WebSuite—A Tool Suite For Harnessing Web Data

C. Beeri, G. Elber, T. Milo, Y. Sagiv, O. Shmueli, N. Tishby,
Y. Kogan, D. Konopnicki, P. Mogilevski, N. Slonim

Abstract

We present a system for searching, collecting, integrating and managing Web-resident data. The system consists of tools, each providing a specific functionality aimed at solving one aspect of the complex task of using and managing Web data. Each tool can be used in a stand-alone mode, in combination with the other tools, or even in conjunction with other systems. Together, the tools offer a range of capabilities that overcome many of the limitations in existing systems for harnessing Web data. The paper describes each tool, and possible ways of combining the tools.

1 Introduction

The *World-Wide Web* (WWW), is a readily-available domain of heterogeneous information sources. Its rapid growth and widespread use have transformed it to the central, if not universal, repository of data on essentially any topic. Unfortunately, current WWW technologies, including Web browsers and search engines, do not satisfactorily address the many problems that arise when trying to access this repository. Some of these problems are:

- Web data is weakly typed. Formats, even when they exist, are loose; many formatting decisions and logical organization are left to the discretion of the data owners. This makes extraction of the desired parts of data a difficult task.
- There often exist many sources with data on a topic. Locating them is a difficult problem. Search engines and their indices often provide grossly inaccurate answers. Even when the location of the data is approximately known, a tiresome exhaustive manual browsing is often needed to accurately pinpoint it.
- A data source may contain only partial data on a topic, and distinct sources may contain complementary data, possibly with some overlap, or with contradictions. This poses a difficult data integration problem.
- Additionally, formats in data sources may differ widely from strict structures in databases, through looser public formats for documents, or for scientific data, to owner-defined formats. Web clients therefore need to be able to handle and translate between many different formats.

To illustrate some of these problems, consider searching for bibliographical data, say on topics in Computer Science. Such data is found in many locations. Search engines sometimes help locating them, often not. Locations may be approximately known, e.g. under a university or department home page, but exact locations are site-dependent and may be difficult to pinpoint. Documents are often only weakly structured using HTML tags, and the inner logical organization is user dependent. Thus, automatic extraction of, say, article titles, becomes a difficult task.

Every developer of Web applications has to deal with, at least, some of the above issues [1, 2, 4, 5, 6, 7, 8, 9, 11, 18, 19]. A problem in many of the efforts undertaken in the past is that they adopt a 'complete system' approach. Thus their solutions are often tailored to a specific application, rather than being general. Additionally, by being parts of complete systems, solutions cannot be used independently, or utilized for other applications. In this paper, we report on a different approach undertaken in a project by the authors. Rather than focusing on building one specific Web application, we chose to build a *set of tools*, where each tool addresses one specific task. Each tool can be used in a stand-alone mode, in combination with the other tools, or even in conjunction with other systems. We believe that given a set of such tools, with generic, easy to use interfaces. Specific Web applications can then be built by a "mix and match" approach, by combining some of the tools together. Thus, the set of tools is 'greater' then the sum of its parts. The goals of our project were first, to construct useful tools, and second, and more importantly, to show that a combination of these tools enables us to easily build powerful complex Web systems. Six tools were developed in this work, five of which are described here:

- A global database server, W3GDB, that provides a global integrated database view of data sources.
- A statistical-learning mechanism, W3LEARN, for classifying, analyzing and tagging loosely structured data.

- A data translator, W3TRANS, for defining data translations among data formats.

- A query engine and a query language, W3QS and W3QL, for querying the Web.

- A personal information manager, PIM, to manage a user's data.

These tools have been described in other publications. The emphasis in this paper is in describing how they can be combined in various ways to help users address the problems described above.

In the following sections we briefly describe each of the tools. Then we describe an example application that combines them into a useful system for search, extraction, and use of bibliographic data. We also consider how tools can provide services needed by others.

2 The Global Database (W3GDB)

The global-database tool (W3GDB) provides a single, unified view of Web-resident data pertaining to a topic of interest, for example, about publications in Computer Science. The global database allows users to pose queries as if the data resides in a *single* database system, although in reality this is a virtual databases — a view. The actual data, originating in various *information sources* on the Web is represented as *local databases,* containing *local relations,*. The latter are the ones that are actually materialized.

The functionalities provided by W3GDB are as follows. First, the ability to define a local database for a new information source; or more generally to define a new global schema, and incrementally define the local databases for it and the relevant mappings. Second, the ability to formulate queries against the global relations (i.e., the conceptual schema). Third, an evaluation of those queries with respect to the local relations representing the various information sources. Note that these capabilities alone are insufficient: one also need to locate informations sources, and then analyze the data they contain, and translate the relevant parts into the local relations. These tasks are the domains of other tools in Websuite. W3GDB offers facilities that allow a user to invoke other tools when needed. Thus, in a typical scenario for using W3GDB, a user browses the available schemas, and if a global database for the desired topic exists, queries can be posed. If answers are found to be unsatisfactory, then others tools are invoked to search for additional sources, to analyze and translate them into additional local relations, until the user is satisfied with the answer.

The W3GDB approach is similar to that of the Information Manifold [15] in the sense that it describes information sources as local relations and maps those local relations to the global ones. (Other systems (e.g., TSIMMIS [12] and HERMES [21]) concentrate on translating queries directly rather than by mapping local relations to global ones.) But there are some important differences, having to do with common problems one faces in using such a system. First, when many sources are available on a topic, one needs to write many translations into local relations, one per source. There is also a need to define a mapping between the local and global relations. Both are error-prone. In W3GDB, special efforts have been invested in addressing these problems. First, the system can handle null values in local relations by using skolem-function based representation. This means that a local schema does not have to be an exact reflection of the structure of the data found at an information source, hence it becomes possible to use the same local schema for information sources of a similar nature. A second important difference is that W3GDB adopts the *global as view* approach (the Information Manifold use the *local as view*): Global relations are expressed in terms of local relations, and therefore, translating a query on the global relations into a query on the local relations is straightforward and is done by means of rule substitutions. Additionally, the mapping does not change much when moving from one information source to another. Thus, the overhead in adding new information sources that are similar to existing ones is minimal.

Further flexibility and ease of use arise since the mapping rules can directly encode some information that is not stored in the local relations. For example, we may want to have for each publication an attribute *Institute* that specifies the institute in which that publication was found. This information is often not included explicitly in publications.

3 Statistical Categorization and Tagging (W3LEARN)

A database query is usually formulated in terms of relation/class names and attributes. But information sources may contain relevant data that is not tagged with these terms; sometimes the data is simply not tagged, sometimes, the tags used in it are different, although the meanings are the same or similar. This presents a severe problem when trying to access such data using a query language.

W3LEARN addresses this issue using statistical categorization from examples, using machine learning techniques. In learning from examples, an unknown input-output relation ("rule") is being *randomly* sampled to provide a set of *training examples*. From these training examples, a *learning algorithm* generates an *hypothesis* relation, from an *hypotheses class*, that approximates the original input-output rule. Under certain technical conditions good performance (low classification error) *outside* of the training sample, on arbitrary data taken from the same distribution as the training data can be guaranteed. Thus, W3LEARN can deal with fuzzy, noisy, and irregular input. This is essential in dealing with the large variety and diversity of information sources over the Web. In Websuite, the tool has been applied to two generic and fundamental problems: automatic file (URL) categorization, and statistical segmentation (tagging) into fields. For lack of space, only the second is described here.

The specific problem addressed here is is automatic segmentation of text into fields with statistically distinguishable signatures, such as bibliography items. The essence of the problem here is in the fast switching between the different statistical sources of the various fields. Here it is not only essential that the fields are statistically well discriminated, but that there is also a stochastic switch which has to be identified precisely. In addition to statistical information, the tool also utilizes information about the temporal order of the items and the possible field separators. It is based on the technique of hidden Markov models (HMM), where the different fields are considered as the hidden states of the Markov source. The training, in this case, is easily performed using *"BIBTEX"* files, which are already tagged. This is also our output format.

The input file to this tool can be, for example, a HTML file which contains bibliography items. Locations of fields such as author(s), title etc. are not indicated by tags, and different people may represent them differently. The output is a tagged file in *BIBTEX* format.

4 The Translator (W3TRANS)

Data on the Web are stored under many different formats: relational or oo databases, SGML or Latex (documents), DX formats (scientific data), Step (CAD/CAM data), etc. But application programs typically can only handle a few formats. To enable an application to manipulate data coming from various sources, such data need be translated to a format expected by the application.

Currently two kinds of tools are available for such mappings. Specialized tools map from one format to another (e.g., from Latex to HTML). They are easy to use but restricted in scope. More general tools allow the user to specify a mapping (e.g., ICA[16]). These still have many drawbacks: they are difficult to use, they do not support inverse translations, which is necessary to propagate updates from target to source, and they do not use a declarative language and thus can hardly be analyzed for optimization or update purposes [10].

W3TRANS is a translation tool that overcomes the above shortcomings. It offers a *middleware data model* to which various data sources are mapped, and a declarative language, TreeLog, for specifying translations within the middleware model. The data model uses *ordered labeled trees*, and is similar to the one used in [6] and to the OEM data model ([12, 19]), but it includes an important extension: It supports order on the children of nodes, so it can represent ordered collections. This data model serves as a common ground for data formats, That is, it can easily represent many of the formats common on the Web. Translations to/from the middleware format to external formats are easy to define, and have been described in the literature.

The translation scheme consists of 3 phases: In the first, data is imported from the source to the middleware data model. Then, the data is translated from one representation to another. Finally, the data is exported from its second representation to the target. The heart of W3TRANS is its handling of the second phase, namely translations inside the middleware. For that, the *TreeLog* language [3, 17] is used. It is a rule-based declarative language, with a fixpoint semantics. It differs from other rule-based languages in its strong tree-awareness: it provides syntactic constructs to search and build complex trees that represent data. A significant advantage of the language is that in most practical cases, translation, correspondences between data elements, and inverse translations, can all be automatically derived from one TreeLog program [3]. It follows that update propagation form a target format back to the source format can be easily supported.

5 W3QL and W3QS

W3QL [14] is a high level query language designed to operate against the WWW. W3QS is a system that provides a W3QL implementation and additional services.

W3QL is a SQL-like query language. It enables specifying connection patterns over the WWW (viewed as a "directed graph") as well as specifying content conditions. The W3QL "From" clause specifies paths and nodes on them. Paths may be *unbounded*, i.e. the path's edges may be left unspecified. Further, one may limit the length of unbounded paths, thus restricting the search. The "Where" clause allows one to apply predicates to the contents of nodes. An important facility offered in the language is to use predicates expressed in external, existing languages, e.g., PERL. Such a facility greatly enhances the expressiveness and usefulness of the language, in that it allows users to express conditions in a language they are familiar with, and it offers, 'for free', the full range of predicates expressible in such languages.

As a typical example, if one is interested in publications by certain authors found on the Web site of a company, one can specify a in the "From" clause a search that starts from the home page of the company, and proceeds one link, a bounded number of links, or an unbounded number of links. In the Where clause, one can use PERL to specify the desired authors, to look for certain keywords, and so on.

The W3QS system offers additional facilities. For example, not always does one know where to start a search. Starting points may be obtained from an established search service (e.g. Infoseek). The system offers support in filling forms for such engines, and for other sites where forms are used, and it remembers previous form filling activities. Other services include query composition, query directories, timed execution of queries, and view management and maintenance.

6 The Personal Information Manager (PIM)

A significant part of the daily work of professionals revolves around their workstation, and much of it involves (email) message and file management. Many systems now support graphical window-based interfaces. There seems to be a much slower convergence of data storage and management services. Users are still very much responsible for the structure and placement of their directories and file. Many tools do not fully exploit the potential of structured data; and whatever facilities they do offer are often tool-specific. The main idea underlying PIM is to explore database technology to offer a collection of data management services on which various applications can be based. These include schema and storage management, creation and management of collections, browsing facilities, querying and updating. Applications are clients of these services. Basing applications on such services facilitates data sharing across applications, and allows connecting data items according to various criteria, and flexible updating. Typical tasks handled by clients included sending, receiving and organizing email, bibliography management, task and meeting scheduling.

To see the benefits of such a system, consider a mailer client. It handles all the user interaction through a graphical interface. Messages that are not discarded right away are stored in one or more folders. Messages can be linked to other messages in the same or other folders. They can be put on task queues, and linked to meeting records. The mailer can extract several useful fields from a message, others (e.g., expiration time, key words) can be added by the user, at any time. A user can update a message, e.g., by comments describing how it has been handled. Email folders, user lists, a meeting calendar, task queues, are all collections of items, uniformly managed by the underlying services, thus enabling data sharing among different clients. Since messages and other items have fields, the query language can be used to define and retrieve collections of items according to varied criteria. An object-oriented approach that associates behavior with objects and collections further facilitates the construction of clients.

7 Cooperation Between Tools

By combining the tools in different ways, it is possible to construct applications that overcome many of the limitations in existing systems for utilizing Web data. Additionally, each of the tools can benefit from specific services offered by the other tools. We describe one application that we have implemented, and mention ways tools can be used by other tools.

The example application concerns publications of Computer Science researchers. A user may manage a personal, local folder of bibliographic data, using PIM. Often there is a need for additional data, so the user utilizes W3GDB, to query additional sources that are represented by a global database. The data for the global database is derived from many local databases, where each local database represents some information source found on the Web. If that turns out to be insufficient, he/she may also search the Web for new information sources about publications and add those sources to the global database. While our system does not eliminate the need for user guidance and intervention in searching for the new sources and adding them to the global database, our tools drastically reduce the amount of human work that is needed. We briefly explain how that reduction is achieved.

First, searching for new information sources is done by combining the capabilities of W3QL and W3LEARN. W3QL enables the user to phrase succinct queries that search only parts of the Web that are deemed as relevant by the user, for example, pages reachable from university home pages. The query may include conditions on the graph structure (e.g., only pages at distance 3 or less from a given page), and conditions on the contents (e.g., "databases" must occur in the page). W3LEARN serves as a completely automatic filter that determines whether a given file contains publications. Technically, W3LEARN gives an estimation of the relevance of the data in the document to the subject of interest (i.e., listings of publications, in our case). Thus, there is no need to manually inspect the result of a W3QL query in order to determine which files of the result are lists of publications.

After locating a source of information about publications, there is a need to build a wrapper. That is accomplished as follows: First, W3LEARN analyzes the document and adds tags, according to a predetermined format, for indicating authors, titles, etc. Then W3TRANS provides the translation. The use of statistical tagging, enables us to use the same wrapper for statistically similar information sources. The flexibility in handling optional and null fields of W3GDB further reduces the needs for different wrappers. Thus, in this application, we defined one local schema with all the fields we expect to find about publications, and only one translation program had to be written. Similarly, the rules defining the mapping between the local and global relations needed only minimal changes when moving from one information source to another. Typically, we had to change the names of local relations and some information that is specific to an information source (e.g., the name of the institute in which the publications were found).

We now briefly describe some inter-tool cooperation options. The Web query engine W3QS can utilize the capabilities of W3LEARN, to provide better data filtering and selection facilities. E.g., in the example application above, the user is interested only in publication files. These can be identified, with high probability by W3LEARN. As another example, the combination of W3LEARN and W3TRANS supports the construction of uniform wrappers for data sources that do not use identical formats, since the tagging provided by W3LEARN offers an alternative, uniform format. This combined mechanism is more general than most of the special-purpose wrappers developed for specific Web applications [12, 13]. Of course, W3TRANS can also used alone for well-structured sources. W3QS provides an automatic view maintenance service for views defined by W3QL queries. This service can be used by W3GDB to automatically maintain local relations up-to-date. Finally, W3LEARN can be used to advantage by PIM and its mailer, for the classification of incoming mail. Garbage can be identified and discarded, the rest can be classified by relative significance, and appropriate folders and related messages may be proposed for each message.

8 Conclusions

In this paper we presented a set of tools for searching, collecting, integrating and managing personal and Web data, each aimed at solving one fundamental problem. Each tool can be used in a stand-alone mode, in combination with the other tools, or even in conjunction with other systems. The combination of these tools enables the building of complex Web based applications in a flexible way. We have experimented with the tools, and with mixing and matching them, in constructing combined applications, one of which is described above. We also discussed other ways the tools can support and enhance each other.

We believe that, in general, such a modular set of tools can be more powerful than monolithic centralized systems. This is because (1) each component is developed by experts in the tool's specific area of application, and thus is state of the art for the specific task, and (2) one can use expertise from a variety of areas, and (3) since the modules can be combined in various ways to serve specific needs. We further believe that developing methodologies for "wrapping applications" out of tools available across the network is a fruitful research direction,

We mention that our project involved another tool that was not discussed above, for lack of space, a virtual environment generator, VRG. It can be used with the tools above, for example, for simulating an office environment, or for simulating a library environment for a document search environment.

References

[1] S. Abiteboul, S. Cluet, T. Milo. Querying the file. *Proc. VLDB*, Dublin, 1993

[2] S. Abiteboul, S. Cluet and T. Milo, A database interface for files update, *Proc. SIGMOD*, May 1995.

[3] S. Abiteboul, S. Cluet, and T. Milo. Correspondence and Translation for Heterogeneous Data. In *Proc. ICDT*, pages 351-363. 1997.

[4] P. Atzeni, S. Labonia, A. Masci, G. Mecca, P. Merialdo and E. Tabet. "The ARANEUS Project," http://poincare.inf.uniroma3.it:8080/Araneus/araneus.html

[5] P. Buneman, S. Davidson, K. Hart, C. Overton, and L. Wong. A data transformation system for biological data sources. In *Proc. VLDB*, 158–169, 1995.

[6] P. Buneman, S. Davidson, and D. Suciu. Programming constructs for unstructured data, May 1996.

[7] M.J. Carey et al. Towards heterogeneous multimedia information systems: The Garlic approach. Technical Report RJ 9911, IBM Almaden Research Center, 1994.

[8] V. Christophides, S. Abiteboul, S. Cluet, and M. Scholl. From structured documents to novel query facilities. In *Proc. SIGMOD*, 1994.

[9] M. Consens, T. Milo. Optimizing Queries on Files, *SIGMOD*, 301-312, 1994.

[10] A. Feng and T. Wakayama. Simon: A Grammar Based Transformation System of Structured Documents, *Proc. Int. Conf. Electronic Publishing*,1994.

[11] J.C. Franchitti and R. King. Amalgame: a tool for creating interoperating persistent, heterogeneous components. *Advanced Database Systems*, pages 313–36, 1993.

[12] H. Garcia-Molina, D. Quass, Y. Papakonstantinou, A. Rajaraman, Y. Sagiv, J. D. Ullman, and J. Widom. "The TSIMMIS Approach to Mediation: Data Models and Languages," to appear *International Journal of Intelligent Information Systems;*

[13] T. Kirk, A. Y. Levy, Y. Sagiv, and D. Srivastava. "The Information Manifold," *AI Spring Symp.,* 1995.

[14] Konopnicki D., and O. Shmueli. W3QS: A Query System for the World-Wide Web. In *Proc. VLDB*, 1995.

[15] A. Levy, A. Rajaraman, and J. Ordille. "The World-Wide Web as a Collection of Views: Query Processing in the Information Manifold" *Proc. Workshop on Materialized Views: Techniques and Applications,* Montreal, Canada, pp. 43–55, 1996.

[16] A. Mamrak, C. O'Connell, Technical Documentation for the Integrated Chameleon Architecture. In *anonymous ftp at ftp.ifi.uio.no /pub/SGML/ICA,* 1992.

[17] P. Mogilevski. Integration and Translation of Heterogeneous Data. M.Sc Thesis, Tel-Aviv University. 1997.

[18] Y. Papakonstantinou, H. Garcia-Molina, and J. Ullman. Medmaker: A mediation system based on declarative specifications. Available by anonymous ftp at `db.stanford.edu` as the file `~/pub/papakonstantinou/1995/medmaker.ps`.

[19] Y. Papakonstantinou, H. Garcia-Molina, and J. Widom. Object exchange across heterogeneous information sources. In *Proc. ICDE*, 1995.

[20] N. Slonim and N. Tishby, *Automatic statistical categorization and segmentation of text*, HUJI Technical Report, To appear.

[21] V.S. Subrahmanian, S. Adali, A. Brink, R. Emery, J. Lu, A. Rajput, T. Rogers, R. Ross, and C. Ward. "HERMES: A Heterogeneous Reasoning and Mediator System. Tech. Report, U. of Maryland, 1995.