Accessing the Web: Exploiting the Data Base Paradigm

Tiziana Catarci, Luca Iocchi, Daniele Nardi, Giuseppe Santucci
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italy
{catarci,iocchi,nardi,santucci}@dis.uniroma1.it

Abstract

The open growth of the Internet, the amount of available information, and the typical access modality (i.e., browsing) cause the puzzled user to search for the information of interest in a labyrinth of links.Web-at-a-Glance (WAG) is a system aiming to allow the user to query (instead of browsing) the Web. The basic idea is to build, once the user has specified a generic domain of interest, the domain conceptual representation, to instantiate it with data extracted from Web sites (so to build a materialized view over the Web), and to query such a conceptual representation through an easy-to-use visual interface. Knowledge representation techniques are used for both the internal modeling of the conceptual representation and for supporting the automatic extraction of data from Web sites to feed the materialized view.

1 Introduction

The WWW is undoubtly a great source of disparate information. This implies that information surfers could have a lot of fun while browsing diverse sites, finding a rainbow of unexpected data. However, what about information seekers, i.e., those who are searching for specific information? They have a real hard time in finding the data of their interest, following the billions of links of the Internet labyrinth and/or checking the thousands of references returned by search engines [1]. They would probably prefer just "querying" the Web, specifying what they want and do not care about the paths to be followed for reaching the information of interest.

Web-At-a-Glance (WAG) [2] is a system thought for such users, since it enables them to query (instead of browsing) the Web. WAG performs this ambitious task by first constructing personalized databases, containing the information pertinent to the user's interests which are relevant for specific domains. The system semi-automatically gleans such information from a Web site or several Web sites, stores it into the databases, cooperatively designed with the user, and allows her/him to query such databases through a visual interface equipped with a powerful multimedia query language.

In order to achieve its aims, WAG exploits a suitable integration of ideas and techniques coming from both the database and the knowledge representation areas and it is based on three key issues:

1. the building of a conceptual representation for any domain of interest, which is populated with data extracted from the Web, so to constitute a fully materialized view;

2. the availability of a visual interface to easily query such a materialized view [3];

3. the adoption of both an internal modeling of the information and several acquisition mechanisms based on Description Logics [4].

There are other approaches (see Section 2) sharing similar objectives with WAG. However, WAG differs from all of them mainly because, instead of requiring an explicit description of the sources, it attempts to semi-automatically classify the information gathered from various sites based on the conceptual model of the domain of interest.

The paper is organized as follows. First, we present a classification and a comparison of diverse systems whose common goal is to access information residing on the WWW. Then, we describe the architecture and the main modules of the WAG system. We conclude by briefly commenting on some preliminary experiments done with the prototype.

2 Web Information Access Systems

The huge amount and heterogeneity of Web data makes it extremely difficult to organize them under a coherent and uniform structure. As a consequence, it is more difficult than usual (i.e., when accessing archives, databases, etc.) to locate the information of interest. Actually, there are some attempts towards establishing a general agreement among the information sources on the WWW, but they appear to be quite far from the fulfillment. More promising seems to be the development of a variety of tools aiming at helping the user in her/his information search.

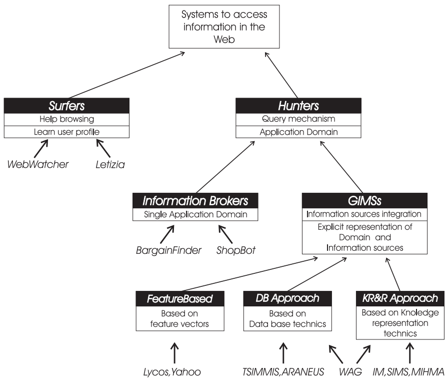

Figure 1: *Classification of current approaches*

Such tools can be classified as in Figure 1 which shows the various classes, how they are hierarchically organized, and, for each class, its discriminant characteristics. First of all, systems are divided into so-called surfers and hunters. The former class comprises browsing systems which facilitate the user during the navigation, based on the user model they incrementally build (e.g., [5, 6]). Such systems are often based on ad-hoc mechanisms, depending on the particular application domain, and do not offer a significant help in locating the information, instead they speed the analysis process once a set of sources or html links has been found. Hunters provide the user with different kinds of query facilities to access the Web data. They may be further refined into two subclasses: information brokers and global information management systems (GIMSs). Information brokers (e.g., [7, 8]) are devoted to the semi-automatic extraction of information for specific predefined application domains. GIMSs try to generalize this approach by removing the constraint on the single application domain. Thus, their ultimate goal is to treat disparate heterogeneous information sources inside a single global structure, allowing the user to just ask queries in order to retrieve the information of interest. Popular keyword-based search engines can be considered as GIMSs, where documents are characterized using feature-based representations. Such representations make it easy to automatically classify documents, but offer very limited capabilities for retrieving the information of interest. More advanced GIMSs typically use sophisticated methods for representing information sources. Such methods can be roughly classified as being based on database or knowledge representation techniques (e.g. [9, 10] and [11, 12, 13] respectively). In the database perspective the goal is to build a fully materialized data warehouse of the information in the Web. Conversely, knowledge representation based methods provide a solution in which specific information is recorded locally, while the ability to answer queries relies on methods for dynamically accessing the information sources. A fully materialized approach carries a number of advantages in the ease and effectiveness of access once the data are stored in the database. However, it raises a number of issues, in particular the problem of database construction and maintenance. Typically, a conceptual schema of the information domain is built and specific procedures are implemented for extracting the information to be stored based on the conceptual schema. However, either there are strong assumptions on the structure and organization of the information sources or the issue is left to (user-made) ad hoc solutions. The dynamics of the information sources raises a maintenance problem for the materialized data. A fully virtual approach is better suited to cope with the dynamics of the information sources, while it is generally problematic with respect to the response time. In a knowledge based approach the idea is that the system handles an *explicit* representation of the information sources, which is used at query time.

In the rest of this section we concentrate on the last two categories of systems, namely GIMs based on either database or knowledge representation techniques.

2.1 Database approaches

In the database-oriented approaches the basic idea is to regard the WEB as a federation of databases. With the difference that database federations typically rely on the presence of a schema describing the sources and on highly structured data, while Web documents are usually unstructured or semi-structured.

One example of this first approach is Tsimmis [9], which describes the common schema with the OEM (Object Exchange Model) language. The associated query language, OEM-QL, is an SQL-like language. Tsimmis makes use of *translators* to translate both data objects into a common information model and queries into requests for an information source, while *mediators* embed the knowledge needed for processing a specific type of information, once the content of information sources is known. Each mediator needs to know which sources it will use to retrieve information. Therefore, a model of information sources has to be explicitly specified, but it is possible to work without a global database schema. Classifiers and extractors can be used to extract information from unstructured documents (e.g. plain text files, mail messages, etc.) and classify them in terms of the domain model. The classifier/extractor components of Tsimmis is based on the Rufus system [14]. Rufus uses an object oriented database to store descriptive information about user's data, and a full text retrieval database to provide access to the textual content of data.

Another proposal along these lines is constituted by the ARANEUS Project [10], whose aim is to make explicit the schema according to which the data are organized in so-called structured servers, and then use this schema to pose queries in a high level language instead of browsing the data. Even though the ability to construct structured description of the information in the Web enables the system to answer effectively user queries, the approach has the following drawbacks that are typical of a Database perspective: 1) Araneus works only on a particular kind of Web sites and pages, which have a clearly specified structure, not on generic ones; 2) the user has to completely specify the relational schema corresponding to the site data; there is no automatic translation from the site to the database; 3) there is no hint for automatic search and conceptualization of WWW sites similar to prototypical ones indicated by the user.

Another worth mentioning research line involves the development of declarative languages to query the Web [15, 16, 17]. Note that this approach is weakly related with the idea of modeling the information stored in the Web sites. Indeed, the main idea is to model the Web document network topology, and to provide the user with a query language enriched with primitives for specifying query conditions on both the structure of single documents and their locality on the network. However, the user cannot query the Web information content.

2.2 Knowledge-based approaches

Knowledge-based GIMSs are systems using a Knowledge Representation approach for information source representation, data acquisition and query processing. Many logical frameworks are used to represent information and reason about them.

The main design element for these systems is the Knowledge Representation language. Also relevant are automatic data acquisition techniques, that are useful to build and update knowledge bases, as well as query-planning techniques, adopted to answer user queries.

As for the Knowledge Representation language and data acquisition aspects, let us remark that a GIMS needs to represent both the application domain and the content of the information sources. Usually a single Knowledge Representation language is adopted. One typical example is constituted by Description Logic, which is suited to represent taxonomic knowledge.

In addition, a basic feature for a GIMS is the possibility of identifying interesting information sources unknown to the user and to automatically gather from them relevant information units. In other words, tools to scale up with the growth of the information space are needed.

The discovery of new information sources, the extraction of information units within them and the interpretation of data coming from these sources are all problems related to information acquisition. This issue is rarely addressed in most systems, as they force the user to hand-code information source models. The main exceptions are ShopBot and ILA [7]. ShopBot addresses the extraction problem learning how to access an on-line catalog (via an HTML form) and how to extract information about products. It uses an unsupervised learning algorithm with a small training set. Whereas ILA (Internet Learning Agent) is focused on the interpretation problem. It learns how to translate information source output into the domain model, using a set of descriptions of objects in the world.

It is worth noting that, especially when dealing with the automatic discovery and integration of information sources, the vocabulary problem is one of the most critical ones. The presence of possibly different terms representing the same concept in the description of a source or an information unit is a significant example. At least three possibilities have been explored to face this problem: 1) unique vocabulary, that is forcing the description of information sources and domain model to share the same vocabulary; 2) a manual mapping, that is relationships between similar concepts are hand-coded; 3) automatic (or semi-automatic) mapping, in which the system takes advantage of existing ontologies that provide synonym, hypernym and hyponym relationships between terms. The use of hypernym and hyponym relationships is a powerful tool to solve questions about the terminology, but involves loss of information when generalizations of terms are used.

As for query answering, a significant body of work on agents able to reason and make plans has been developed. In this case, the representation of the information sources is known to the system. The use of planning techniques to retrieve information requested by a user query has been very common in this context and is in general aimed at introducing a certain degree of flexibility in exploring the information sources and extracting information from them.

For instance, in Information Manifold [18] the content of information sources is described by query expressions that are used to determine precisely which sources are needed to answer the query. The planning algorithm first computes information sources relevant to each subgoal, next conjunctive plans are constructed so that the soundness and completeness of information retrieval and the minimization of the number of information sources to be accessed are guaranteed. In this system, interleaving planning and execution is a useful way to obtain information for reducing the cost of the query during plan execution.

SIMS [11] defines operators for query reformulation and uses them to select relevant sources and to integrate available information to satisfy the query. Since source selection is integrated into the planning system, SIMS can use information about resource availability and access costs to minimize the overall cost of a query.

A final note is on the closed world assumption adopted by all the above systems. That is, they work on the assumption that the domain model contains all information needed and that all unavailable information does not exist. On the contrary Internet Softbot [13] provides a framework to reason with incomplete information, executing sensing actions to provide forms of local closure, i.e., to verify the actual presence of information in the source during plan execution.

3 Web-At-a-Glance (WAG)

The work briefly surveyed in the previous sections shows the efforts that have been separately made by both the database and the knowledge representation communities to find effective ways to model the information contained in the Web. However, we argue that many of the still existing open problems could be solved by combining database and artificial intelligence techniques. This is the idea we are following in the WAG (Web-At-a-Glance) project [19, 2], by coupling a database conceptual model (namely the Graph Model [20, 21]) and its environment to interact with the user [3] with the Classic knowledge representation system [4], in a system aiming to semi-automatically build conceptual views over information extracted from various Web sites and to allow the user to query such views.

The main differences with other Database approaches (e.g. Tsimmis and ARANEUS) are the following.

1. Instead of requiring an explicit description of the sources, WAG attempts to semi-automatically classify the information gathered from various sites based on the conceptual model of the domain of interest.

2. The result of such a classification is fully materialized. The idea is that a certain degree of inaccuracy in the classification is acceptable in "unknown" domains. In order to make feasible the site conceptualization, a set of tools are available to match the information acquired visiting new sites with the domain model owned by the system.

3. WAG provides a visual interface to query the databases (each one related with a specific domain or sub-domain) resulting from the integration of the information extracted from the various sites.

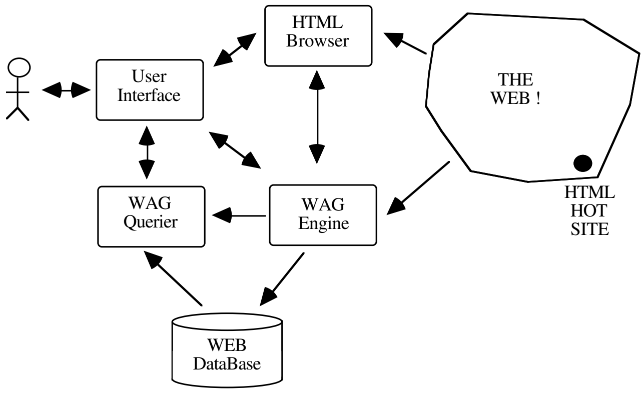

Figure 2: *The System Architecture*

In this section we present the architecture of WAG. In Fig. 2 the main components of the system are shown. WAG has a highly modular architecture, in which several components cooperate to accomplish the task. The user interacts with the user interface that allows for switching among a conventional HTML browser, a WAG Querier, and the WAG Engine. Each time the user meets a site containing pieces of information about a relevant matter s/he can activate the WAG engine in order to analyze it.

The WAG Engine reaches the site pointed out by the user and collects the HTML pages belonging to it. Once the site pages are locally available, the WAG Engine starts its analysis. In doing that, some additional information on the domain of interest is needed; it is provided either by the system knowledge base or by the user, through an interactive session. In the latter case, the pieces of information gathered by the user are added to the knowledge base for further reuse. The main objective of the analysis process is to associate with the site under observation a conceptual data schema and to populate it. The results of such a process, that may again involve the user in an interactive fashion, are stored in the WEB DataBase. More precisely, the WEB DataBase contains both the data and the locations in which such data are available (e.g., the page URL, the page paragraph, etc.).

Once the site has been fully analyzed, the user is provided with a new powerful way to access the information stored in the site itself: s/he can query the WEB through the WAG Querier, according to several query modalities provided by the system. The WAG Querier handles all the phases needed to answer a user query: query formulation, query distribution, and query result presentation. In particular, it provides the user with a multiparadigmatic visual environment (see [3]), equipped with different visualizations and interaction mechanisms, including a synchronized browser on the database schema and instances and several ad-hoc features for interacting with multimedia data.

Below we focus on the two main submodules of the WAG engine: The *Page Classifier* and the *Conceptualizer*.

3.1 Page Classifier

Pirolli et al. ([22]) present a classification technique of WWW pages, which is used to identify and rank particular kinds of them, such as index pages and organization home pages. We build upon their work in order to come up with a particular page categorization which provides useful information to the Conceptualizer module. We define four page categories:

- organizational home page: these pages represent the entry point for different kinds of organizations and institutions;
- index: these pages contain a large number of links (with respect to the page size) to navigate towards other (usually related) pages;
- personal home page: these pages belong to individuals, who may or may not be affiliated with some organization;
- document: these pages have the purpose of delivering specific information and, consequently, the percentage of outgoing links vs. the total page size is very low.

Once encountered by the classifier, a page is analyzed in order to categorize it, and figure out some basic characteristics. There are two different kinds of analysis: the first one checks the syntactical structure of the page, in order to verify the presence of HTML keywords which signal specific objects, i.e., lists, nested lists, forms, tables, and applets; the second one calculates the probability of the page to belong to each of the above four categories. The result of the classification phase is a feature vector associated with the page. The vector contains several page characteristics, useful for the Conceptualizer activities. In particular we store in the feature vector quantitative pieces of information about the page (size, number of incoming links, number of forms contained in the page, etc.) and several probabilistic figures (e.g., the probability for the page to be an index page). Quantitative pieces of information are directly collected from the analysis of the HTML source; the probability figures are computed using statistical techniques, based on a set of relevant properties of the page. The properties we take into account are: page size, number of local (i.e., coming from the same site) incoming links, number of outgoing links; frequency of access, which indicates how often the page has been visited, and depth of the children nodes reachable by that page. We collected sample pages on the Web through a Java robot and we analyzed about six thousand pages (pages were also analyzed by hand to determine the success or failure of the automatic classification). We obtained satisfactory results when classifying index and document pages (up to 0.967 success percentage), while the statistical approach failed when considering home pages. This is presumably due to the fact that home pages are designed in a very non uniform way and that, consequently, the distribution of their features is very close to the overall page distribution. Consequently, we adopted suitable heuristics for the classification of the home pages. Looking at the presence of a pair $\langle name, surname \rangle$ in the text (plus the optional words "home page") we got up to 0.97 success considering personal home pages. In the case of organization home pages, analyzing the graph structure of the pages (see also [23]) and looking at the deeper paths and at the URL structure we got a success percentage up to 0.92. Using a mixed approach (heuristics for home pages and probabilities for document and index pages) we have now a working prototype able to classify pages with an overall success percentage of 0.96.

3.2 The Conceptualizer

The Conceptualizer is the core of the system. It builds a conceptual schema from the HTML pages of a certain site, and then populates the schema with different kinds of instances (e.g., URL, tuples, multimedia objects, etc.) extracted from the site.

The Conceptualizer relies on two formal models, which are strictly related. The first one is an object-based data model, the Graph Model. It has two important features: 1) it is semantically rich; 2) it is equipped with a relationally-complete set of graphical query primitives, which have been already used as formal basis for a multiparadigmatic visual interface [3]. A Graph Model DataBase (GMDB) is a triple $< g, c, m >$, constituted by: 1) an intensional part, comprising the schema of the database, the so-called *Typed Graph* g, and a set of *constraints* c, which includes the standard ISA and cardinality constraints, and 2) an extensional part, i.e., the instances of the database, represented by m, which is called *Interpretation*.

As for the second model, the main need is to have schemata modeling the information of the domains of interest, plus knowledge representation mechanisms and reasoning services to support the construction and maintenance of such schemata. In WAG, we choose to express the information content of the various domains in a knowledge representation formalism of the family of Description Logics. The formalism is equipped with special reasoning capabilities (for example to detect containment relations between classes, or to classify new classes with respect to a set of existing ones) and has a strict relationship with semantic data models [24, 25].

In particular, we have chosen to represent the system generic ontology using the knowledge representation system CLASSIC [4], while the user's views on the various domains are represented using the Graph Model structures, namely as pairs $< g, c >$. The idea is to use a restricted representation to reason efficiently, while adopting a richer framework for modeling the data, thus providing the user with a suitable model for the interaction with the system.

While analyzing and structuring the site information, the Conceptualizer executes three main activities (for a detailed description of the techniques used while executing each activity see [26]):

Site Structure Discovery. For each site, the system, starting from the Page Classifier inputs, analyzes the page graph in order to cluster homogeneous sets of pages and links that may be grouped into classes and relationships. During this phase various techniques are utilized, e.g. text analysis, link analysis, pattern matching;

Schema Definition. The site conceptual schema is matched against the description of the domain knowledge (which is either already part of the system ontology or is built up with the help of user's suggestions) in order to build a complete conceptual schema;

Database Population. Once the conceptual schema resulting from the previous phases has been completely built, the WAG engine visits the site and extracts from the HTML pages data to materialize, which constitute the instance level of the conceptual schema [1]. In doing this, both text analysis techniques and domain specific plus general knowledge are exploited.

3.3 Working Prototype

The current version of the WAG prototype embodies an implementation of the interface (including the site analysis, querying and schema editing functionalities), the Page Classifier[2], and the Conceptualizer. Presently, the latter fully supports the phase of Database Population, while the phase of Schema Definition is currently based on an interactive approach to acquire from the user information about site and page structures; the phase of Site Structure Discovery is still under development. Thus, the user is asked to provide the system with the information that cannot yet be automatically derived.

The present prototype has been tested on a specific domain concerning typical teaching and research activities carried out in universities (the domain contains the concepts of department, professor, course, lab, research area, etc.). WAG has been activated to search and classify information contained in Web sites related with the University domain (at this stage of the experiment we restricted the search to italian sites (i.e., `.it`). Once the database construction has been completed we asked a set of queries about teachers, courses and labs of computer science departments. Then, in order to compare the performances of WAG with those of well known search engines (SEs), namely Lycos, Yahoo! and Metacrawler, we submitted the same queries to SEs and manually analyzed the first 20 top ranked retrieved documents seeking for information automatically found by our system.

We got encouraging results, showing that a) in the domain under consideration the system can acquire automatically a substantial percentage of `relevant` data (While SEs return an enormous amount of insignificant links), and b) the answers of WAG on queries concerning specific information (e.g. the phone number of a professor) are satisfactory, whereas SEs typically fail.

[1] Actually, data are stored into a relational database, straightforwardly derived from the conceptual schema

[2] http://ftp.dis.uniroma1.it/pub/santucci/WAG/ CLASSIFIER/index.html

4 Conclusions

It is our opinion that, in order to exploit the enormous amount of disparate information contained on the Web, the user has not to desperately search for it, instead s/he should just ask a query and get back the desired answer. To obtain this ambitious goal is not an easy task. First of all, there is the need of effective ways to retrieve, extract and model the information of interest.

Our proposal, namely the WAG system, aims to address these issues by a suitable integration of ideas and techniques coming from both the database and the knowledge representation areas. Indeed, the basic idea of WAG is to allow the user to access the Web data by simply issuing a visual query on the conceptual schema of a database, and to rely on sophisticated knowledge representation techniques in order to build such a database.

The results of a first bunch of experiments on real Web data supports the goodness of such an idea.

Moreover, even if designed to cope with the WWWW (Whole WWW), the Wag system can be usefully exploited in a Corporate Intranet as well. During an initial phase, the WAG classification techniques can be used to gather and structure the pieces of information available in the organization. Afterwards, the system can be tuned to be more efficient in the isolate world of an organization where, under the reasonable assumption that a relaxed standard about the production of HTML pages is available, WEB documents show a more stable and predictable structure.

References

[1] T. Catarci. Interacting with databases in the global information infrastructure. *IEEE Communications Magazine*, 35(5):72–76, 1997.

[2] M. De Rosa, T. Catarci, L.Iocchi, D. Nardi, and G. Santucci. Materializing the web. In *Proc. of COOPIS-98*, 1998. to appear.

[3] T. Catarci, S. K. Chang, M. F. Costabile, S. Levialdi, and G. Santucci. A graph-based framework for multiparadigmatic visual access to databases. *IEEE Transactions on Software Engineering*, 8(3):455–475, 1996.

[4] Alexander Borgida, Ronald J. Brachman, Deborah L. McGuinness, and Lori Alperin Resnick. CLASSIC: A structural data model for objects. In *Proc. ACM-SIGMOD Conference on the Management of Data*, pages 59–67, 1989.

[5] Robert Armstrong, Freitag Dayne, Thorsten Joachims, and Tom Mitchell. WebWatcher: A learning apprentice for the World Wide Web. In *Proc. of AAAI Spring Symposium on Information Gathering form Heterogeneous Distributed Environments*, 1995.

[6] Henry Lieberman. Letizia: an Agent that assist Web Browsing. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, 1995.

[7] Mike Perkowitz, Robert B. Doorebons, Oren Etzioni, and Daniel S. Weld. Learning to understand information on the Internet: an example-based approach. *Journal of Intelligent Information Systems*, 1996.

[8] Bruce Krulwich. BargainFinder. `http://bf.cstar.ac.com/bf/`, 1996.

[9] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. The TSIMMIS Project: Integration of Heterogeneous Information Sources. In *Proc. of IPSJ Conference*, pages 7–18, 1994.

[10] P. Atzeni, G. Mecca, and P. Merialdo. Design and maintenance of data-intensive web sites. Technical Report 25, Dipartimento di Informatica e Automazione, Università di Roma Tre, 1997.

[11] Y. Arens, C. A. Knoblock, and W. Shen. Query reformulation for dynamic information integration. *Journal of Intelligent Information Systems*, 1996.

[12] T. Kirk, A. Y. Levy, Y. Sagiv, and D. Srivastava. The Information Manifold. In *Proc. of the AAAI Spring Symposium on Information Gathering in Distributed Heterogeneous Environments*, 1995.

[13] Oren Etzioni and Daniel Weld. A Softbot-Based Interface to the Internet. *CACM*, 37(7), 1994.

[14] K. Shoens, A. Luniewski, P. Schwarz, J. Stamos, and J. Thomas. The Rufus system: Information organization for semi-structured data. In *Proceedings of the Nineteenth International Conference on Very Large Data Bases (VLDB-93)*, 1993.

[15] D. Konopnicki and O. Shmueli. W3QS Home Page. `http://www.cs.technion.ac.il/~konop/w3qs.html`.

[16] L. Lakshmanan, F. Sadri, and I. N. Subramanian. WebLog Project. `http://www.cs.concordia.ca/~special/bibdb/weblog.html`.

[17] A. Mendelzon and G. A. Mihaila. WebSQL Home Page. `http://www.cs.utoronto.ca/~georgem/WebSQL.html`.

[18] A. Y. Levy, A. Rajaraman, and J. J. Ordille. Querying Heterogeneous Information Sources Using Source Descriptions. In *Proc. of 22nd International Conference on Very Large Databases (VLDB-96)*, 1996.

[19] T. Catarci, S.K. Chang, D.Nardi, G. Santucci, and M. Lenzerini. Wag: Web-at-a-glace. In *Proc. of the Hawaii International Conference on System Sciences (HICSS-31)*, January 1998.

[20] T. Catarci, G. Santucci, and M. Angelaccio. Fundamental graphical primitives for visual query languages. *Information Systems*, 18(2):75–98, 1993.

[21] T. Catarci, G. Santucci, and J. Cardiff. Graphical interaction with heterogeneous databases. *VLDB Journal*, 6(2):97–120, 1997.

[22] P. Pirolli, J. Pitkow, and R. Rao. Silk from a sow's ear: Extracting usable structures from the web. In *Proc. of CHI'96*, 1996.

[23] E. Spertus. Parasite: Mining structural information on the web. In *Proc. of the Sixth WWW Conference*, 1997.

[24] Diego Calvanese, Maurizio Lenzerini, and Daniele Nardi. A unified framework for class based representation formalisms. In J. Doyle, E. Sandewall, and P. Torasso, editors, *Proceedings of the Fourth International Conference on the Principles of Knowledge Representation and Reasoning (KR-94)*, pages 109–120, Bonn, 1994. Morgan Kaufmann, Los Altos.

[25] T. Catarci, G. Santucci, and J. Cardiff. Knowledge-based schema integration in a heterogeneous environment. In *Proc. of the Second International Workshop on Next Generation Information Technologies and Systems (NGITS95)*, 1995.

[26] T. Catarci, S. K. Chang, D. Nardi, and G. Santucci. Wag: Web-at-a-glance. Technical Report 03-97, Dipartimento di Informatica e Sistemistica, Università "La Sapienza" di Roma, 1997.