Domain-independent ontologies for Cooperative Information Agents

Mario Gomez, Chema Abasolo, Enric Plaza

IIIA - Artificial Intelligence Research Institute
CSIC - Spanish Council for Scientific Research
Campus UAB, 08193 Bellaterra, Catalonia, Spain.
{mario,abasolo,enric}@iiia.csic.es
http://www.iiia.csic.es

Abstract. Cooperative Information Agents and modern information systems in general have to access large amount of information distributed across multiple heterogeneous sources. A great challenge of such systems is to evolve by adding new information sources or adapting the existing components for different domain knowledge. We propose the UPML framework as a methodology to build Information Agents by reusing a library of problem solving components that are defined in a domain-independent manner. Moreover, the UPML language is used as an Agent Capability Description Language(ACDL) suitable to configure or build an application. From this approach, a new application can be build by linking the components of the library with a particular domain and a collection of heterogeneous information sources. Adaptability and dynamic configuration of such a system is achieved by reasoning about the UPML specifications of agent capabilities. Independence of the domain and semantic interoperability are achieved by using ontologies and bridges (mappings between ontologies), while independence from the information sources is based on the use of ontologies to overcome semantic heterogeneity and wrappers to achieve syntactic interoperability.

Keywords: Information Agents, Intelligent Information Integration, Ontologies, Reuse, Heterogeneous Information Sources, Problem Solving Libraries.

1 Introduction

Modern information systems shall manage or have access to large amounts of information and computing services. The different system components can conduct computation concurrently, communicating and cooperating to achieve a common goal. These systems have been called *cooperative information systems*[3]. One major goal of this field is to develop and build information systems from reusable software components. This goal can be achieved by assembling information services on demand from a montage of networked legacy applications and information sources [4]. A promising approach to this problem is provided by Cooperative Information Agents, computational software entities that accesses

one or multiple, heterogeneous and distributed information sources [1]. A critical requirement for these systems is the independence of the reasoning components from the domain knowledge. We propose the UPML framework as a methodology to build Information Agents by reusing a library of problem solving components that are defined in a domain-independent manner. Moreover, the UPML language is used as an Agent Capability Description Language. ¿From this approach, a new application can be build by linking the components of the library with a particular domain and a collection of heterogeneous information sources. Independence of the domain and semantic interoperability are achieved by using ontologies and bridges (mappings between ontologies), while independence from the information sources is based again in the use of ontologies to overcome semantic heterogeneity and wrappers to achieve syntactic interoperability.

We have built an application that shows how to build Cooperative Information Agents by using the UPML framework the Web Information Mediator. The overall goal of WIM is to provide a mediation service for information tasks of a professional user. A mediator is an agent that offers an added value to the information sources it accesses[10]. Typical services offered by a mediator include selection of information sources, information retrieval, and fusion of information from different sources. We have built a library of components to solve this kind of tasks belonging to the field of Intelligent Information Integration (I3) [11]. WIM is a multiagent information system dealing with the problem of looking for medical literature, thus it has been build by connecting the components in the I3 Library with a medicine domain and some web-based information sources that serves bibliographic references in medicine. We want to emphasize the independence between the library (I3), the domain knowledge (medicine) and the external information sources.

The overall description of UPML is presented in §2, the I3 Library is described in §3. The WIM application is described at the conceptual level in §4. The "reuse" question is addressed in §5. The WIM multiagent architecture is described in §6, including a brief discussion on the use of UPML as an ACDL in §6.1. Finally, some conclusions are summarized in S7.

2 An overview of UPML.

The goal of *software architectures* is learning from system developing experience in order to provide the abstract recurring patterns for improving further system development. As such, software architectures contribution is mainly methodological in providing a way to specify systems. A software architecture has the following elements: (i) components, (ii) connectors, and (iii) a configuration of how the components should be connected [2]. UPML [5] is a software architecture for knowledge systems where its components are tasks, problem-solving methods and domain models. The connectors in UPML are called bridges and the configuration is shown in Fig.1.

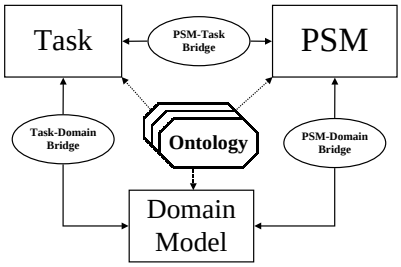

Fig. 1. The UPML configuration. Note that the relation between an ontology and a component is that the component use this ontology.

First we will briefly explain the basic elements of the UPML architecture, and second how this UPML architecture addresses the problem of the domain-independence.

2.1 UPML components

There are three classes of components in the UPML framework: tasks, problem-solving methods and domain models.

Tasks define the goals we have in solving problems, that is, a task specifies what we want to do. A task is characterized by preconditions and goals. A Task specifies constrains and properties to be satisfied for a Problem Solving Method (PSM) that can solve a particular task.

Problem-Solving Methods (PSM) describe which reasoning steps and which types of knowledge are needed to perform a task. A PSM specifies a particular way to solve a task. The main attributes of a PSM are the input/output roles, plus the preconditions and postconditions to be fulfilled by the input and output roles. There are two subclasses of PSM: *Problem Decomposers* and *Reasoning Resources*. Problem Decomposers specify decomposition of tasks into subtasks. Reasoning Resources specifies how to solve a task, it does not describe its internal structure, which is regarded as an implementation aspect.

Domain models model the domain knowledge that can be used by tasks and PSMs. A domain model is characterized by a domain knowledge and its properties.

The UPML framework has the notion of *Ontology*. An ontology defines a terminology and its properties. UPML has different ontologies, one for task's description, one for PSM's description and the domain ontology for domain models. The fact of describing tasks, PSMs and domain models with different ontologies, makes task and PSM independent of the domain. This independence enable the reuse of task descriptions in different domains, the reuse of PSMs across different tasks and domain, and the reuse of domain knowledge for different tasks or PSMs.

UPML, as a software architecture, has a specific kind of connectors called *Bridges*. A bridge models the relationship between two different components. The function of the bridges is to connect components with different ontologies,translating the concepts of the components ontologies. A bridge provides mapping axioms and assumptions about the components that it relates. There three kinds of bridges: Task-PSM, Task-Domain Model and PSM-Domain Model (see Fig.1).

2.2 UPML and Domain-Independence

Once we have described the UPML components, let's see how the UPML framework addresses the issue of domain-independence [6]. First of all we have to differentiate two concepts: *Library* and *Application*.

A *Library* is a collection of UPML descriptions of tasks and the PSMs. A Library is totally independent of the domain because tasks and PSMs are described in terms of their own ontologies, and not in terms of the domain ontology.

An *Application* is made of one or more libraries, a set of domain models, and the bridges linking those library's components with the domain models employed. The mapping axioms of the bridge allow to translate the concepts of the domain model's ontology into concepts of the PSM's ontology. This translation enable the PSM to work with the domain knowledge of the domain model.

This approach makes the library independent of the domain. This independence allows the library to be reusable, in the sense that the same library can be used with different applications, with different domain models or even with different domains.

In our architecture, the agents define their capabilities in terms of the UPML descriptions of tasks and PSMs. The agents register their capabilities in a Library. We will see this link between agents and UPML in §6.

3 The I3 library

The I3 library offers a collection of methods to solve some of the usual tasks carried on by Information Agents (see for instance [13] [14] [15] [16][17], which are typically known as Intelligent Information Integration[11]), a concept that originally means the integration of multiple databases and nowadays is being focused to the integration of multiple web sources with the use of ontologies [7].

Instead of adopting the information retrieval approach (IR), we adopt a vision more close to meta-search. IR focuses on improving retrieval methods, while meta-search focus on exploiting existing "information sources", where each resource posses a specific content accessible by a "local" retrieval engine. For this reason, the library and the WIM application focus on this process -and do not include components that can be found inside retrieval engines.

A second consideration is the paradigmatic distinction between the concept of the "relevance" in classical IR and the more rich conceptualizations currently in use for intelligent information agents [12]. The canonical concept of relevance

in IR is a test where the results for of a query by a retrieval engine are compared to a gold standard provided by a human expert that assesses false positives and false negatives. The problem of that approach is that "relevance" is independent of the purpose of the specific user in posing a query. The I3 Library includes some methods to elaborate and rank results according to an utility measure.

I3 can be seen as an adaptation process with four subtasks: transformation of the user consultation , selection of information sources, information retrieval and integration of information from multiple sources.

Adaption refers to the process of elaborating the user consultation to better fulfill his interest, as well as adapting the queries for the idiosyncratic syntactics of each information source. Once the retrieval is performed, the results from different queries should be aggregated (§3.2) for each source, and finally the results for each source are also aggregated to obtain a unique result for the user consultation.

We have adopted a general approach for the overall process of information integration that is based on using query weighting and numerical aggregation operators. Query weighting refers to the process of assigning weights to queries generated according to some domain knowledge, while numerical aggregation operators are the mechanism used to merge items coming from different queries -and sources-, and combining the different weights to obtain an unique weight for each item summarizing the relative importance of that item for the user interest. This mechanism allows to score documents retrieved from engines that originally do not give any score, and defining user-oriented utility measures simply by defining the appropiate knowledge categories (see §4.1).

3.1 Adaptation of queries

We adopt a very well known approach to queries as vectors of keywords instead of complex database query languages. This decision is justified because nowadays professional databases could often be accessed through the use of a web-based search-engine, where queries are made of keywords belonging to a particular domain. We also include search filters as optional constrains allowing to restrict a search procedure. A bibliographic ontology have been used to model the kind of filters allowed by professional bibliography search-engines like publication date, language and so on. Let's see the both types of query adaptation, adaptation with respect to the domain, and customization for particular information sources.

Query elaboration: refers to the adaptation of queries for a particular user interest, within a particular domain. This task can be achieved by using domain knowledge, like synonyms or predefined knowledge categories.

Query customization: a query is customized to a particular information source by translating keywords and filters into search modes and filter of each selected source. This task is different from the task achieved by wrappers, where keyword-based queries are transformed in the particular syntax of the source, following the rules and particular features of each source at the lowest level.

Selection of sources: it isn't a query elaboration method, but is needed when more than one source is available, so it is very related with the query adaptation

process, and in particular, with the query customization task. The selection of sources could be done by asking the user or by using a particular method, like Case-Based Reasoning (CBR) . In the current version of WIM three information sources are available (see section §4.1).

In our approach to query adaptation, not only new queries are enriched with domain knowledge, but these queries are weighted according to the strength of the relation between the original and the elaborated query. For example, in a query elaboration with synonyms, new queries are weighted according to the correlation coefficients between the original keywords and the new ones. Let's see some examples of the query adaptation methods.

3.2 Aggregation of results

Aggregation is a kind of merging where the rankings assigned to the repeated apparitions of an item are combined using an aggregation operator to obtain a unique ranking.

A numerical aggregation operator is necessary because of the nature of the query adaptation procedure, where queries are weighted with numerical values. Hence, the results for a query inherit the weight of a query. It means that even results non ranked by the retrieval engine can be ranked, by using the weight associated to the query for which they are a result. If the queries are weighted according to "utility" rather than relevance -see the example of query elaboration with categories-, then the results will also be ranked taking into account these utility criteria.

Three different numerical aggregation operators have been implemented as problem solving methods in the the library: the weighted-mean, the Ordered Weighting Average (OWA) and the Weighted OWA[19].

4 The WIM application

The core of WIM is the I3 Library, but there is other components need to build a complete application (See §2.2). In our approach, an application is builded by linking the reasoning components in the library in the library with a particular domain knowledge and a collection of information sources.

4.1 Linking the Library with the domain knowledge

Domain knowledge do not belongs to the library; this is one of the most important feature of the UPML approach, because the independence from the domain is considered a basic requirement to achieve one of the most challenging goals of the knowledge modelling and the software engineering communities: reuse of existing components[8].

The domain chosen to build the WIM application is medicine, and in particular, *Evidence-Based Medicine* (EBM). EBM proposes a medicine practice which calls for careful clinical judgment in evaluating the "best available evidence"[21].

The main task for the WIM application is looking for medical literature, and the utility criteria used to rank documents are those given by the EBM to asses the quality of medical bibliography. Hence, we need also some bibliographic knowledge to describe queries and results for the queries, and some knowledge about the information sources selected for the application: PubMed and HealthStar. Let's see the different domain models and how are they used by PSMs to solve a task.

The domain knowledge is organized into four domain models.

- A general *medical thesaurus* is used to select the keywords to pose a query and during the elaboration of the queries. We choose the MeSH, a thesaurus that can be accessed through a web-based retrieval engine called MeSH Browser. This domain model is used by the PSM *query-elaboration-with-synonyms*.
- An ontology about *bibliographic data*, used to describe the filters typical in bibliographic search engines. After considering standard bibliographic ontologies like the Stanford Bibliographic Data, we decide to use only a limited set of bilbiographic concepts, those used to build queries and process the results. This domain model is used to pose the queries and by the query-customization PSM.
- A collection of *source descriptions*, where the search modes and allowed filters of each source are described, including the translation sentences between the common bibliographic ontology and the particular terms used by that source. This is used by the *query-customization* method.
- A collection of predefined categories describing the keywords and filters that are useful to rank documents according to the EBM criteria, plus. Used by the PSM *query-elaboration-with-categories*.

Example 1:The query weighting approach adopted for the query adaptation task has great advantages to rank documents by different criteria, not only classical IR's relevance. To introduce new utility criteria we build a method to elaborate queries by using predefined knowledge categories. A category is a collection of terms and filters associated to one topic, which are weighted according to the strength of that association. For example *Good-Evidence* is Category that defines some filters to get only papers based on a good evidence quality. Two of the filters are the following:

```
(Publication Type = "Meta-Analysis", Weight = 1)
(Publication Type = "Randomized Controlled Trial", Weight = 0)
```

Given the query Q = (Levofloxacin, Pneumonia) and applying the PSM *Query-expansion-with-categories* with this category, we get the following set of queries:

```
Q1 = (Levofloxacin, Pneumonia, Publication Type = Meta-Analysis),
     Weight = 1.
Q2 = (Levofloxacin, Pneumonia, Publication Type = Randomized Controlled Trial),
     Weight = 0.9
```

4.2 Linking the Library with the information sources

Information sources are not domain models, they are external PSMs that solve the *retrieval* task. But there is a domain model consisting of source descriptions, as explained in previous section (§4.1).

Example 2: The *query-customization* PSM expands a query expressed in a source independent way in a collection of queries in terms of a particular information source, by using the search modes and filters allowed by that source. This knowledge is described in the sources domain model, for example, this is our description of the HealthStar information source -when accessed through the retrieval engine Internet Grateful Med (IGM):

- Search Modes: (Subject, weight = 1), (Title, weight = 0.5)
- Filter Translations: (Begin Year = begyear), (Publication Type = publication)

Given the query Q = (AIDS, Diagnosis, Begin Year = 1980), the resultant set of queries, after applying the *Query-Customization* method is given below:

- Q1 = (Subject = AIDS, Subject = Diagnosis, (begyear = 1980), Weight = 1)
- Q2 = (Title = AIDS, Subject= Diagnosis, begyear = 1980), Weight = 0.5
- Q3 = (Subject =AIDS, Title = Diagnosis, begyear = 1980), Weight = 0.5

New sources may be added to the application by including their descriptions according to the source domain model, and building the appropiate wrappers (task-psm bridges) between the *retrieve* task and the retrieval engines.

5 Reusing knowledge components in WIM

UPML has been defined as "a framework for knowledge system reuse"[6]. Components reuse is achieved in UPML by separating the specification of the different components of a software architecture. The separation between domain and PSM has been extended to the separation of Task and PSM specifications to maximize reusability. Ontologies play a crucial role in modern knowledge systems to separate the reasoning components from the domain knowledge. In fact, ontologies are often defined as "shared conceptualizations" or "shared terminologies", putting the emphasis in the function of an ontology as a vehicle to achieve interoperability between different entities. Separation of components is needed to achieve reuse, while ontologies are needed to link the separated elements. The linking can be automatized if the ontologies are defined in a formal manner

There is a class of components in UPML defined specifically to achieve reuse: bridges. A bridge can be seen basically as a mapping schema between two ontologies[9]. There are three kinds of bridges: Task-Domain, Task-PSM and PSM-Domain bridges. The tasks and PSMs in the library are expressed using the same ontologies, so there is no need for Task-PSM bridges within the library. Some of the domain models are builded ad-hoc for the WIM application, therefore they are expressed using the ontologies in the library. The consequence is that we do not need bridges to connect the library with this domain knowledge. In

fact, we only need Task-Domain and PSM-Domain bridges when the domain is provided by a third party using its own ontology. An example which needs a bridge is given by the PSM *query-elaboration-with-synonyms*, that uses an external domain knowledge, the MeSH thesaurus. MeSH do not speak explicitly of synonyms, instead of synonyms it uses the terms "See also" and "Related terms" to indicate semantically related terms. We need a bridge to map between the concept of "correlated term" (synonyms are terms with a correlation near to 1) used in the PSM *query-elaboration-with-synonyms*, and the concepts used in MeSH

We say that we do not need Task-PSM bridges within the library. However, this kind of bridges are needed to connect the retrieval task with the external information sources. The retrieval engines of such information sources can be seen as external PSMs, thus they are not defined using the ontologies in the library. Those components that are not local just view the whole *web as a library*. The philosophy here is WIM to have wrappers (adapters) to external resources of data and/other services. The wrappers play two roles: a) they hold the UPML description of the resource; b) they effectively support the interoperability with the web resource.

6 The WIM multiagent architecture

This section deals with the mapping between the knowledge components of WIM -within and without the library- and multiagent systems.

Different multiagent architectures could be used to build an application upon a library of knowledge components. This section will overview the kind of agents used in the WIM system together with a brief comment about some interoperability issues and the use of UPML as an Agent Capability Description Language (ACDL).

We distinguish five classes of agents in WIM:

Problem-solving agents: they are the agents that solve the tasks in the library. The tasks they are able to solve and the avilable PSMs are both specified in UPML and registered to the Librarian.

The Librarian: The librarian holds the UPML specifications of the components in the library: tasks and PSMs. UPML is the ACDL for describing the competencies of the Problem-solving agents: the PSMs they provide and the tasks they are able to solve.

Wrappers: They are the ones that accesses the heterogeneous information sources, solving the interoperability problem at the syntactic level and technical levels, while the customization PSM deals with the semantic level. From the UPML point of view, a wrapper is a software implementation of a bridge between a task in the library and an external PSM that solves that task. From a MAS perspective, wrappers are components used to agentify components that are not agents, like external information sources.

Application agents: These agents are necessary to build and run a new application by selecting, configuring and communicating with the agents in the

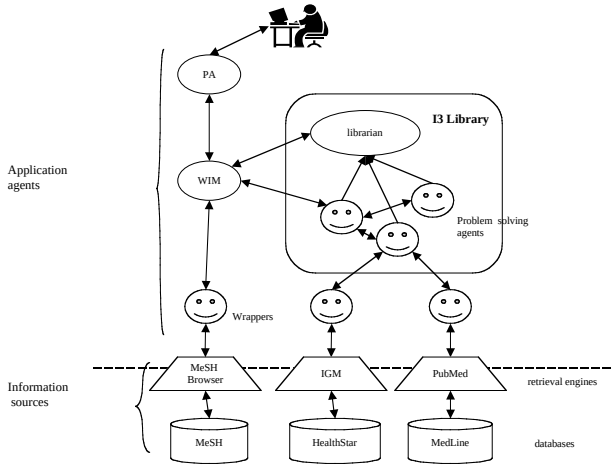

Fig. 2. WIM multiagent architecture

library. We consider at least two roles to be carried on by the application agents: brokering among the configurable components of the library, and mediating between the user's personal agent and the problem solving agents.

Personal agents: They are responsible of of interacting with the user asking for data, presenting him back the results of a consultation and holding information about the user preferences and a history of their previous requests. Information about the user preferences and the history of previous cases is used to help configuring or adapting the problem solving library for his particular interest, or for other users in a group oriented environment.

6.1 UPML as an ACDL

Describing agent capabilities is an active field of research, as this is an important issue to achieve interoperability in open agent architectures. The most widely adopted approach to solve the interoperability problem is the combined use of middle agents[20] and agent capabilities description languages (ACDL). An example of an ACDL is LARKS, and ACDL that is being included in the RETSINA multi-agent infrastructure[18] to be used within the matchmaking process. (http://www.cs.cmu.edu/ softagents/interop.html).

The advantage of UPML is that it is able to describe the both the tasks an agent is able to solve, plus the methods it is equipped with to solve that task, and these descriptions are domain independent. UPML as and ACDL allows developing Information Agent with independence of the domain, an critical issue to achieve component reuse and scalability.

7 Conclusions

We have presented a methodology (UPML) and an implemented cooperative information agents system developed using that methodology (WIM). The methodology aims at maintaining a strict independence between domain (domain knowledge and ontology) and the information agents (with separate ontologies). The methodology is based on using UPML as an agent capability description language. UPML defines bridges as a type of connectors that link domain-independent description of agent capabilities to concrete domain models. The advantage that UPML allows the agent to talk about both the *tasks* they are able to solve and the *methods* by which they are able to solve them. Moreover, we have distinguished between library and application. A library is a repository of UPML descriptions of agents capabilities. An application is built selecting a collection of agents and developing the bridges required to link library-registered agents with domain resources and domain ontologies.

We have described the WIM application for medical information. Agents in WIM work with a domain independent ontology—the same by which tasks and PSMs are described. The WIM application is built by linking the "knowledge requirements" specified in UPML to specific domain resources using bridges—examples shown are the MeSH thesaurus and the EBM models. We have left out of the scope of this paper the brokering process, i.e. the process by which a collection of PSMs (and their corresponding agents) are selected to realize an application. We are currently developing several prototypes of automatic brokering that use UPML descriptions to configure on the fly an application for a given "input" task specification in the framework of the IBROW project (http://www.swi.psy.uva.nl/projects/ibrow/home.html).

References

1. M. Klusch (ed) Intelligent Information Agents. Springer, 1999
2. D. Garland and D. Perry (Eds.). Special issue on software architectures. *IEEE Transactions on Software Engineering*, 1995.
3. Second International Conference on Cooperative Information Systems (CoopIS-94). Springer, 1994.
4. G. De Michelis, E. Dubois, M. Jarke, F. Matthes, J. Mylopoulos, K. Pohl, J. Schmidt, C. Woo, and E. Yu. Cooperative Information Systems: A Manifesto. In 4th Intl. Conf. on Cooperative Information Systems, Brussels, Belgium, 1996.
5. D. Fensel, V. R. Benjamins, M. Gaspari S. Decker, R. Groenboom, W. Grosso, M. Musen, E. Motta, E. Plaza, G. Schreiber, R. Studer, and B. Wielinga. The component model of upml in a nutshell. In *Proceedings of the International Workshop on Knowledge Acquisition KAW'98*, 1998.
6. D. Fensel, V. R. Benjamins, E. Motta, and B. Wielinga. UPML: A framework for knowledge system reuse. In *Proceedings of the International Workshop on Knowledge Acquisition KAW'98*, 1998.
7. D. Fensel, C. Knoblock, N. Kushmerick and M. Rousset Workshop on Intelligent Information Integration Workshop Report. American Association for Artificial Intelligence, 2000

8. E. Motta Reusable components for Knowledge Modelling. Frontiers in Artificial Intelligence and Applications, 53 IOS Press, 1999
9. D. Fensel,S. Decker, E. Motta and Z. Zdrahal Using Ontologies For Defining Task, Problem-Solving Methods and Their Mappings. In *Proceedings of European Knowledge Acquisition Workshop (EKAW'97)*, Lecture Notes in Artificial Intelligence (LNAI), Springer-Verlag, 1997.
10. Wiederhold, G. Mediators in the architecture of future information systems. In *IEEE Computer* 25, 3 (March), 38-49, 1992
11. Wiederhold, G. Intelligent Integration of Information. In *Proceedings of ACM SIGMOD Conference on Management of Data, Washington*, DC, pp. 434–437, 1993.
12. J. Carbonell A machine learning perspective on information filtering. *ISMIS'00 Invited Talk*, October 11, 2000.
13. S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, and 6Jennifer Widom J. D. Ullman. The tsimmis project: Integration of heterogeneous information sources. In *Proceedings of the Fourteenth International Conference on Data Engineering, February 23-27, 1998, Orlando, Florida, USA*. IEEE Computer Society, 1998.
14. Yigal Arens and Craig A. Knoblock. Sims: Retrieving and integrating information from multiple sources. In Peter Buneman and Sushil Jajodia, editors, *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., May 26-28, 1993*, pages 562–563. ACM Press, 1993.
15. Michael R. Genesereth, Arthur M. Keller, and Oliver Duschka. Infomaster: An information integration system. In *SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data, May 13-15, 1997, Tucson, Arizona, USA*. ACM Press, 1997.
16. A. Y. Levy, A. Rajaraman, and J. J. Ordille. Query-answering algorithms for information agents. In *Proceedings of the AAAI-96, Portland, Oregon, August 4-8*, 1996
17. Marian H. Nodine, William Bohrer, and Anne H. H. Ngu. Semantic brokering over dynamic heterogeneous data sources in infosleuth. In *ICDE*, pages 358–365. IEEE Computer Society, 1999.
18. Katia P. Sycara and Anandeep Pannu. The retsina multiagnet system: Towards integrating planning, execution and information gathering. In *Agents*, pages 350-351, 1998.
19. V. Torra Weighted OWA Operators for Synthesis of Information In *Proc. 5th IEEE Inter. Conference on Fuzzy Systems*, New Orleans, USA, 1996, 966-971
20. K. Decker,K. Sycara and M. Williamson. Middle-Agents for the Internet. P *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, 1997
21. A.R. Feinstein and R.I. Horwitz. Problems in the "Evidence" of "Evidence-Based Medicine" Article of the *Excerptia Medica*, inc. 1997