of the parameters, query term(s) that affect the ambiguity most importantly are determined.

[16] presents another query refinement approach called *information-need driven query refinement*. This approach is a formalised one; it bases on 1) the definition of an order between queries, in order to create the map of the query neighbourhood – query map or so-called query space, and 2) the characterisation of the query ambiguity, in order to control the navigation in the query space – compass. The query refinement process is then realized as the movement through the query's neighbourhood in order to change the ambiguity of that query.

Similarly, [17] presents a comprehensive approach for the refinement of ontology-based queries, which is founded on the incrementally and interactively tailoring of a query to the current information needs of a user. These needs are implicitly and on-line elicited by analyzing the user's behaviour during the searching process. The gap between a user's need and his query is quantified by measuring several types of query ambiguities. Consequently, in the refinement process a user is provided with a ranked list of refinements, which should lead to a significant decrease of these ambiguities. Moreover, by exploiting the ontology background, the approach supports the detection of "similar" results that should help a user to satisfy his information need.

The third source for making the query's refinement recommendations in [15] mentioned above requires an analysis of the users' activities in an ontology-based application. That is also the approach of many query refinement mechanisms and OntoLoger [14] is a one of them. OntoLoger bases on the log-ontology (usage-data) and analyses the user's behaviour in order to guide the user in refinement process. By doing this, the refinement process will support a user in fine-tuning of his/her initial query. Thereafter, it ranks the received resources according to their relevance of the user's query, and finally, the system relaxes the user's query such that its best approximation can be found.

In a similar manner to OntoLoger, [12] deals with the query refinement based on the domain ontology and user annotation on data. The *Recommendation system* of Ontogator utilises the domain ontology together with annotated data and recommendation rules to recommend the user to view other related information which maybe missed by his initial query. This process is known s the *semantic browsing function*. Through this kind of system, the user can refine his/her queries by selecting related information that suits his needs.

Query refinement in Knowledge Sifter [20] is an aggregation of *query expansion* (Query Formulation Agent), which is also used in [1], [2] and [3], and *recommendation system* (Integration Agent) techniques. The Query Formulation Agent consults the Ontology Agent to refine or generalise the query based on the semantic median provided by the available ontology services. Besides, the Integration Agent is responsible for compiling the sub-query results from various sources, ranking them according user preferences.

V. Methodologies in Common

While surveying the field of ontology-based querying research, some common methodologies can be determined. Some are intrinsic to the RDF formalism and are present in almost semantic web applications. The knowledge and understanding of these common methods as well as how they are used in the various actual approaches are of great importance for future methodologies of ontology-based search/query systems.

A. *Role of Ontology*

In the regarded systems, ontologies are very crucial and play a key-role. Ontologies appear from the starting (query formulation) until the end (query answering) of querying processes. We can conclude the roles of ontology as following: (1) providing a pre-defined set of terms for exchanging information between users and systems; (2) providing knowledge for systems to infer information which is relevant to user's requests; (3) filtering and classifying information; and (4) indexing information gathered and classified for presentation.

B. *Keyword - Concept Mapping*

Mapping between keywords and formal concepts is a common pattern appearing in ontology-based search/query modules. There are a number of reasons for its prevalence. The first is that an assumption of

TABLE I
SUMMARY OF ONTOLOGY-BASED SEARCH/QUERY SYSTEMS BASED ON THE SURVEY CRITERIA

Systems	Search method		Approach		Query Refinement	Inference	Query formulation		Using annotated data
	Enhanced-keyword	Ontology-based	Front-end	Back-end			Query UI	Ontology-based	
Ontogator	✓			✓	✓				✓
OntoLoger		✓	✓		✓			✓	
Knowledge Sifter		✓	✓		✓				
OntoViews	✓	✓	✓	✓	✓			✓	✓
OntoDoc	✓	✓		✓				✓	
GeoShare		✓		✓	✓	✓	✓	✓	
Ontobroker		✓		✓			✓	✓	✓
SHOE		✓		✓			✓	✓	
SEAL		✓		✓				✓	
Haystack		✓		✓	✓		✓	✓	
SemanticSearch	✓		✓				✓[3]		✓

all knowledge required being formally encoded is blindly optimistic. Huge research efforts are specifically achieved on the issue of combining searching through textual material with searching through formally defined information (for example in [12] and OntoWebs-based systems).

A second obvious reason is that natural language is the form of expression that comes most naturally to humans. Mapping patterns in the graph to sentences, such as in [24] can give the user a clearer picture of the represented relationships, and conversely the user may be more comfortable in formulating his queries as natural language sentences. In this case, keywords provide an entry-point for a quick way of locating information. Keywords and other textual/numeric restrictions can be easily specified for the given search fields, complemented by graphical navigating the ontology in order to locate the concepts and graph patterns to be used as search constraints.

C. Graph Patterns

Whether described via RDF path languages or in logical languages, graph patterns are an important concept in semantic web search methods, used in their different functions. Firstly, because of the way the RDF data model is organised, graph patterns are often used to formulate and encode complex constraint queries as discussed in Section III.

In some systems, such as OntoViews, general RDF path patterns are also used to link interesting resources to each other, or, as in [27], to formulate patterns for locating interesting connecting paths between named resources. Also, in result visualization, the parameters on where to fetch information pertaining to the item are also usually given as simple graph patterns.

D. Query Refinement

All query refinement methods are ontology-based approaches aimed at disambiguation the posted user's queries. In the IR community, generally, we can see two directions of modifying queries or query results to the needs of users: query expansion and recommendation systems. *Query expansion* is aimed at supporting the users to make a better formulated query, i.e. it attempts to improve retrieval effectiveness by replacing or adding extra terms into the initial query. The *interactive query expansion* supports such an expansion task by suggesting candidate expansion terms to the users based on some indices or concept hierarchies. *Recommendation systems* try to recommend items similar to those a given user has liked in the past, or identify users whose tastes are similar to those of the given user. More and more *ontology-based query refinement* techniques are formalised, and more complex and more effective approaches have been introduced, that are [14] (usage-based), [15] (ambiguity-driven), [16] (information-need driven) and [17] (step-by-step).

E. Inference

Obviously inference on the semantic web must be regarded as a very complex problem. The fact that the Semantic Web is designed to work under the open world assumption, whereas most well explored logics operate only on the base of a closed world assumption, builds a fundamental difficulty. Also, the vision of the semantic web which comprises a large amount of data, constitutes a problem for most current inference algorithms. GeoShare [22] is one among the very few actual applications which currently use inference based on OWL. Meanwhile, many of others that do, such as [26], could have also been developed using simpler graph patterns.

F. Fuzzy Concepts, Fuzzy Relations, Fuzzy Logics

In the research direction of augmenting text search with ontology techniques, there is a need for formalisms which allow the combining fuzzy annotations based on text search with the firmness of semantic annotations. As a result, a number of formalizations and experimentations with fuzzy logics, fuzzy relations and fuzzy concepts have been undertaken in that field ([25] is herefore an excellent example.) Fuzzy logics are, however, not only useful in combining text search with ontologies. On the research side of prototypical search methods, [28] applies fuzzy qualifiers to complex constraint queries, while in [29], the idea is presented that user profiling could be used as a basis for weighting the relevance of an ontological relation to be used in the search.

VI. Discussion and Conclusion

A number of common patterns can be detected in the approaches described in this paper. On the technical level, it can be concluded that in the working context of an RDF model, quite many of the used common methodologies are of general nature.

Usually complex constraint queries are focused on models where individuals and classes are the interesting information items; we can observe relations which are present as equal partners in all the graph pattern, path and logic formalisms. After the deduction of a result set by using complex constraints, there are strong tendencies to use graph traversal algorithms to locate additional result items. While fuzzy logic formalisms and fuzzy concepts allow us to combine keyword search results as equal partners in complex constraint querying.

Besides, the ontology-based query refinement, which includes ranking issue and user-interaction, can be recognized as innovative approach for improvement of query precision and helping users clarify their queries from ambiguous initial ones. The query refinement has been started very early along with the query process in semantic web application, which uses simple expansion algorithms. The current approaches have proved their power with effective refinement strategies based on ontologies. The only approach which does not neatly wrap into the others is inference-based problem solving. Inference in general builds a much greater challenge for the most usual cases of ontology-based query systems.

A summary of the discussed ontology-based query systems according the common criteria is presented in the Table I.

References

[1] M. Rila, “The Use of WordNet in information retrieval,” ACL Workshop on the Usage of WordNet In Natural Language Processing Systems, pp. 31-37, 1998.

[2] D.I. Moldovan, R. Mihalcea, “Using WordNet and lexical operators to improve internet searches,” *J. IEEE Internet Computing*, pp. 34–43, April 2000.

[3] D. Buscaldi, P. Rosso, E.S. Arnal, “A WordNet-based query expansion method for geographical information retrieval,” In Working Notes for the CLEF Workshop, 2005.

[4] R. Guha, R. McCool, E. Miller, “Semantic search,” *In Proc. 12th Int. Conf. WWW ‘03*, ACM Press, pp. 700–709, 2003.

[5] R. Guha, R. McCool, “TAP: a semantic web platform,” *Int. J. Computer and Telecommunications Networking*, Vol. 42 (5), Aug 2003, pp. 557-577, NY, USA, 2003

[6] C. Rocha, D. Schwabe, M. P. de Aragao, “A hybrid approach for searching in the semantic web,” *In Proc. 13th Int. Conf. WWW ‘04*, pp. 374–383, 2004.

[7] E. Airio, K. Järvelin, P. Saatsi, J. Kekäläinen, S. Suomela, “CIRI - an ontology-based query interface for text retrieval,” *In Proc 11th Finnish Artificial Intelligence Conf.*, 2004.

[8] J. Heflin, J., Hendler, Searching the web with SHOE, In *Papers from the AAAI Workshop*, pp. 35-40, 2000.

[9] A. Maedche, S. Staab, N. Stojanovic, R. Studer, Y. Sure, “SEAL - a framework for developing semantic web portals,” *In Proc. 18th British National Conf. on DB*, pp. 1–22, 2001.

[10] E. Mäkelä, E. Hyvönen, T. Sidoroff, “View-based user interfaces for information retrieval on the semantic web,” *In Proc. Workshop End User Interaction*, 2005.

[11] E. Mäkelä, E. Hyvönen, S. Saarela, K. Viljanen, “OntoViews - a tool for creating semantic web portals,” *In Proc. 3rd Int. Conf. Semantic Web*, Springer Verlag, 2004.

[12] E. Hyvönen, S. Saarela, K. Viljanen, “Ontogator: Combining view- and ontology-based search with semantic browsing,” *In Proc. XMLFinland ’03*, Oct 2003

[13] D. Reynolds, P. Shabajee, S. Cayzer, “Semantic information portals,” *In Proc. 13th Int. World Wide Web Conf.*, 2004.

[14] N. Stojanovic, J. Gonzalez, L. Stojanovic, “Ontologer – A system for usage-driven management of ontology-based information portals,” *In Proc. L-CAP ’03 Conf.*, 2003.

[15] N. Stojanovic, “On the role of a Librarian Agent in ontology-based knowledge management systems,” *J. Universal Computer Science*, Vol. 9 (7), pp. 697-718, 2003.

[16] N. Stojanovic, “Information-need driven query refinement,” *In Proc. IEEE/WIC Int. Conf. Web Intelligence*, 2003.

[17] N. Stojanovic, R. Studer, L. Stojanovic, “An Approach for Step-By-Step Query Refinement in the Ontology-Based Information Retrieval,” *In Proc. Int. Conf. on Web Intelligence*, 2004.

[18] D. R. Karger, K. Bakshi, D. Huynh, D. Quan, V. Sinha, “Haystack: A general-purpose information management tool for end users based on semistructured data,” *In Proc. CIDR Conf.*, pp. 13–26, 2005

[19] D. Quan, D. Huynh, D. R. Karger, “Haystack: A platform for authoring end user semantic web applications,” *In Proc 2nd Int. Semantic Web Conf.*, pp. 738–753, 2003.

[20] L. Kerschberg, M. Chowdhury, A. Damiano, et al, “Knowledge Sifter: Ontology-Driven Search over Heterogeneous Databases,” *In Proc. 16th Int. Conf. Scientific and Statistical DB Management*, 2004.

[21] S. Decker, M. Erdmann, D. Fensel, R. Studer, “Ontobroker: Ontology-based Access to Distributed and Semi-Structured Information,” *J. Database Semantics: Semantic Issues in Multimedia Systems*, pp. 351-369, Kluwer Publishers, 1999.

[22] S. Hübner, R. Spittel, U. Visser, T. J. Vögele, “Ontology-Based Search for Interactive Digital Maps,” *J. IEEE Intelligent Systems*, Vol. 19 (3), pp. 80 – 86, May-Jun 2004.

[23] N. Athanasis, V. Christophides, D. Kotzinos, “Generating on the fly queries for the semantic web: The ICS-Forth Graphical RQL Interface (GRQL),” *In Proc. 3rd Int. Semantic Web Conf.*, pp. 486–501, 2004.

[24] T. Catarci, P. Dongilli, T. D. Mascio, E. Franconi, G. Santucci, S. Tessaris, “An ontology based visual tool for query formulation support,” *In Proc. 16th Euro. Conf. on AI*, pp. 308–312, 2004.

[25] L. Zhang, Y. Yu, J. Zhou, C. Lin, Y. Yang, “An enhanced model for searching in semantic portals,” *In Proc. 14th Int. Conf. WWW ‘05*, pp. 453–462, NY, USA, 2005.

[26] R. Fikes, P. Hayes, I. Horrocks, “OWL-QL: A language for deductive query answering on the semantic web,” Technical report, KSL, Stanford University, 2003.

[27] K. Anyanwu, A. P. Sheth, “ρ-queries: enabling querying for semantic associations on the semantic web,” *In Proc. 12th Int. Conf. WWW’03*, pp. 690–699, 2003.

[28] S. Singh, L. Dey, M. Abulaish, “A framework for extending fuzzy description logic to ontology based document processing,” *In Proc. 2nd Int. Atlantic Web Intelligence Conf.*, pp. 95–104, 2004.

[29] D. Parry, “A fuzzy ontology for medical document retrieval,” *In Proc. 2nd Workshop on Australasian Infor. Sec., DM and WI, and Soft. Inter.*, New Zealand, pp.121–126, 2004.