AGATHE: an Agent and Ontology based System for Restricted-Domain Information Gathering on the Web

B. Espinasse*, F. Freitas**, and S. Fournier*

* LSIS UMR CNRS 6168, Universités d'Aix-Marseille, Domaine Universitaire de St Jerôme, F-13997, Marseille Cedex 20, France.

** Universidade Federal de Pernambuco, CIn-UFPE, Centro de Informática, Cx Postal 7851 50372-970 Recife, PE, Brasil

***Abstract*— Due to Web size and diversity of information, relevant information gathering in the Web is a very complex task. The main problem with most information retrieval approaches is neglecting the context of the pages, mainly because search engines are based on keyword-based indexing. In this paper we present the software architecture and the functionalities of the AGATHE system, a system for restricted-domain information gathering on the Web based on software intelligent agents and ontologies, based on the MASTER-Web system.**

***Index Terms*— multi-agents, information agents, agent-oriented software engineering, cooperative information gathering, information extraction, classification**

I. Introduction

Due to Web size and diversity of information, relevant information gathering in Web is a very complex task. Search engines such as Google, Excite and others - were designed adopting keyword-based indexing and retrieval methods. This approach, although robust, is inherently imprecise, and the output usually presents a good deal of irrelevant documents. Indeed, current search engines suffer from low precision rates because users are allowed only to perform statistically lexical-based searches that cannot provide the context underlying information that most users need.

The main problem with most information retrieval approaches is neglecting the context of the pages. We justify this statement with several reasons. First, the approaches of considering that a single system can provide users with information processing about any subject present in the Web is too ambitious: using only words, terms and its respective frequencies is probably not enough to carry out a task that no human being is capable of, i.e., to deal with documents about any subject in any language. In addition, search engines fail when they face the classical problems of natural language processing (NLP), particularly polysemy (a word with several meanings).

Leaving out context, retrieval approaches let slip off any form of organizing parts of the Web into specific information clusters. Clusters could contain classes of pages with related information, and such an approach would allow the recognition and classification of pages scattered on the Web, whose set can be identified as a class (e.g. the set of pages that displays data of the stock market, the set of pages about researchers and many others). With this context at hand, better information processing is possible and many other desirable tasks become feasible. One is the extraction of most of the relevant pieces of data that common pages of a class present (e.g. value of the dollar, topics of interest of a researcher). Another advantage is enabling users to state queries combining information of different classes of pages for instance, the retrieval of complex queries, such as the papers published in a certain series of conferences.

Departing from the MASTER-Web system (Multi-Agent System for Text Extraction and Retrieval over the Web) [3,4,5], which performs cooperative information gathering - we have defined a similar system reengineering its agent topology in order to assure flexibility, extensibility, scalability and reusability, among other benefits. The main technical challenge resided on the integration of techniques from Agent Oriented Software Engineering, Restricted-Domain and ontological based Information Gathering (Information Retrieval and Extraction).

This article is organized as follow: section II presents the restricted information gathering problematic and the relevance of software agents and ontologies to develop efficient information gathering systems. In section III, the main objective of the AGATHE system is defined, its general software architecture is presented, and its general functioning is introduced. In the following sections are presented in detail the internal architecture and functionalities of each of the three main subsystems of the AGATHE system: Search Subsystem (section IV), Extraction Subsystem (section V) and Front Office Subsystem (section VI). Finally, we conclude and present some perspectives to this research.

II. Related Work – The Restricted Information Gathering problematic

A lesson learned in Artificial Intelligence in the 70's, stating that knowledge works only over restricted domains, still holds for this task. Information retrieval (IR) researchers share this intuition of domain restriction; and this assertive is supported by some evidences.

Firstly, good part of the evaluation tests for IR systems are carried out over homogeneous *corpi*, whose texts are about one single subject, and many times its texts come from same source and not from sets of texts so varied in content and style as ones available in the Web. This fact gave origin to a new trend in IR: the *specialized* search engines [11]. They act on sufficiently specific domains of the Web, as daily news (as *Newstracker* - www.newstracker.com) and personal pages (*HPSearch* - http://hpsearch.uni-trier.de/).

Then, another evidence is the existence of the new field of *Information Extraction* (IE) itself, a recent branch of IR, whose objective consists on extracting relevant data from specific classes of pages to databases (therefore domain restricted), where it can be queried in complex ways. These queries can also combine information collected from many pages of geographically distributed sites. On the following we elaborate more on the theme. According to Kushmerick [9] information extraction "is the task of identifying specific fragments of a document that constitutes the nucleus of its semantic content".

There are basically two types of extraction systems: systems based on natural language processing and wrappers. Both types benefit from machine learning techniques to guarantee scalability and portability. These techniques provide a fast development of new extractors by the partial or total automation of the knowledge acquisition.

A. Cooperative Information Gathering

Extractors are very useful, particularly because users are much more interested in the pieces of information themselves than in the Web pages where these pieces are present. However, if, on one hand, they are domain-restricted, on the other they are too domain-restricted, i.e., some domains have relationships, that, if employed, would bring many benefits for cooperative extractors, such as accelerating the search of niches from which to extract data, finding safer contexts for these niches, and many others. So a cooperative approach is rather welcome in this setting.

Oates and al. [10] pointed out possible research lines with respect to knowledge-based information gathering over the Internet. Defining appropriately Information Gathering as the join of the processes of information acquisition and retrieval, this article proposes multi-agent systems, composed of independent and cooperative agents that can manage to integrate, to process and to construct consistent relevant niches of information excellent. Distributed Problem Solving (DPS) is recommended as a way to make the cooperative information agents negotiate among themselves for the discovery of these niches, and the intensive use of technologies knowledge-based techniques constitutes a premise for the solution proposed. The authors "assume the availability of means to generate "descriptors" which are representative of the content of the documents in terms of a set of domain primitives", mentioning explicitly extraction systems as technologies to be used in such a scenario. The work presented here shares this perspective, enhancing it with the clear perception that these two distinct tasks (acquisition and retrieval) are necessary to embody these concepts of cooperative manipulation into practical systems.

Ambite and Knoblock [1] have considered later information gathering agents that coincide exactly with the retrieval task of Oates and al. model [10]. Ambite and Knoblock's system presents the following functioning: databases of a vast digital library are grouped into a hierarchy of classes, where each class possesses its own agent, which is equipped with explicit knowledge about its class and the capabilities of the other agents. In typical database approach, agents construct retrieval plans which optimize the retrieval efficiency of complex queries which have to access many database tables. To carry this type of tool to the Web, other multi-agent systems would be necessary, also with specific agents for each class, which would fill up the databases tables with data extracted from Web pages, thus performing the above-mentioned task of acquisition.

The task of information acquisition must precede retrieval. It basically encompasses three tasks:

-- the *learning task* about domains departing from explicit models of these domains,

-- *data classification* and *extraction* by agents from pages pertaining to their domains. According to the authors' jargon, these pages pertain to the *information servers,* information sources where this data can be safely found, identified and processed. In the approach proposed by us, agents fit exactly with this second part of the problem, complementing the task of intelligent information retrieval over databases. These agents are in charge of populating databases with data classified and extracted from Web pages from *clusters*. The data can afterwards be queried and retrieved employing optimization techniques.

Thus, the couple formed by the two types of systems - acquisition and retrieval – enables us to provide useful and focused information retrieval from Web pages of specific domains.

B. Ontologies and their suitability for Cooperative Information Gathering

Our work is also based on the premise that, for the task of cooperative information gathering, declarative solutions are suitable instead of procedural ones, due to many reasons. First of all, it is much easier to maintain and modify a system whose knowledge is coded outside from the executed code. So, if the Web pages being treated change their patterns – and this is the case of the web with its dynamicity -, one has not to halt the system, recompile it and put it to use. Another good consequence of declarative solutions is easy reuse. When transporting the system to new domains, its users need only to replace its knowledge, which is outside the system; therefore, no code alterations are needed. The only task is reusing or creating ontologies about this new domain, together

with the cases where these rules are fired for extraction and classification.

Moreover, expressivity and clarity of declarative solutions constitute definitely an advantage: complex restrictions and relations, multiple inheritance hierarchies, and terminological reasoning can be mentioned among the many appealing arguments in favour of declarative, and particularly of ontologies' usage. The ontological engagement [6] of the available knowledge accelerates the processes of knowledge acquisition and transfer, once, in ontologies a more direct translation of the domain with its organization, theories, concepts, restrictions, relations and terminology could be included in the domain ontology.

Furthermore, the advent of ontologies also supported creation of high-level communication model known as "peer-to-peer", in which the concepts defined as domain knowledge are common to communicating agents, playing the role of shared vocabulary for communication among them. Within this model agents can express their intentions to the others by speech acts such as to inform, to ask, to recruit or to exchange messages, using this vocabulary.

To sum up, while traditional information retrieval systems employs keyword and statistics, a knowledge-based approach favours the inclusion of any sort of useful knowledge about the domain being treated and its pages, like sentences about page structure, regions, presence/absence of concepts in the texts and even linguistic patterns, when natural language processing takes part of the solution. Figure 1 displays from the main classes, slots and relations from the Web ontology, one of the ontologies used by AGATHE.

III. The AGATHE Gathering System

Our project takes place in the research field of agents and ontology-based information gathering. The resulting system, called AGATHE, is especially defined in order to treat all information gathering process. It performs information retrieval, information classification and extraction. Our project is innovative since it clearly tackles the extraction part of the problem. Enhancing our solution with ontologies, we manage to improve system flexibility quite a lot. Thus, cluster entities (domain knowledge) can be defined with proper granularity, representing the subtle grading differences among entities. We apply AGATHE to scientific events and tourism domains.

AGATHE is a system for restricted-domain information gathering on the Web, as a reengineering of the agent topology from MASTER-Web system [3,4,5], and includes some new capabilities such as multi-clustering search. These systems, which allow for gathering information, can be applied to several domains and use software agents technology and domain ontologies.

The first specific objective was to define an adaptative, scalable and extensible multi-agent architecture, in order to support our intelligent gathering system on the Web, once a MASTER-Web agent, beyond performing classification and extraction is burdened with all administrative tasks of retrieving, pre-processing and storing, which clearly cannot scale to deal with a large number of pages in a good timely manner. This new architecture is based on specific agent organization which defines different type of agents in interaction with specific roles. All agents are FIPA-compliant

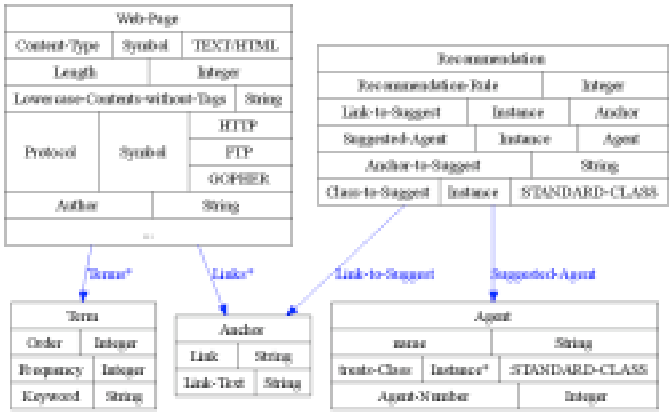

Fig. 1 main classes, slots and relations from the Web ontology used by AGATHE.

The AGATHE system is currently under development. This development integrates the JADE [7] multi-agents platform, the JESS [8] inference engine, for intelligent agent behaviours, and the PROTÉGÉ environment, for specification and handling of ontologies.

In this section, firstly we define the main objective of the AGATHE system, then we present the general architecture of this system, and finally, its general functioning is described.

A. AGATHE Objective

The objective of AGATHE system is to accomplish, for one or more users, intelligent information gathering concerning one or more specific domains. For example, such information gathering can concerns scientist community in a specific field. These researchers would be interested to have information on scientific events, as scientific international conferences or workshops, etc, to come in their scientific field. This application case has been used to develop the AGATHE system.

The first relevant set of information useful for the researchers' community related to these scientific events concerns Calls For Paper, from which, for example, date, place, title, sponsors, scientific domains concerned, privileged tracks, paper formats, etc can be extracted and organized in a database. This information set can concern one or more scientific domains. To each of these domains we associated specific domain ontology.

Again related to a scientific event, other pieces of information would be relevant for researchers. For example, information related to accommodation possibilities in the place of the event, information about transport possibilities, but also touristic information, such as famous places to visit (monuments, galleries, …), cultural events occurring in the same time period and the social program proposed by the event program committee. These additional information data concerns one or more new domains, domains that are defined by other specific ontologies.

B. General Architecture

As illustrated in the figure 2, the AGATHE system is

composed of three main subsystems in interaction:

(i) the *Search Subsystem* (SS), which is in charge of querying external search engines in order to supply pages to be selected and processed by the *Extraction Subsystem*;

(ii) the *Extraction Subsystem* (ES), composed of different *Extraction Clusters* (EC), each of them specialized in processing pages of a specific domain (such as Academia, Tourism, etc), and

(iii) the *Front Office Subsystem* (FOS), which is responsible for storing the data extracted from te processed pages and provide a interface for users (humans or other software agents).

These three main subsystems of the AGATHE system are multi-agent systems, composed of information agents. Some of them use domain ontologies to perform their tasks. Each of these subsystems will be described in detail in the following sections.

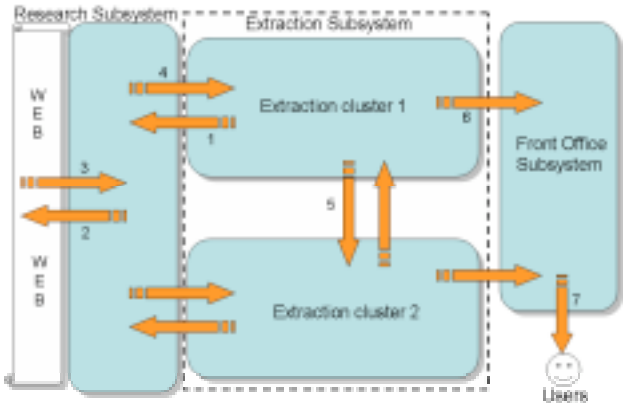

Fig. 2 General AGATHE architecture

AGATHE system functioning is illustrated in figure 2, together with the numbered sequence of the interaction arrows among its subsystems:

- Arrow 1: An *Extraction Cluster* of the *Extraction Subsystem* requests Web pages to the *Search Subsystem*.
- Arrows 2 and 3: The *Search Subsystem* works as a search meta-robot, retrieving Web pages, by querying existing search engines such as Google, Altavista and others.
- Arrow 4: These pages are then transmitted to the *Extraction Subsystem*, precisely to the exact agent in the *Extraction Cluster* that has done the request (arrow 1).
- Arrow 5: If necessary, recommendations are sent by this cluster to others clusters, in order to propose some pages that have potential interest for these others clusters.
- Arrow 6: Gathered information is transmitted to the *Front Office Subsystem*, in order to be stored in a specific database, which is accessible by users (arrow 7).

IV. The AGATHE Search Subsystem

As illustrated in figure 3, the *Search Subsystem* receives requests for Web pages from specific agents of a cluster. For instance, the Articles agent from the Science cluster asks for pages that contain keywords such as "introduction", "related work", "conclusion", and some others. Then the Search Subsystems forwards this query to existing search engines, gathers the Web pages and return them to the specific *Extractor Agents* from the *Extractor Cluster*. Three types of agents contribute to the information gathering process: (i) *Search Agents*, (ii) *Resource Agents* and (iii) *Supervisor Agent*. Each of these agents is described in detail in the following subsections.

The *Search Agent* performs the requests to existing search engines (Google, Altavista, Yahoo, …), It receives requests from an specific *Extractor Agent*,. After having merged the different results obtained by the various engines, the *Search Agent* directly transmits them to the *Extraction Subsystem*.

The *Resource Agent* is similar to a *Search Agent*, but it performs requests only to specialized resources of the Web. For example, for information related to the academic research, such resources can be the CITESEER site for publications, the DBLP site for authors, a specific web service, or a specialized database. It is requested by the *Supervisor Agent*, and it transmits its results directly to an specific Extractor agent in the *Extraction Subsystem*.

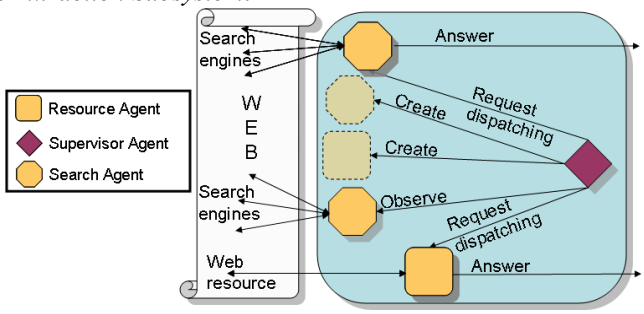

Fig. 3 Internal architecture of the Search Subsystem

The *Supervisor Agent* supervises the activity of the Search Subsystem. First, it takes delivery of requests from the Extraction Subsystem. Knowing the load of various Search and Resource Agents, it manages for the better allocation of these requests to be treated. According to the need, it creates (or even deletes) some Search and Resource Agents. For performance reasons, the Supervisor Agent can also decide migration of these Search Agents to CPU less loaded. The Supervisor Agent manages also subscriptions permitting the Extraction Subsystem to receive periodically results from a specific information request. The strategy used can be called "push" and the results are directly transmitted, without the necessity of formulating further requests.

V. The AGATHE Extraction Subsystem

In AGATHE, the realisation of this subsystem is based on the MASTER-Web system, which has been developed in Brazil [3,4,5]. This system is a multi-agent system for information retrieval and extraction over the Web, composed of one kind of agents that performs various tasks of classification and extraction, thanks to one domain ontology.

To assume an efficient contextualisation of the information treatment implying several knowledge domains, the AGATHE's proposal is a bit more ambitious: its Extraction Subsystem is composed of a set of extraction "cluster" (while MASTER-Web was designed to treat only one "cluster"). Each of these clusters is related to a specific domain, at which is associated a specific ontology, but there could be pages

relating both (e.g. a Call for Papers page, which will be processed by the Science cluster, usually brings information about trips, hotels, social and cultural events which are simultaneous or in dates near to the conference, and so on. These ontologies are manipulated by the Protégé environment. The figure 4 shows an example of part of ontology in the field of academic research and concerning publication, CFP (Call For Papers).

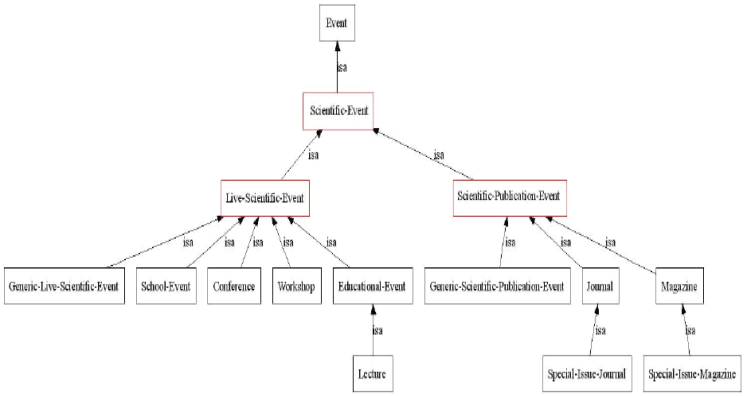

Fig. 4 A part of a domain ontology concerning scientific events.

The Extraction Subsystem has the following tasks to carry out in sequence: (i) to generate Web pages requests to the Search Subsystem; (ii) the main task, to treat the results of these requests (the Web pages that the Search Subsystem has returned to it). This last task consists in a validation, a functional classification, and an information extraction.

As illustrated in the figure 5, each of these Extraction Clusters is a multi-agent system performing the classification of the Web pages, and an information extraction on these pages.

The various agents that compose the cluster are: a set of *Extractor Agents* and of *Preparation Agents*, a *Supervisor Agent*, a *Recommendation Agent*, and a *Storage Agent*. All of these agents are cognitive agents. They employ the Jess inferences engine in their reasoning for the tasks of classification and extraction. The behaviours of these agents are mainly specified by production rules. Most of them exploit ontologies by using the Protégé plug-in JessTab.

The *Preparation Agents* receive Web pages from the *Search Subsystem*. These agents are created by the *Supervisor Agent* of the considered extraction cluster and are deleted by this same agent when they are not being used. These agents perform the first treatments, thus permitting more easily to exploit the Web pages obtained by the *Search Subsystem*.

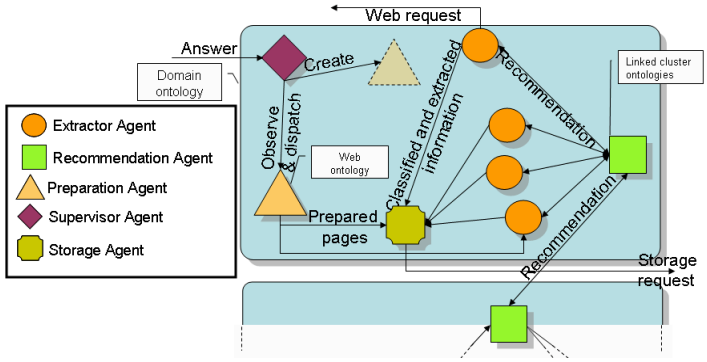

Fig. 5 Internal architecture of an Extraction Cluster

These treatments are inspired in the treatment performed by MASTER-Web [3,4,5], which are explained below:

-- *Validation*. This treatment verifies if Web pages obtained are in HTML format, if pages are accessible, and if the pages are always stored in the data base. The pages that do not verify these conditions will not be considered in the following treatments.

-- *Pre-processing*. This treatment identify the content, the title, the links, and the email, using techniques of information retrieval and eventually, if necessary, natural language techniques.

-- *Functional classification*. This treatment is knowledge-based and use the inferences engine Jess and exploit ontology related to the Web. This Web ontology presented in the figure 1 concerns some Web concepts. Thanks to a specific knowledge base (production rules), the *Preparation Agent* use this ontology to classify the Web pages according a functional aspect. The functional categories obtained are pages formed by messages, the lists, (i.e. of links, of persons), the auxiliary pages (pages that contain some pieces of information but don't represent an entity, e.g. a separate page of topics of a conference which has its own page), the pages selected for extraction, and finally pages considered as not valid.

Indeed, to achieve more performance and flexibility some of these treatments have been extracted from the original MASTER-Web agents. Then, it is now possible to duplicate such agents when the load over some of them is too heavy.

The *Extractor Agents* composing the *Extraction Cluster* are associated to a specific domain ontology linked to the cluster. The task of these agents is to perform semantic classification and then information extraction in the Web pages that they receive from the *Preparation Agents*. Each *Extractor Agent* of the considered cluster is associated to a particular class (concept) of this ontology. As the *Preparation Agent*, the *Extractor Agent* is based on the extraction and content classification of the MASTER-Web agent, and uses Jess and JessTab [2] to benefit from this ontology.

For instance, within the domain of science, sub-domain of scientific events, a particular *Extractor Agent* is associated to the concept or class "*Call For Papers*" of the ontology and performs filtering and information extraction in the Web pages it receives. These Web pages result of an information request performed on the keywords "call for papers", "*Call For Participation*", and other similar ones, which were retrieved by the *Search Subsystem* by request from this specific agent.

The extracted information is then transmitted to the *Storage Agent*. *Extractor Agents* may also classify pages. For example, Agent defined as "*Call For Papers"* agent could classify different call for papers for conferences, journals, book chapters and many other classes from the Science ontology..

The *Recommendation Agent* receives prepared pages from *Preparation Agent* and dispatches them to other agents in same cluster or to other cluster. It performs three main tasks:

-- Recommend pages/links that have some interest to other *Extractor Agents* of the cluster.

-- Recommend some pages/links to other *Extraction Clusters*, pages that could be interesting for them (an example is explained just after the next figure). For this task, the Recommendation Agent has to know all the ontologies of the different clusters involved. It dispatches these pages to the various *Recommendation Agents* of the *Extraction* cluster concerned.

-- Dispatch pages that have been recommended by another *Recommendation Agent* of other *Extraction Cluster*.

Two clusters are linked if it exists contextual link between them. For instance, the agent responsible for scientific events can suggest for a Tourism cluster some information found on "call for papers" pages, information which are related to accommodation and transports facilities to participate to such scientific events.

The *Storage Agent* is in charge of storing the extracted/classification information in the database of the *Front Office Subsystem*. This agent does not actually perform the storage; instead, it only prepares the storage. It treats this information to conform to the storage format, according some information from the storage structure of the database. Then these formatted data are transmitted to the *Front Office Subsystem* to be physically stored in the database and exploited by the users.

The *Supervisor Agent* of the *Extraction Cluster* has functionalities similar to one of the *Supervisor Agent* of the *Search Subsystem*. It supervises *Preparation Agents* activity. First of all, it receives the request results from the *Search Subsystem*, then it creates one or more *Preparation Agents* that will treat these results before transmits to the *Recommendation Agent* of the cluster considered.

VI. Front Office Subsystem of AGATHE

The *Front Office Subsystem*, figure 6, is the subsystem supporting user interactions with the AGATHE system. Users exploit the extracted information produced by the *Extraction Subsystem*, in exploiting a relational database where these results are stored.

The F*ront Office* Subsystem is composed of two main components: the *Mediator* and the *User Interface*. The first has for task to update the database, from the extracted information transmitted by the *Extraction Subsystem* (*Storage Agent*). The second supports user interactions with the AGATHE system

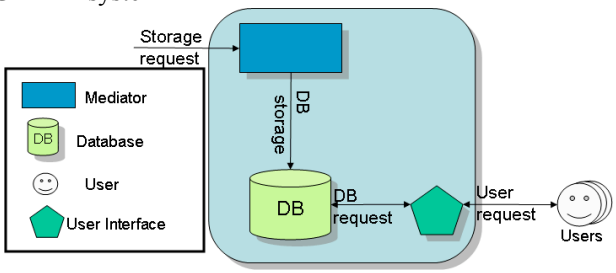

Fig. 6 Front Office internal architecture

VII. Conclusion

In this paper we have presented the AGATHE system, mainly its software architecture, and functionalities. This system performs information retrieval, information classification (Web pages), and information extraction (in these Web pages) with the use of domain ontologies and Web ontology.

AGATHE is composed of three main subsystems in interaction: the *Search Subsystem* which realise request on the Web through search engines, the *Classification-Extraction Subsystem*, which perform an information extraction from pages obtained by the previous subsystem, and *Front Office Subsystem*, which permits to the users to exploit the extracted information stored in a relational database. These subsystems are composed of information software agents. Some of them are able to use ontologies and perform tasks, as classification, filtering or extraction by reasoning.

Today, a first prototype version is running and some results have to be soon produced in order to assess the benefits generated by the new agent architecture in comparison with MASTER-Web results.

References

[1] Ambite, J., Knoblock, C.: Agents for information gathering. In Software Agents. Bradshaw, J. (ed.), MIT Press, Pittsburgh, PA, USA, 1997.

[2] Eriksson H., Using JessTab to Integrate Protégé and Jess; IEEE Intelligent Systems, 18(2):43-50, 2003

[3] Freitas F., G. Bittencourt, *An Ontology-Based Architecture for CooperativeInformation Agents,* IJCAI, 2003.

[4] Freitas F., G. Bittencourt**,** *Cognitive Multi-agent Systems for Integrated Information Retrieval and Extraction over the Web*, IBERAMIA-SBIA, 2000

[5] Freitas F., G. Bittencourt, J. Calmet, MASTER Web: An Ontology-Based Internet Data Mining Multi-Agent System, Second International Conference on Advances in Infrastructure for E-Business, E-Science and E-Education, 2001.

[6] Gruber, Thomas R.; Towards Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal of Human and Computer Studies*, 43(5/6): 907-928. 1995.

[7] JADE, JADE tutorial http://jade.tilab.com

[8] Jess, http://hezberg.casandia.gov/jess.

[9] Kushmerick N., "Gleaning the Web". IEEE Intelligent Systems. EUA 1999.

[10] Oates T., M.V. Nagendra Prasad and V. R. Lesser, Cooperative Information Gathering: A Distributed Problem Solving Approach. Computer Science Technical Report, University of Massachusetts, 1994.

[11] Steele R., *Techniques for Specialized Search Engines.* http://www-staff.it.uts.edu.au/~rsteele/SpecSearch3.pdf