META LEVEL: ENABLER FOR SEMANTIC STEERED MULTIMEDIA RETRIEVAL IN AN INDUSTRIAL DESIGN DOMAIN

Gorka Marcos[1†], Tim Smithers[2], Iván Jiménez[1], Carlos Toro[1]
[gmarcos ijimenez ctoro]@vicomtech.es [tsmithers]@ fatronik.com
[1] VICOMTech, Donostia / San Sebastián
[2] FATRONIK, Donostia / San Sebastian
†Correspondent author

ABSTRACT

Globalization and the digitalization of data and automation of the processes involved in information management, have turn most of nowadays design companies deeply dependent on their information systems [10] [4] [8]. The description of information management systems in design environments usually include terms that belong to different fields of knowledge whose definitions are similar but as they are being provided by different areas of knowledge and multidisciplinary teams, heterogeneous information sources and collaborative working tools need to be developed. Very important efforts are being presented by the scientific community in order to improve the information management systems in design environments. However actual systems could be improved by the use of semantic technologies in order to enhance their accuracy and efficiency with the consequent reduction in time and money derived from such outcomes [2],[5],[14]. In this paper we present a possible architecture of a module that provides semantic services (The Meta Level) which has been developed within the framework of a European project called WIDE [6]. In our presented approach the main focus is to improve the information management in a design environment by means of semantics using an alternative (cyclic) information retrieval paradigm that is supported strongly on the most recent semantic technologies. The presented architecture could be easily adaptable to different systems, facilitating the implementation of such information handling in other fields in where retrieval of information could be needed.

KEYWORDS

Semantic data retrieval, OWL, Semantic steered technologies, Meta Levels

1. INTRODUCTION

Industrial Design processes usually require that huge amount of information is accessed by multidisciplinary groups. The management of such information, which is increased considerably everyday in most of the companies, has become not only an urgent need but also a success feature mark of well known enterprises. Companies that do not have a highly effective information governance process, around 73% and 79% respectively, say the quality of their internal and external unstructured data is fair or poor [4].

Plenty of methodologies, technologies and tools have been developed to support concurrent access to the existing information in the companies. Commendable and interesting efforts are being made in the field of content based information retrieval, developing and implementing algorithms and techniques that allow automatic o semiautomatic content indexing [2],[3],[8].

Companies usually aim the integration of these management systems in their workflows. However, the deployment process implies new difficulties due to the intrinsic peculiarities of each company, a fact that it is worsened with the appearance of more and more types of information sources.

Here is where Semantic Web Technology [1] comes into play. There is dense bibliography about the application of the Semantic Web technologies to enhance the information retrieval process[9]. An interesting reader is referred to [5] for different examples of the application of the Semantic Web Technology on the handling of information and retrieval process in some heterogeneous environments.

One of the systems that have tackled the mentioned issues was developed within the frame of the EC project WIDE (IST-2001-34417) [6]. In this project the aim was to explore the **heterogeneous information integration in a design environment based on Semantic Web Technologies** such as ontologies, software agents, etc. for improved off-line collaboration (information and knowledge retrieval and exchange) as well as for improved online collaboration in multi-disciplinary teams has been explored. There are several publications [6] describing the aim, the results and the system's architecture.

The main objective of this paper is to describe the implementation issues of the module that provides the services needed to bring semantics all over the information retrieval process in a design environment on the WIDE architecture. We call this specific module the **Meta Level**.
This paper is organized as follows: in **section 2** we present a brief description of the WIDE approach architecture. In **section 3** we describe the search approach proposed by the project consortium. Finally in **section 4** we describe the architecture of the module that enables the core set of semantic services: the "Meta Level", lastly we present some conclusions and possible future work.

2. WIDE ARCHITECTURE

The focus of this section is to describe the way the WIDE system has been designed to support a more effective knowledge-sharing and semantically enhanced multimedia information retrieval. The following sections outline the mechanisms that have been developed. Figure 1 depicts the main components of the WIDE Semantic based Information System, and their relations.

Fig 1 WIDE Architecture

The system is composed by four subsystems: the *User Interface* (UI), *the Meta Level* (ML), *the Agency* and *the Content Level*. Although in this article we will focus only in the ML, the reader should keep in mind the whole Architecture for a better understanding of the module needs and behaviour. The *UI (User Interface)* is a graphical front end (Fig 2) that is in charge of handling all the aspects dealing with the user interface.
The *CL (Content Level)* compiles the different information sources of the company. In the project RDFS information sources, relational databases, internet websites and proprietary product management systems have been tested. The *Agent Platform* acts as the glue between the users (UI) and the information sources (CL). In order to carry out that intermediation, it makes use of the functionalities provided by the ML.
The *ML* represents all the knowledge gathered by the domain and provides different semantic services to the other modules. ML internal architecture has been designed to facilitate this and will be explained in section 4.

Fig 2 WIDE User Interface Front End

3. WIDE WORKFLOW

This section describes a new interaction paradigm proposed within the WIDE project. This paradigm, deeply explained in [7] has been defined to cover the needs of information in the engineering design activity, where different people work collaboratively during different steps of a global process.

The WIDE approach is a new interaction paradigm based on the semantic enrichment of the retrieval process. In order to see this, it is necessary to understand the Classical Model for Search Engines that can be seen in Fig 3. A couple of highlights will that the flow information is characteristically linear and for that reason could have a compromised efficiency and its exactness could be improved by a cyclic approach in where semantics take a big role to specialize the information retrieval process.

Fig 3 Classical Search Model

Fig 4 represents the WIDE model of information retrieval which has been developed to better support effective and efficient information retrieval and re-use by users involved in a design process who do not have strong prior knowledge of the organization and structure of the information sources they need to access.

Fig 4 WIDE Search Model

The first step, "**Explore and Specify**", includes two sub-processes. The first one represents the system facilities that allow the user to explore and browse graphically his/her domain of interest in order to discover what might form possibly good search specifications. A change of the user's mind at this very first step implies a considerable optimization of the search process. The Specify sub-process allows the user either to type the target query or to drag and drop concepts from the explored graph. The **Search** and the **Filter** processes do not differ too much from the Classical model. However, the first one can use all the peculiar advantages of the Semantic Web based search engines and the second one has a wider range of criterions (information about the user, knowledge about the domain of the search, semantic comprehension of the query, etc.) that facilitates the filtering. Once the results have been collected, filtered and ranked by the system, they are **Presented** in the WIDE User Interface (UI) that facilitates its **Semantic Browsing**. This interface lets the user not only get the final results (fifth step) but also use the shown information to refine the query and go back to the first step of the process. Thus, the model defined is circular.
Besides this, the WIDE system provides the user with some online collaboration facilities in order to interact with other users.

4. META LEVEL (ML) ARCHITECTURE

As has been explained in the previous chapter, WIDE Workflow the WIDE system supports the user in all the steps of what has been named "Circular Search" approach and over the online collaboration mechanisms. The presented architecture proposed here as the main novelty of this publication and is described in Fig 5.

Fig 5 ML Architecture

In the architecture a level differentiation can be recognized. Each level is composed by several items, that we call entities. The naming schema proposed for these entities is "Support Processes" for the outer layer, "Processing Elements" for the intermediate one and "Knowledge Base" for the internal. The outer layers make use of the inner layers. Thus, the farther away from the centre the user is, the more sophisticated behaviour the entities present. This **ML Knowledge Base**, is the core of the ML subsystem and, contains the Meta Level ontologies and a BNF [15] grammar. This entity by itself is not able to carry out any task, since it has no processing facilities. The "**Processing Elements**" compose the second layer of the architecture. These entities interact with the knowledge stored in the ML Knowledge Base in order to perform some atomic tasks. For example, the "Result Ranker" can be considered as a black box that receives a set of concepts and produces a graph out of them, basing on the existing relationships among them in the Knowledge Base.
Finally, the "**Support Processes**" are in charge of the combination of the different "Processing Elements" to interact with the rest of the WIDE Subsystems and to provide the user with the workflow above mentioned. The "Processing Elements" are devoted to work for the other components of the WIDE System, and in fact to any component involved in the search and retrieval process, in order to spread the semantics represented in the knowledge based wherever they could be applied.

4.1. Meta Level Knowledge Base

This section describes what in the ML is understood as Knowledge Base. As has been stated in the introductory section, the ML Knowledge Base is composed of every piece of knowledge used by the Processing Elements to carry our their different atomic tasks, which allow the Support Processes interact with the different WIDE subsystems all over "the WIDE cycle". In the WIDE project, this ML Knowledge Base is mainly composed by the ML Ontologies and the BNF grammar. However, the composition of the ML Knowledge Base is highly dependent on the requirements of the system. For instance, the BNF grammar could be replaced with a Natural Language Engine, the number and nature of the ontologies could be reduced or enhanced, some machine learning techniques could be added and so on. Nevertheless the first sub-section of this chapter will describe the ML Ontologies all over their life cycle (edition, storage, inference, etc.) whereas the last section will describe the BNF grammar, stressing its role in the ML Architecture.

A. ML Ontologies

ML is composed of seven interrelated Ontologies that represent all the knowledge related with the domain: processes of the company, the different users of the companies and their preferred terminology, the different tasks accomplished by those users, the concepts of that domain and so on. The language used to edit ML Ontologies is OWL (Ontology Web Language). The Edition of ML Ontologies has been performed with the Ontology Editor Protégé. Once the Ontologies are edited we need to ensure an appropriate access to the information represented with those sets of classes and relationships. The lack of mature of the existing OWL ontologies reasoning technologies during the WIDE development phase, lead the ML team to implement an optimized SQL repository that represented the basis of the knowledge gathered by the Ontologies. Additionally mechanisms were developed to allow the storage of some pre-inferred information. Fig 6 show the architecture with these modules and the relationships between them.

Fig 6 ML OWL Repository Architecture

B. BNF Grammar

The implementation of a Natural Language Processor was out of the objectives of the WIDE project. The approach tackled by the consortium to handle "semi-natural" queries was based on a BNF grammar [15] that represented some of the most common types of queries: "Images with wild animals", "BMW cars", "Pictures about cars with 12V engine" and so on.

4.2. ML Processing Elements

As has been stated, ML Processing Elements (PEs from now on) can be considered as black boxes that perform atomic tasks. The PEs are independent or not aware of the search itself. Their main characteristic is that they are able to **perform some task by reading/writing, carrying out inference, invoking either the ML-Knowledge Base or other PEs or even some external sources as web thesaurus, but without paying attention to the search process itself**. In the following paragraphs, some PE descriptions are included, in order to facilitate the understanding of the whole approach.

A. Ontology Concepts Relation Engine

This PE is in charge of inference the OWL-Repository in order to find out if the relation about the concepts. The Ontology Concepts Relation Engine implements an optimized algorithm that, when receives 2 terms that have been modelled within the Knowledge Base, finds out if both terms are related, retrieving for each case useful and needed information.
The following picture shows some of the relations that this PE is able to infer in an affordably short period of time.

Fig 7 Relationships Model

B. User Profile Based Mapping

The User Profile Based Mapping is responsible for the terminology mapping made by the ML at very different moment of the process in order to handle the different user's terminology, which is one of the main objectives of the project.

C. BNF Syntactic Parser

This Processing Element checks if a string is compliant with the BNF notation and provide a set of "error codes" as an output.

D. Semantic Interpreter & Query Expansion.

This processing element carries out one of the most critical tasks in the ML. The input of this process is an AST tree (BNF parser output) that has been successfully syntactically parsed and which semantic coherence has been checked. This represents the user query and has also complete information about the terms that compose the tree, taking into account what is represented in the ontology.
The output of that process is a set of trees that represent the different System Queries. The expansion process is based on interpretation of the user query. This interpretation is based on the structure of the query typed (according to the BNF grammar) by the user and on the inference of the OWL-Rep. Besides this some rules are defined in order to apply some replacements whenever this PE considers is needed.
This allows to transform an AST that represents the query "Pictures about BMW cars" into different ASTs representing different system queries being one of them "IMAGES ABOUT CAR WITH BRAND = BMW".

E. Graph Constructor

This PE can be considered as a black box that generates the graphs that are visualized by the user. Having clear that the output is the graph, we list here the input needed by this PE:

- The list of concepts that must be shown in the graph (mandatory).
- A list of concept that are related with the context of the graph (optional).
- Level of expansion wished (optional).

F. Query Builder.

The RQL (RDF Query Language [12]) Query Builder is another of the Processing Elements defined within the ML. As has been stated, the analysis and process of the queries carried out by the ML is based on the output of the trees generated by the Syntactic Parser based on the ML BNF Notation.
However the language chosen in the WIDE system in order to exchange the queries between the ML and the Agency is RQL. Later on, this subsystem, the Agency, analyses this RQL queries in order to map and apply them into the different information sources available in the project.
According to this, the RQL query Builder is responsible for the translation of the system queries generated by the ML into the language RQL. The following image represents the RQL query for “document that has_info_about tests for audi cars with cylinder > 8”.

Fig 8 System Query in RQL

G. Result Ranker

This Processing Element is invoked just before sending the results to the User Interface, once per each node of the graph.
Its role is ranking the results in the graph. This ranking is based on several criteria. Some of the most important are the following: relevance of the metadata of the result, similarity metrics between the result metadata and the ML Knowledge base, context information (if available) and so on.

4.3. Support Processes

This section takes care of the last kind of entities that compose the ML Architecture: the Support Processes. Those processes, as has been previously stated, are in charge of the interaction with the rest of the subsystems of the retrieval process and their main role is to spread the semantics all over the search, explore, retrieval and consume processes.

In the WIDE project, seven Support Processes (SPs) have been developed in order to contribute in the same number of services carried out by the system:

a) Domain Contextualization SP : This SP supports the system to learn about the context (task carried out by the user, his/her profile, the process he/she could be trying to realize and so on), giving the user the information about the domain that could be of his/her interest. Thus, this SP is focused on supporting the user in the learning and querying process of the domain.

b) Query Development SP: The usage of this Processing Element makes possible the drag and drop facility that allows the user to drag a node, drop it in the query bar and the query is refined in order to include the dragged concept in a semantically coherent way. Each time the user drags a concept from the graph and drops it into the query bar, an internal inference process is started by this SP through the invocation of different PEs.

c) Semantic Query Processing SP: The Semantic Query Processing SP is in charge of the process that starts when the user query is received by the Meta Level and finishes when the different System Queries (SQs) generated by the ML are sent to the Agency. Thus, this SP handles the structural and syntactical analysis of the query, its semantic interpretation and its expansion into different queries.

d) Search SP: Defining the search as the process that starts when the System Queries generated by the ML are sent to the Agency and the finalizes when the results reach the Meta Level, the role of the ML is related to the assistance to the Agency in the terminological and structural mapping among the ML internal terminology and the terminology of the different Information Sources.

e) Result Evaluation SP: when the results are retrieved, the Result Evaluation Support Process makes use of some Processing Elements (Result Ranker, Result Evaluator, SQ Handler, Ontology Concepts Relation Engine and Process Support Handler) in order to analyse and rank the results.

f) Result Presentation SP: After the internal evaluation of the results, the system prepares the visualization of the information retrieved. This process is in charge of the following tasks: generation of the graph, assignation of the ranked results to the appropriate node of the graph, providing some high level features for semantic based results browsing.

g) Collaborative Interaction SP: The WIDE system includes a mechanism to share graphs between the users. Thus, the users are able to send the results they are visualizing to other users, in order to work collaboratively over the results. This SP provides some terminological support to cover this task.

5. EXAMPLE OF USE

In this chapter we present a simplified example of use for pour approach. The symbol (*) is used to remark where the functionalities implemented in the ML architecture described are used.

Paul, a designer, logs into the system because he needs to look for inspiration in order to design a new door for a new car that is being prototyped in his company. In the logging process he is shown (*) the different processes usually done in his company and specially the ones related with his profile. He chooses “Station Wagon door design” and then a graph with the main concepts involved in the task appears in the screen (*). As he is used to work in this task, he avoids to browse, query and expand this graph (*) and he types in the query bar “Images of Maserati doors”.

What happens internally is that the system analyses if the query fits with the BNF grammar (*). Once this has been checked, some terminological translations may be done (*) according to Paul’s profile and personal dictionary (*). After that, the system is ready to create several system queries out of his query (*). The result of this process is a set of system queries like “Images of doors of cars with brand = Maserati” or “Image of doors with brand = Maserati” which are transformed into RQL (*) and submitted to the agents, which will apply them to the Information Sources.

Once the results arrive, they are ranked (*), evaluated (*) and graphically displayed making use of all the semantic information available in each result.

6. CONCLUSIONS

In this article we presented architecture of a module belonging for the semantic retrieval of information called ML, which has the following characteristics.

It is defined to enhance the search and retrieval carried out by external modules. This is due to the fact that its layered architecture facilitates the independence between the domain (ML Knowledge Base), the machinery (PEs) and the integration with whatever modules participate in the search & retrieval operations.

It is optimized to implement a Circular Search approach (section 3). Thus, it provides the needed mechanism to allow a new interaction with the search engines, based on the retrieval process.

It is domain independent and easily migrated from one domain to other. Thanks to the isolation of the ML-Knowledge base, replacing the domain gathered there (ontologies, models, etc.) the module can be used in different domains.

It is easily extended, configured and exchanged. Just adding new SP, PE or including new functionalities in the ML-Knowledge base (i.e. a natural language engine) the behavior of the system can be extended and enhanced.

This module has been implemented, tested and evaluated with 1K concepts ontology and it suitable to be used on real time demanding processes as part of an European project called WIDE [6].

7. ACKNOWLEDGEMENTS

The work presented here has been developed within the framework of the European project WIDE IST-2001-34417, which was co-funded by the IST program (5th Frame Program) at the European Commission. We also want to express our gratitude to the wide consortium members: Fraunhofer IGD, CEFRIEL, Centro Computaçao Gráfico, Schenck and Italdesign, as without their expertise and support this work would have not been possible.

8. BIBLIOGRAPHY

[1] T. Berners-Lee, J. Hendler, and O. Lassila, “The Semantic Web,” Scientific American, pp 34–43, May 2001.

[2] S.C. Brandt et al, “Ontology-Based Information Management in Design Processes” 9th International Symposium on Process Systems Engineering and 16th European Symposium on Computer Aided Process Engineering, July 9 - 13, 2006, Garmisch-Partenkirchen/Germany

[3] Jian-Kang Wu, “Content-based indexing of multimedia databases”, Knowledge and Data Engineering, IEEE Transactions on, Volume: 9, Issue 6, pages 978-989. Nov.-Dec 1997.

[4] May 2006 Survey: The Big Payback from Effective Information Management http://www.cioinsight.com/article2/0,1540,1969965,00.asp

[5] Amit Sheth, “Semantic Web-Based Information Systems: State-of-the-Art Applications”. IGI Global (November 16, 2006). ISBN: 978-1599044262.

[6] WIDE Project HomePage < www.ist-wide.info >

[7] Marcos Gorka et al, “A Semantic Web based approach to multimedia retrieval”. 2005 Fourth International Workshop on Content-Based Multimedia Indexing (CBMI05). Riga, Latvia, 21-23 June 2005. Proceedings of the conference.

[8] John Restrepo and Henri Christiaans, “Problem Structuring and Information Access in Design”. Journal of Design Research 2004 - Vol. 4, No.2

[9] Urvi Shah et al, “Information retrieval on the semantic web”. Proceedings of the eleventh international conference on Information and knowledge management table of contents. McLean, Virginia, USA Pages: 461 - 468 Year of Publication: 2002 ISBN:1-58113-492-4

[10] Use Case of successful deployment of an Information System in a Design Company http://www.eeproductcenter.com/showPressRelease.jhtml?articleID=X511454

[11] S. Staab and R. Studer, (Eds.), “Handbook on Ontologies,” Springer Verlag, 2004

[12] Sean Bechhofer, Frank van Harmelen, James Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider and Lynn Andrea Stein eds. OWL Web Ontology Language Reference. http://www.w3.org/TR/owl-ref/, Feb 2004.

[13] Gregory Karvounarakis et al. “RQL: A Declarative Query Language for RDF”. The 11th Intl. World Wide Web Conference (WWW2002).

[14] Simeon J. Simoff and Mary Lou Maher, “Ontology-Based Multimedia Data Mining for Design Information Retrieval” Proceedings of Computing in Civil Engineering (1998). Pp. 212-223.

[15] Paul B Mann, A translational BNF grammar notation (TBNF) ACM SIGPLAN. Volume 41 , Issue 4 (April 2006) Pages: 16 - 23 Year of Publication: 2006

FIGURES

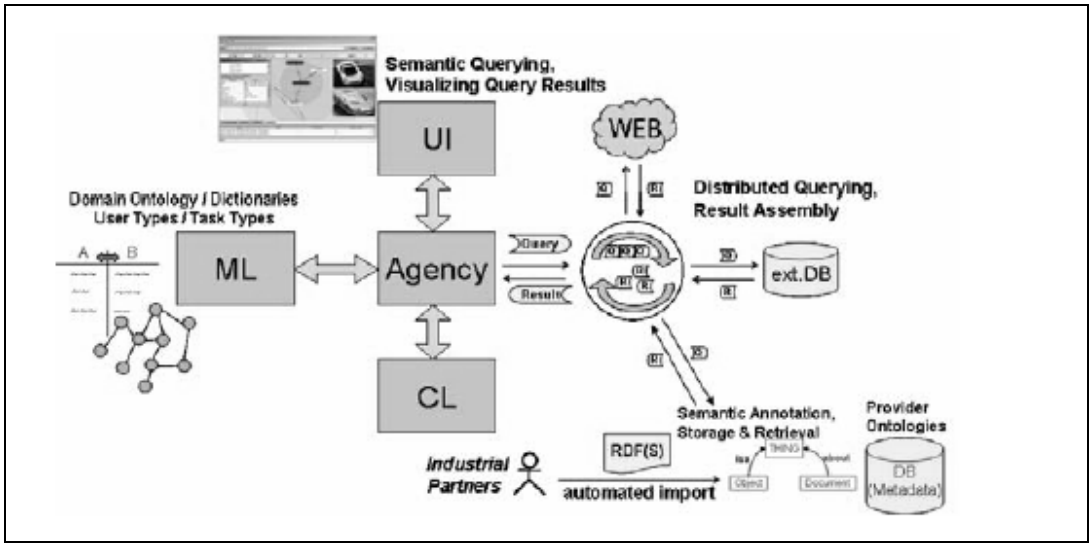

Fig 1. Architecture

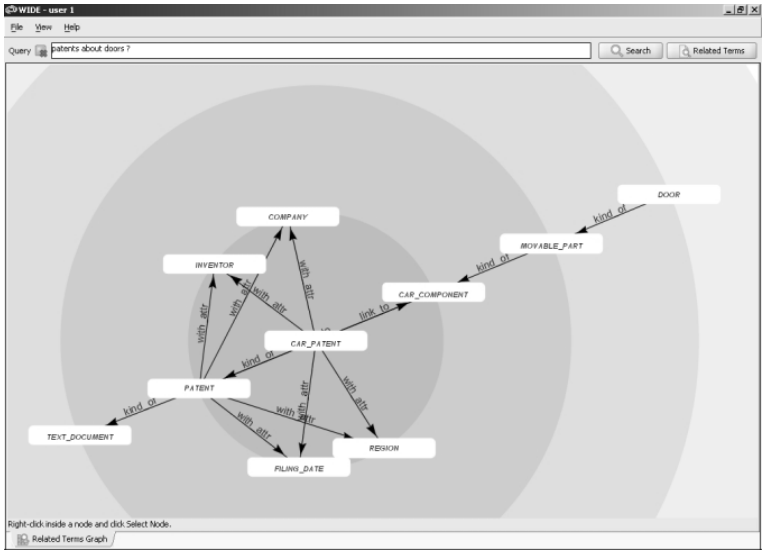

Fig 2. WIDE User Interface Front End

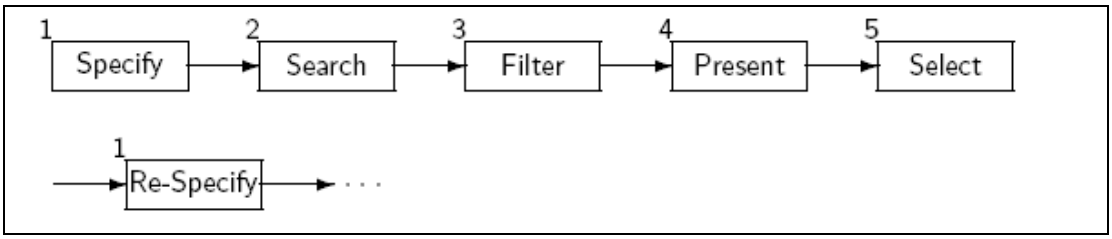

Fig 3. Classical Search Model

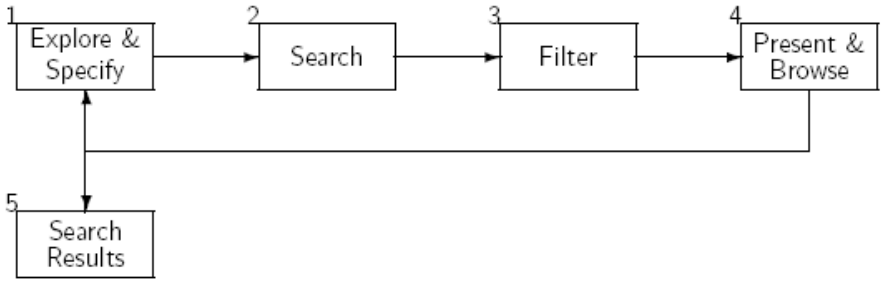

Fig 4. WIDE Search Model

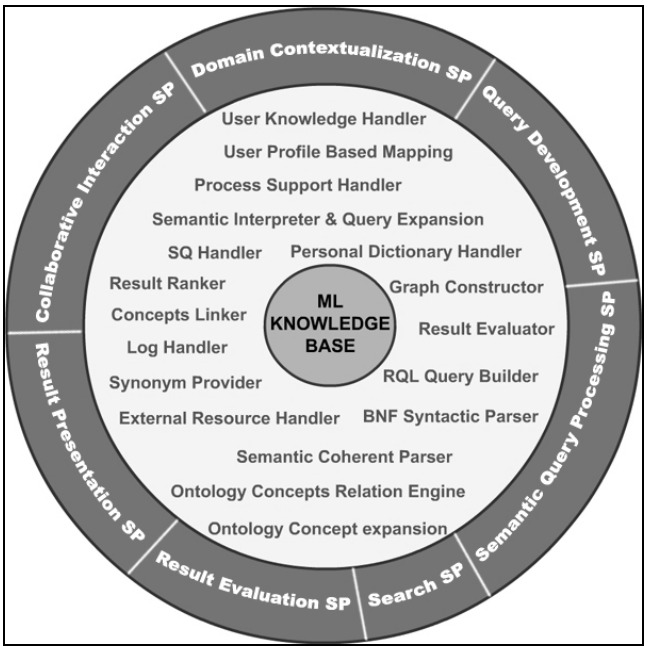

Fig 5. ML Architecture

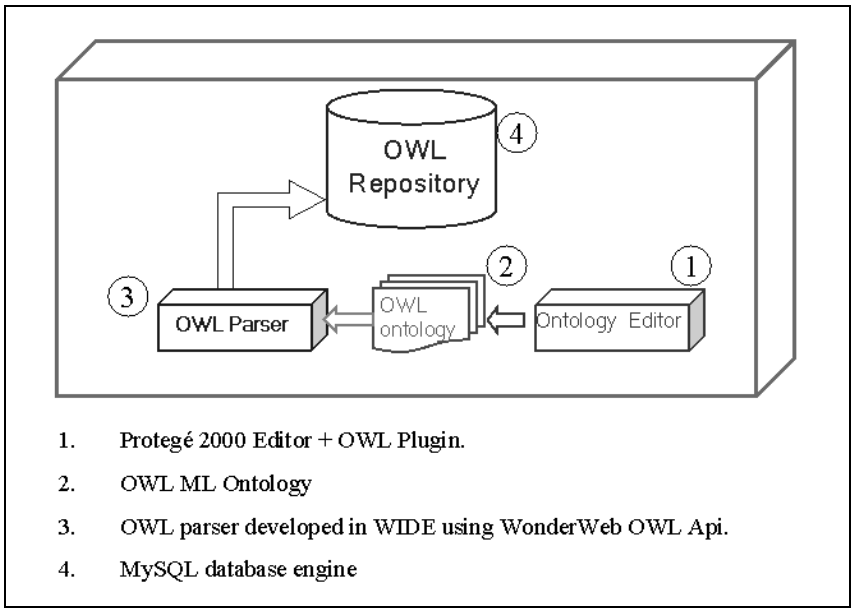

Fig 6. ML OWL Repository Architecture

Fig 7. Relationships Model

```
SELECT pt, mc, c1, c2, c3
  FROM {pt : $pt} @p {mc : $mc}, {rc1} @w_a1 {c1 : $c1}, {rc2} @w_a2 {c2 : $c2},
        {rc3} @w_v1 {v1 : Literal}, {rc4} @w_a3 {c3 : $c3}, {rc5} @w_v2 {v2 : Literal}
 WHERE @p = "has_info_about" AND ($pt = "DOCUMENT")
    AND $mc = "TEST" AND mc = rc1 AND @w_a1 = "link_to" AND $c1 = "CAR"
    AND ((c1 = rc2 AND @w_a2 = "with_attr" AND $c2 = "BRAND"
      AND c2 = rc3 AND @w_v1 = "with_value" AND v1 = "AUDI")
     AND (c1 = rc4 AND @w_a3 = "with_attr" AND $c3 = "CYLINDER"
      AND c3 = rc5 AND @w_v2 = ">" AND v2 = "8"))
```

Fig 8. System Query in RQL