An Ontology-based Architecture for Cooperative Information Agents

Frederico L. G. Freitas
Universidade Catolica de Santos
R. Carvalho de Mendon9a, 144, Vila Mathias
11.070-906 Santos, SP, Brazil
red@unisantos.brf

Guilherme Bittencourt
Departamento de Automacao e Sistemas
Universidade Federal de Santa Catarina
Cx. Postal 476, 88.040-900, Florianopolis, Brazil
gb@das.ufsc.br

Abstract

In the Web, extractor agents process classes of pages (like 'call for papers' pages, researchers' pages, etc), neglecting the relevant fact that some of them are interrelated forming clusters (e.g., science). We propose here an architecture for cognitive multi-agent systems to retrieve and classify pages from these clusters, based on data extraction. To enable cooperation, two design requirements are crucial: (a) a Web vision coupling a vision for contents (classes and attributes to be extracted) to a functional vision (the role of pages in information presentation); (b) explicit representation of agents' knowledge and abilities in the form of ontologies, both about the cluster's domain and agents' tasks. Employing this Web vision and agents' cooperation can accelerate the retrieval of useful pages. We got encouraging results with two agents for the page classes of scientific events and articles. A comparison of results to similar systems comes up with two requirements for such systems: functional categorization and a thoroughly detailed ontology of the cluster.

1 Introduction

Although the terms "Web information agents" and "cooperative information gathering" (CIG) are in fashion, there arc scarce examples of Web systems endowed with some sort of cooperation or integration among tasks related to text processing, *viz* classification, retrieval, extraction and summarization. When defining the main aspects of CIG [Oates et al 1994], information extraction was mentioned as a key technology that addresses information acquisition in complex environments, suggesting implicitly task integration for CIG.

Indeed, there is room for cooperation, in particular for extractor agents on the Web. These systems are being designed to process classes of pages with similar structuring and contents (e.g., call for papers, scientific articles, ads, etc). However, the relevant fact that many of these classes are interrelated forming *clusters* (e.g., Science) has been neglected so far. Cooperation among extractors could increase the retrieval of useful pages, taking advantage of these relations.

Aiming at task integration and cooperation, we designed an architecture for cognitive multi-agent systems that retrieve and classify pages from clusters of information in the Web, extracting slots of data from it. Two design requirements are crucial. One is a Web vision coupling a vision for contents (classes and attributes to be extracted) to a functional vision (the role of pages in information presentation, i.e., whether a page is a list, message, class instance or garbage). The other, necessary to enable cooperation, consists of explicit knowledge representation in the form of ontologies, both about the cluster's domain and agents' tasks.

Experiments held with two agents, that treated the classes of scientific events and articles, produced promising results, leading to two conclusions: (a) functional categorization can be profitable; (b) key requirements for success in Web CIG systems arc this categorization and a detailed ontology of the cluster.

The article is organized as follows: Section 2 describes the proposed Web vision. Section 3 introduces the architecture, its components and many types of reuse enabled. Section 4 outlines a case study: MASTER-Web (Multi-Agent System for Text Extraction, classification and Retrieval over the Web), applied to the Science cluster, with the two agents cited above. The results on contents and functional classifications are also shown. Section 5 compares MASTER-Web to similar systems. Section 6 brings future work and conclusions.

2 A Proposed Web Vision for CIG

Web extractors seek pages that are instances of pages' classes (e.g., call for papers pages, researchers', etc). These pages share many common attributes (slots), like date and place of a conference. Pages' classes outline a Web division *by contents.* The slots usually found in a class are *discriminant,* once they help distinguish class members. This fact supports the use of extraction in class categorization. Researchers' pages, for instance, usually contain typical data, such as projects, interest areas, etc.

Most links in classes' pages point to pages containing data from a few other classes. A set of interrelated classes and their relations gather a body of knowledge about a specific domain (e.g., science, tourism, etc). They form a *cluster* of classes. In researchers' pages, e.g., we often find links to papers, calls for papers and other classes from the scientific cluster.

Another view of the Web, based on the work of Pirolli et alii [1996], focus *on functionality,* dividing pages by the role played in linkage and information storage. We separate them into five functional groups: *content* pages (class members), resource directories *(lists* of content pages, e.g., lists of researchers or scientific articles), *messages* about content pages, *recommendations* (pages that arc members of other classes) and garbage.

Joining these two visions, we can accurately identify not only the information to be extracted from page classes, but also instances of relations among classes in a cluster. Thus, it can improve the search of useful pages.

3 Proposed Architecture for CIG

We propose an architecture for cognitive multi-agent system (MAS) to retrieve and extract data from Web pages belonging to classes of a cluster. The core idea of employing a MAS resides on taking advantage from the relations among classes through agents' cooperation. The architecture overview is illustrated in figure 1.

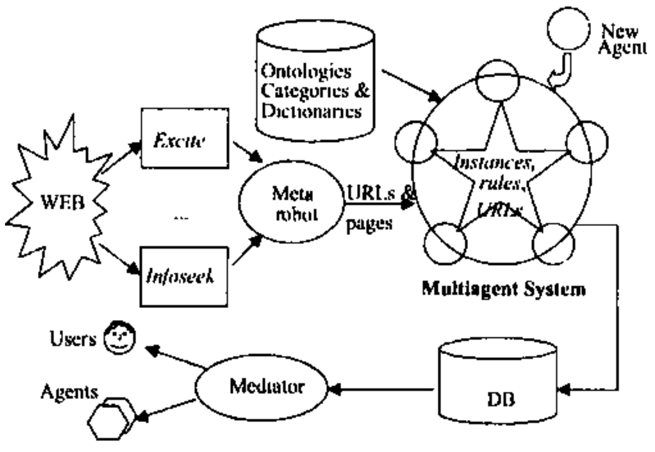


Figure 1: Architecture overview.

Each agent, represented as a circle in the figure, recognizes, filters and classifies pages, extracting attributes (slots) from them. The pages are supposed to belong to the class of pages that the agent processes. For instance, CFP pages are processed by the CFP agent, and papers pages by an articles' agent, in the Science cluster. The MAS is based on Distributed Problem Solving, where agents cooperate without overlapping functions.

Each agent relies on a meta-robot that can be connected to multiple search engines - like AltaVista, Excite, etc. The meta-robot queries the search engines with keywords that assure recall for the agent's page class (e.g., the terms 'call for papers' and 'call for participation' for the CFP agent). Due to the lack of precision, the resultant URL set from the queries is characterized by a wide variation of functional groups, containing many lists, messages, content pages from its class and from others agents' classes, and garbage. The retrieved URLs feeds a low priority queue of URLs to be processed.

An agent continuously accesses this queue and another one, assigned with high priority, which stores URLs sent as recommendations by other agents of the MAS or taken from pages considered as lists. These links are considered as "hot hints", because they were found under a safer context. Therefore, they arc expected to attain higher precision. Cooperation pays off if these suggestions contain less garbage than search engine results do.

An agent processes URLs by extracting data and classifying the pages functionally and by contents. The extracted data and the results arc kept in a database together. A mediator facilitates database access to users or agents, from the system or external, offering simpler reduced non-normalized database views.

When an agent joins the system, it announces itself and sends instances and rules to the other agents. This knowledge will be employed by them on the recognition of links or pages likely to belong to the page class processed by the sender agent. So, the agents update their knowledge bases and send to the new agent, in return, their own recognition instances and rules. When a link or page fires any other agent's recognition rule, the agent sends the link or page to that agent as a recommendation.

3.1 Agents' tasks

An agent performs the following steps for each URL. Figure 2 depicts an agent in detail.

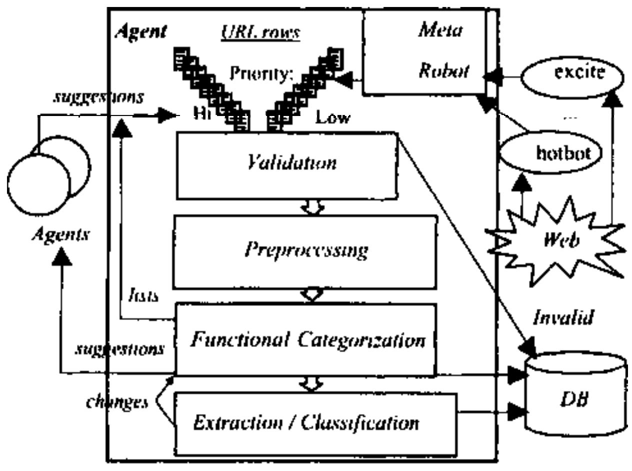


Figure 2: An agent and its processing steps.

Validation. Non-html, non-http, inaccessible and pages already stored in the database arc ruled out in this step.

Preprocessing. Representation elements - such as contents with and without html tags, centroid, title, links and e-mails, among other elements - are generated from each valid page applying IR techniques, like stop-lists, stemming and tagging. If necessary, shallow natural language representations can be used. The representation elements are passed to the agent's inference engine.

Functional classification. An agent deduces to what functional group the page fits to, whether a list, message, garbage or class member dealt by the agent or by another.

It is also here that links in a page arc suggested to other agents when any of their identification rules about the anchor or URL fires. For example, an anchor or URL with the word "conference" is useful for a CFP agent. Suggestions may trigger scanning on directory structure prefixes, like /staff/ for a researchers' agent. These directories usually keeps plenty of content pages.

Extraction and contents classification. These two tasks arc quite correlated; Riloff [1994] has already demonstrated the applicability of extraction for filtering. In our architecture, extraction assists classification, since each agent manages many subclasses, e.g., the CFP agent treats conferences, workshops and journals. After categorizing a page according to its semantic contents, the remaining slots, which weren't discovered yet, arc searched and extracted. In case contradictions or strange facts appear during extraction, the functional class assigned to the page can be modified. For instance, a supposed CFP page is retracted to be a list or garbage, if dates farther than a year are met in the top.

Using as many representation elements as required, each slot of data is extracted by a union of templates, cases and rules. When a slot is part of a predefined list, it is gleaned by matching terms from the dictionaries (e.g. checking the presence of acronyms to find US states).

3.2 Agents' knowledge

Ontologies play a central role in the architecture, serving not only as vocabulary in agent messages, but also defining proper concepts, slots, instances, restrictions and relations. The architecture encompasses five ontologies:

Cluster ontology. The main ontology ought to focus on the original domain of the cluster (e.g. Science), aiming at a rich and comprehensive representation of its classes, even if they don't appear in the cluster. There are some reasons for this. First, reuse possibilities by any application dealing with the domain are enhanced. Moreover, it provides a suitable structure to extract slots. To sum up, a detailed domain ontology assures performance in contents categorization (see section 5.1).

Web ontology. Classes here contain representations of the pages, chosen according to their adequacy in the required tasks. They comprise HTTP protocol data, Information Retrieval (IR) page representations (such as terms and frequencies, contents with and without tags, links, e-mails, etc), and Natural Language Processing (NLP) representations, suitable for unstructured classes.

CIG ontology. Operational knowledge classes (and instances) for the tasks of functional categorization, contents classification and extraction. It includes:

• Templates to extract slots (e.g. deadline of a CFP), to recognize functional categories (such as garbage, messages, recommendations, etc), and to classify content pages (e.g. conference or workshop CFP);
• Auxiliary classes like dictionaries concepts (with keywords and synonyms), agents and abilities, etc;
• Functions to determine regions, extract, convert, format, check consistency and dismiss extracted data;
• Complex and expressive cases specifying conditions, concepts and sets of slots whose presence or absence characterize a content page class. An example below:

```
([Articles-with-author-org-and-place] of Case
  (All-of-Concepts-Keywords TRUE)
  (Absent-Concepts-in-the-Beginning [thesis] )
  (Concepts-in-the-Beginning [abstract])
  (Slots-in-the-Beginning [Author-Name]
       [Organization-Name] [Location]))
```

This case detects articles. If, in the beginning of a page, an author name, an organization and a country (or US state) were extracted, terms similar to "abstract" were found, but not terms related to the concept "thesis" (like "partial fulfillment"), a rule bound to this case is fired. The attribute All-of-Concepts-Keywords determines whether all the slots have to appear in the page; if it was false only one would be necessary. Next, a Class-Recognizer instance (displayed below) and a rule creates an instance of the class associated with the case.

```
( [Part-Publication] of Class-Recognizer
  (Cases [Articles-with-author-org-and-place])
  (Class [Part-Publication]))
```

The agents first categorize pages into abstract classes (which can't have instances, as, in the example, Part-Publication), to assign later the exact class the pages fit to (e.g. Conference-Article). Cases, recognizers, concepts and rules are also applied on slot extraction and search of links and URLs to be suggested to other agents.

Auxiliary ontologies. Comprise ontologies for NLP (like WordNet [Miller 1995]), time and places (with countries, states, etc), and other specific ontologies from other subject areas, which can be helpful for an agent or domain (like bibliographic data, for the "articles" agent).

Indeed, a knowledge-based approach is adequate to permit not only high-level intentional cooperation, but also massive reuse of code, search engines' services, DB definitions - once agents' tables arc the same (pages not recognized, dictionaries, search engines and their queries), except the particular tablc where an agent stores its extracted data (for instance, CFPs to the CFP agent) - and knowledge. This latter is the most important reuse and can happen in several ways. Ontologies can be downloaded from repositories, like Ontolingua. A cluster ontology serves to all agents in a cluster, as well as most knowledge from dictionaries and rules. Thus, a new agent is quickly built, needing only new rules and instances of concepts, cases and templates, given that knowledge acquisition has already been accomplished.

4 Case Study: the Science domain

A model of agent was developed in Java, and a mediator, in Delphi and HTML. We utilized KQML for agents' communication. Ontologies were written using Protege[1]. Rules were specified in the inference engine Jess[2]. The search engines AltaVista, NorthernLight and Goggle were queried for retrieval. NLP wasn't implemented yet. The data to be extracted was identified to support the classifications, but not extracted

We reused an ontology of science from the project $(KA)^2$ [Benjamins et al, 1998], introducing many refinements. A key one was the creation of abstract classes grouping classes with common features, in order to improve classification. For example, class Scientific-Event derived abstract subclasses Live-Sc-Event - with concrete subclasses Conference and Workshop - and Sc-Publication-Event, with subclasses Journal and Magazine

Two agents from the Science cluster were developed:

CFP Agent. Handles any CFP pages. Extraction templates for 21 slots were defined (e.g. Place, Deadline, Editor-in-Chief) from 8 scientific events subclasses (the ones already cited plus Generic-Live-Sc-Event, Generic-Sc-Publication-Event and Special-Issues for Journal and Magazine), classified by 28 cases and templates.

Scientific articles agent. The search engines were queried with terms that appear often in articles: "abstract", "introduction", "related work", "conclusion", "discussion", and others. The agent is responsible for 10 classes: workshop, conference, journal and magazine articles, book chapters and generic articles, as well as thesis, dissertations, technical and project reports. 8 slot templates, 16 cases and 36 templates were created.

Three tests were executed with each agent. The first two treated *carpi* obtained from the search engines' queries; one was applied for knowledge acquisition, to specify the cases, templates and rules, and the other, for blind test. The third test was made directly from the Web.

4.1 Results

Table 1 displays tests' results. In the table, recognition stands for an evaluation if the agent correctly identified member pages. Note that a fourth test was run with CFP agent, in which it benefited from lists of content pages. Lists of CFPs on a focused subject are usually maintained by researchers, departments and organizations.

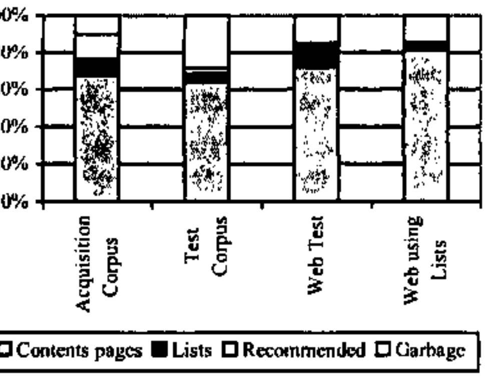
CFP Agent. More than 70% of content pages were conference CFPs, mostly recognized by the presence, in the top of the pages, of the concept CFP, and the slots initial date and location The few false content pages were very long and about communities of users (Linux, XML, etc), containing many of the slot triggering concepts. The agent failed to recognize CFPs with little data or not using the jargon ("deadline", etc). Lists were detected when many anchors contained terms related to events, or when lots of time intervals in a page (e.g., 1-4 December) were present. Special care should be taken to lists, once false positives can queue bad links as "hot hints".

	"Call for Papers" Agent				Articles Agent		
	Acquisition Corpus	Test Corpus	Web Test	Web using Lists	Acquisition Corpus	Test Corpus	Web Test
Recognition	97,1	93,9	96,1	96,3	93,1	82,7	87,8
Functional Categ.	93,8	93,9	93,8	95,7	96,8	94,0	95,1
Contents Classification	94,9	93,3	92,9	91,7	97,0	93,0	81,4
Nr. of Pages Processed	244	147	129	188	190	150	184

Tabic 1: Agents' performances.

The lists increased substantially the retrieval of content pages. The set of pages retrieved in the fourth test was clearly more focused. Pages from other categories, like lists, messages and recommendation to the fictitious Organizations agent and Dividable-Publication agent (process Proceedings), were substituted by content pages in the form of frames. In the other tests, only one frame appeared. In this test, the number of contents pages rose between 13 and 22% (see Figure 3), due to frames pages.

Figure 3: Functional categories distribution in the CFP agent



Even the garbage patterns changed: instead of false content pages, caused by search engines' lack of precision, garbage were graphical, initial pages of CFPs. It is also remarkable that some scientific events found in lists were not discovered in the other tests. These facts demonstrate that functional categorization, and, in particular, the use of lists, pays off.

Scientific articles agent. Recognition errors happened in articles with few data, data in the end, or with affiliations in unknown companies. An article must have at least three of the slots Organization-Name

http://protcge.stanford.edu

[2] http://herzberg.ca.sandia.gov/Jess

Author-Name, Location, e-mail and Department. The agent missed articles containing only the authors' names, but this is normal in other similar systems like CiteSeer.

Contents classification was based on information about articles' publication, usually put in the top of papers. More than half of the articles didn't bring this data.

Cooperation. We executed one more test, in which the agents cooperated. The CFP agent asked the articles' agent for anchors in top of papers containing concepts like "conference", "journal", etc. The articles' agent transformed these requests into rules to suggest anchors to the CFP agent. Although it worked and no suggestion was wrong, only three anchors were sent.

To prove the usefulness of cooperation, the CFP agent scanned anchors in Program Committees seeking for suggestions to a future "researchers" agent. No extraction technique (e.g. proper names) or dictionary was present, only simple heuristics. The agent recommended 30 correct and 7 wrong anchors, a promising rate, given that researchers' pages are less structured and, therefore, harder to be retrieved by search engines queries.

5 Related Work and Discussion

Many subject areas are involved with this project: information, retrieval, extraction, classification, multi-agents, information agents, NLP and ontologies, at least. Instead of trying to review these areas, we will focus on comparing and discussing solutions similar to ours.

5.1 WebKB

This system [Craven et al, 1999] integrates extraction and classification as separate tasks, supported by machine learning. Pages from computer science departments' sites in universities are classified against a domain ontology previously defined, with classes and relations. The pages were represented using IR techniques. Classification and extraction rules were learnt from annotated *corpi.*

The decision of relying on machine learning depends upon some issues. The first one is a comparison between the costs of annotating *corpi* against inspecting them for knowledge acquisition, as we did. ML brings gains such as speed and adaptability. However, there are drawbacks, like readability and ontological engagement of the learned rules (which tends to be quite specific), difficulties to include a priori knowledge and to capture some rules without adding lots of features, and even to generalize through a huge number of features or classes.

WebKB's ontology was composed of four basic classes: activities - with subclasses projects and courses - persons - student, faculty and *staff* -, departments and other. Relations among classes were defined, as in our project: project members, advisors, instructors, etc.

Discussion. The authors evaluated contents' classifications only by the false positives, reporting rates between 73 and 83 %, except for the classes "staff member" and "other". Nonetheless, if false negatives were computed, the class "other" reaches 93,6%, "student" 43% and the remaining 6 classes less than 27%, lowering the average rate to around 50%.

Other drawbacks are the huge incidence in the class "other", almost three times the sum of the other classes together, 65% of them wrong. This class severely affected performance, since 90% of the mistakes of the tests came from it. Anyway, even if the classification was correct, the utility and costs of such system would be questionable, since more than 73% of the pages would have to be categorized as "other".

This deluge of garbage pages seems to stem from two facts: (a) The absence of functional categorization, and (b) the role of ontologies was underestimated in WebKB, since its ontology doesn't seem to be comprehensive enough. On the contrary, MASTER-Wcb's ontology incorporates classes not useful yet, such as projects and products, because agents to process these classes can be available in the future. Moreover, current agents apply these definitions to manage their own classes.

On the other hand, the learning process may face problems on generalization with an ontology with many classes. Much more annotated pages would probably be needed as well. Therefore, any solution shall rely on cooperation, with learning or inference agents.

5.2 CiteSeer and DEADLINER

These systems perform efficient retrieval, filtering and extraction over the Web, exploiting statistical and learning methods joined with a priori knowledge.

CiteSeer. One of the most accessed software in the search of scientific articles [Bollacker et al, 1999]. They are found monitoring mailing lists, newsgroups and publishers, or querying search engines with keywords "papers", "publications" and "postscript". An article is recognized by the existence of a reference section.

The system extracts bibliographic data from each article and from its reference section, which plays the role of the functional category of list, helping to locate other articles. The number of citations of an article by others represents a measure of its relevance. Databases of authors and journals, as well as complex techniques, are applied to identify co-references for authors and articles.

DEADLINER. This system [Kruger et al 2000] searches the Web for scientific events. It scans newsgroups and broadcast e-mails, and has its own meta-robot. From each CFP, it extracts initial and final date, deadline, program committee, and affiliation of each committee member, themes, events' name and country. Extraction is carried out by a sequence of positional filters integrated using automatic learning. Extraction performance achieves more than 70% for all the slots. For program committee extraction, the system benefits and updates CiteSeer's researchers' database.

Recognition performance is similar to ours (more than 95%), but its definition of event is more restrictive: all the slots are required, except country, plus some submission information. MASTER-Web's CFP agent offers more recall and flexibility, accepting announcements of book chapters, magazines, contests, workshops and journals. Moreover, the requirements are expressed in cases, which arc much more flexible. CFP agent was designed bearing in mind that users are interested in any CFPs for their research works.

Discussion. Although developed by the same research group, CitcSccr and DEADLINER nowadays are not able to cooperate directly. They handle anchors and pages that interest each other, only cooperating in the construction of the researchers' database.

This is a common problem for extractor agents on the Web: the simplicity, adaptability and speed of wrapping, learning and finite-state methods prevent these systems to cooperate in an intentional knowledge-based manner. Thus, they couldn't ask for data of a certain type, employing, as shared vocabulary for communication, ontologies about the domain, their abilities, desired pages, features among other pieces of knowledge.

As the number of Web extractors grows, each with its robot collector, strain on bandwidth can be. Conversely, cooperation could turn out to be even more profitable than search engines results, constituting the only starting way to look for unstructured classes, like researchers' pages. Furthermore, cooperation can take place also at the *level of data:* the CFP agent can warn a researchers' agent that a certain researcher was chair of an event, a fact that could be missing in her homepage.

This stems from the knowledge representation. In statistical and finite-state approaches, knowledge is hidden behind algorithms. It can be neither reasoned nor communicated at knowledge level, blocking cooperation among extractors and integrated extraction. On the other hand, declarative systems can change their behavior dynamically in favor of the cluster's processing, communicating in an elegant manner. Another advantage of these systems resides on the fact that, with an approach like CiteSeer and DEADLINER to process a new class, most of the new system has to be made from scratch. With our architecture, reuse is massive, and only part of the knowledge has to be acquired and specified.

6 Future Work and Conclusions

This project has a long road ahead. New agents for the Science cluster have to be deployed, like a researchers' agent, to make cooperation actually effective. We must extend our tests to other clusters formed by interrelated page classes, such as a tourism cluster - linking events, hotels, and transport pages.

Learning and NLP techniques must be accomodated to deal with less structured page classes. The agents should also check data duplicity, and documents written in other formats (like .ps and XML) must be covered too.

With this work, we tried to fertilize the field of CIG with some new issues and principles, with cooperation as the guiding idea: task integration, declarative inferential extractors capable of intentional cooperation, a Web vision coupling functional and contents aspects, mapping of semantic relations among page classes and identification of linkage patterns. A key requirement for CIG systems came up with the tests: a detailed ontology is needed to process the richness of a cluster's domain.

On the practical side, we designed a fully reusable cognitive MAS to process data from whole regions of the Web, and got encouraging results from it. We claim that keyword-based search engines can be a basis for more accurate ontology-based domain-restricted cooperative information agents in a near future.

Acknowledgments

The authors thank Tercio Sampaio and Rafael Teske who helped with implementation, and the referees.

References

[Benjamins et al, 1998] R. Benjamins, D. Fensel and A. G. Perez. *Knowledge Management through Ontologies.* Proc. of the 2nd International Conf. on Practical Aspects of Knowledge Management, Basel, Switzerland. 1998.

[Bollacker et al, 1998] K. Bollacker, S. Lawrence and C. L. Giles. *CiteSeer: An Autonomous Web Agent for Automatic Retrieval and Identification of Interesting Publications.* Proceedings of the 2nd International ACM Conference on Autonomous Agents, USA. 1998.

[Craven et al, 1999] M. Craven, A. McCallum, D. DiPasquo, T. Mitchell, D. Freitag, K. Nigam and S. Slattery. *Learning to Extract Symbolic Knowledge from the World Wide Web.* Technical Report CMU-CS-98-122. School of Computer Science. CMU, USA. 1999.

[Kruger et al, 2000] A. Kruger, C. L. Giles, F. Coetze, E. Glover, G. Flake, S. Lawrence and C. Omlin. *DEADLINER: Building a new niche search engine.* Conference on Information and Knowledge Management, Washington DC, November, 2000.

[Miller, 1995] George Miller. *WordNet: A Lexical Database for English.* Communications of the ACM. 38(11):39-41. USA. 1995.

[Oates et al, 1994] T. Oatcs, M. Prasad and V. Lesser, *Cooperative Information Gathering: A Distributed Problem Solving Approach.* Computer Science Technical Report 94-66-v2. Univ. of Mass, Amherst, USA. 1994.

[Pirolli et al, 96] P. Pirolli, J. Pitkow and R. Rao, *Silk from a Sow's Ear: Extracting Usable Structures from Web.* http://www.acm.org/sigchi/chi96/proceedings/papers/Pirolli_2/pp2.html. 1996.

[Riloff, 1994] Ellen Riloff. *Information Extraction as a Basis for Portable Text Classification Syterns.* PhD. thesis. Department of Computer Science. University of Massachusetts at Amherst. USA. 1994.