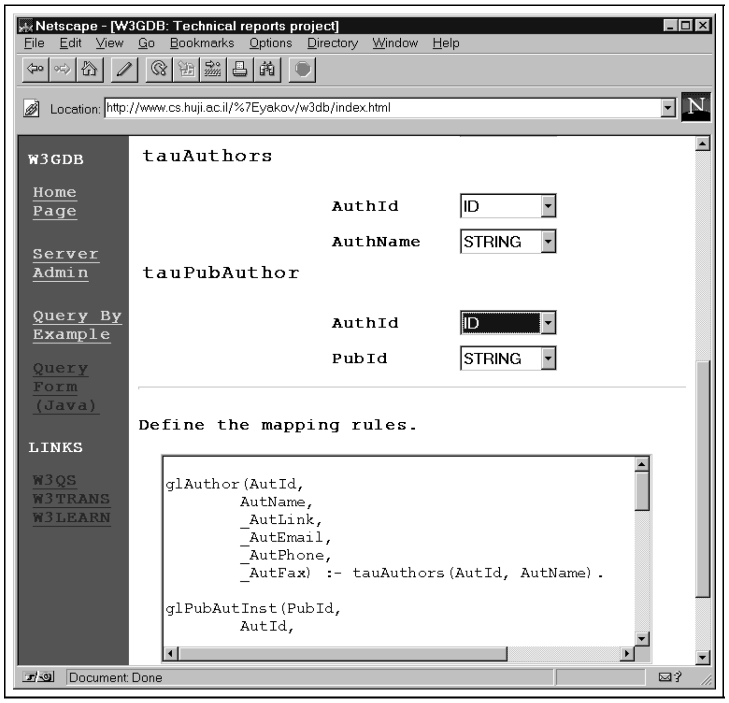

Netscape - [W3GDB: Technical reports project]
File Edit View Go Bookmarks Options Directory Window Help
Location: http://www.cs.huji.ac.il/%7Eyakov/w3db/index.html
W3GDB
Home Page
Server Admin
Query By Example
Query Form (Java)
LINKS
W3QS
W3TRANS
W3LEARN
Document: Done

tauAuthors

AuthId	ID
AuthName	STRING

tauPubAuthor

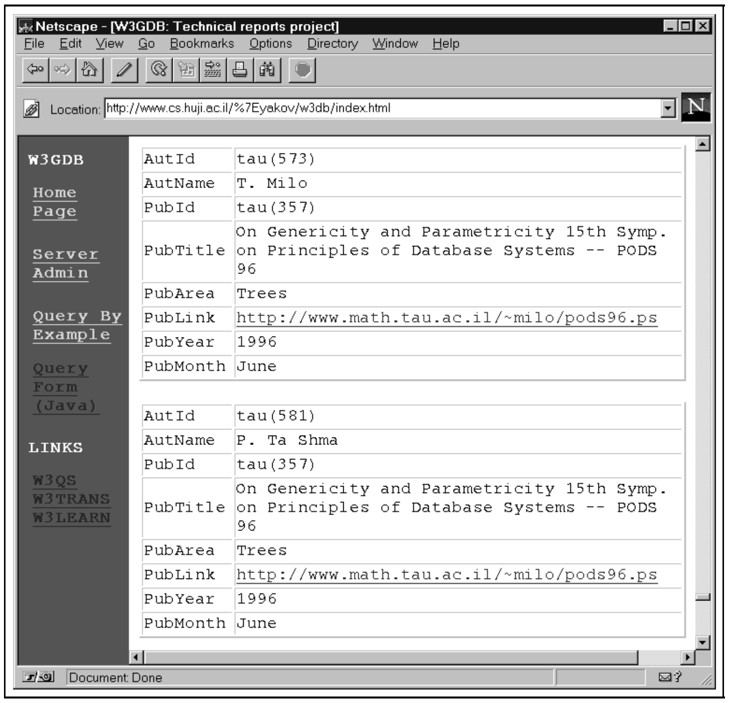
AuthId	ID
PubId	STRING

Define the mapping rules.

```
glAuthor(AutId,
        AutName,
        _AutLink,
        _AutEmail,
        _AutPhone,
        _AutFax)  :- tauAuthors(AutId, AutName).

glPubAutInst(PubId,
        AutId,
```

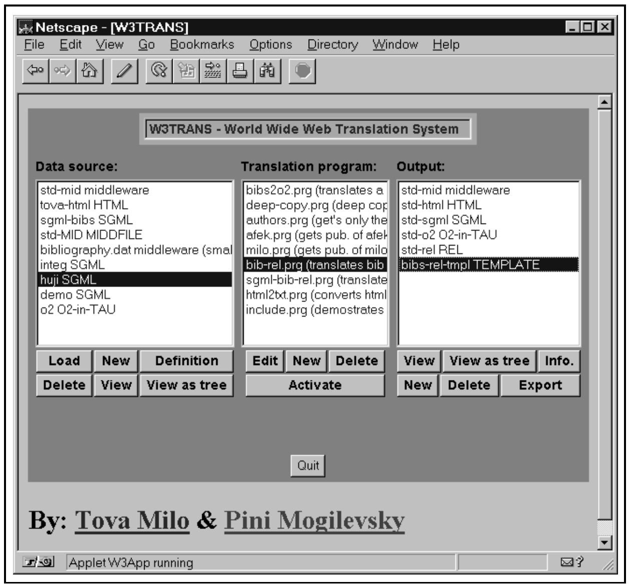
Figure 7: Defining the Mapping from the Local to the Global Relations.


Netscape - [W3GDB: Technical reports project]
File Edit View Go Bookmarks Options Directory Window Help
Location: http://www.cs.huji.ac.il/%7Eyakov/w3db/index.html
W3GDB
Home Page
Server Admin
Query By Example
Query Form (Java)
LINKS
W3QS
W3TRANS
W3LEARN
Document: Done

AutId	tau(573)
AutName	T. Milo
PubId	tau(357)
PubTitle	On Genericity and Parametricity 15th Symp. on Principles of Database Systems -- PODS 96
PubArea	Trees
PubLink	http://www.math.tau.ac.il/~milo/pods96.ps
PubYear	1996
PubMonth	June

AutId	tau(581)
AutName	P. Ta Shma
PubId	tau(357)
PubTitle	On Genericity and Parametricity 15th Symp. on Principles of Database Systems -- PODS 96
PubArea	Trees
PubLink	http://www.math.tau.ac.il/~milo/pods96.ps
PubYear	1996
PubMonth	June

Figure 8: The Result of the Global Query.



Data source:

std-mid middleware
tova-html HTML
sgml-bibs SGML
std-MID MIDDFILE
bibliography.dat middleware (smal
integ SGML
huji SGML
demo SGML
o2 O2-in-TAU

Translation program:

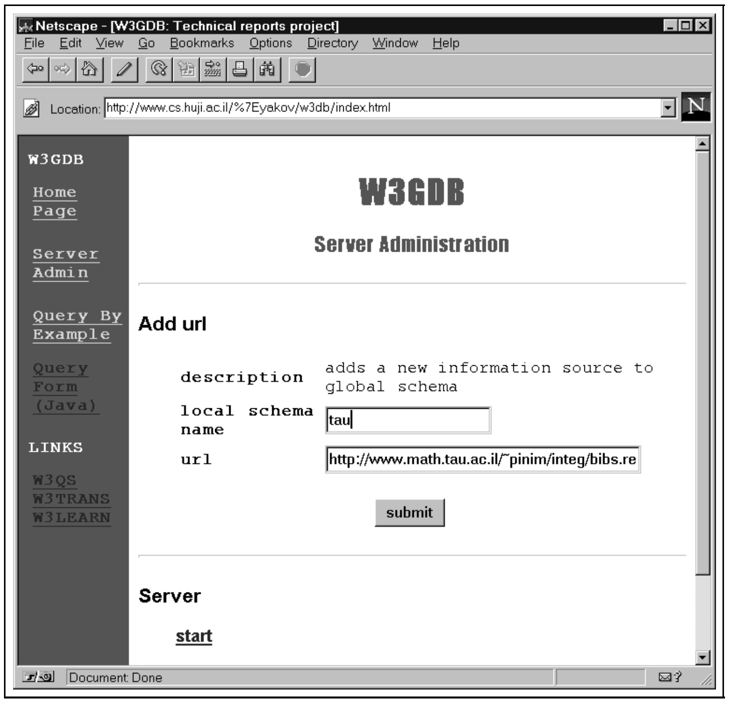
bibs2o2.prg (translates a
deep-copy.prg (deep cop
authors.prg (get's only the
afek.prg (gets pub. of afek
milo.prg (gets pub. of milo
bib-rel.prg (translates bib
sgml-bib-rel.prg (translate
html2txt.prg (converts html
include.prg (demostrates

Output:

std-mid middleware
std-html HTML
std-sgml SGML
std-o2 O2-in-TAU
std-rel REL
bibs-rel-tmpl TEMPLATE

By: Tova Milo & Pini Mogilevsky

Figure 5: Translating the Data into Local Relations.


Netscape - [W3GDB: Technical reports project]
File Edit View Go Bookmarks Options Directory Window Help
Location: http://www.cs.huji.ac.il/%7Eyakov/w3db/index.html
W3GDB
Home Page
Server Admin
Query By Example
Query Form (Java)
LINKS
W3QS
W3TRANS
W3LEARN
submit
start
Document: Done

W3GDB

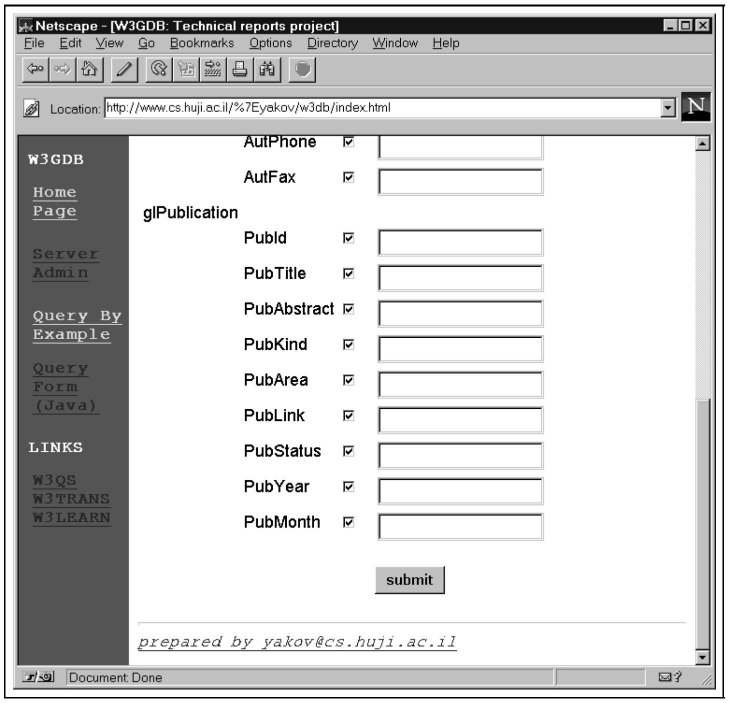
Server Administration

Add url

description	adds a new information source to global schema
local schema name	tau
url	http://www.math.tau.ac.il/~pinim/integ/bibs.re

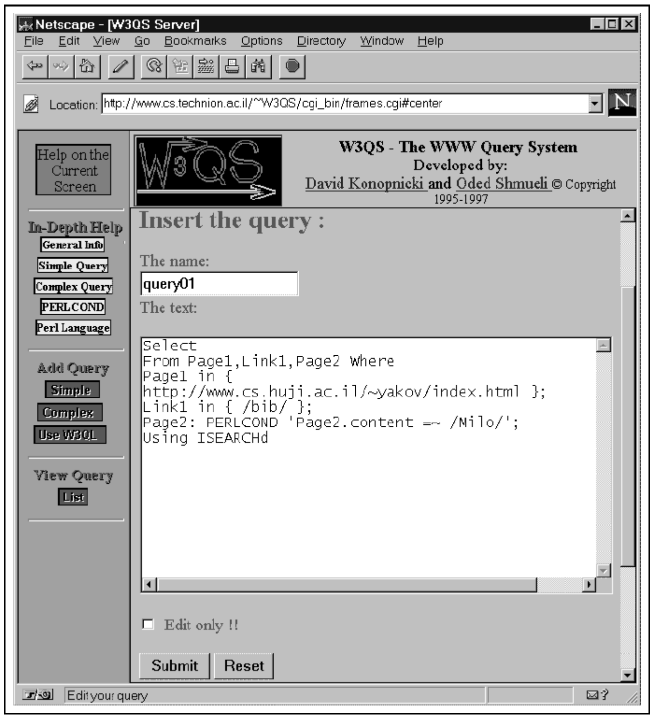
Server

Figure 6: Adding a New Source.


Netscape - [W3GDB: Technical reports project]
File Edit View Go Bookmarks Options Directory Window Help
Location: http://www.cs.huji.ac.il/%7Eyakov/w3db/index.html
W3GDB
Home Page
Server Admin
Query By Example
Query Form (Java)
LINKS
W3QS
W3TRANS
W3LEARN
submit
prepared by yakov@cs.huji.ac.il
Document: Done

AutPhone	☑
AutFax	☑
glPublication	
PubId	☑
PubTitle	☑
PubAbstract	☑
PubKind	☑
PubArea	☑
PubLink	☑
PubStatus	☑
PubYear	☑
PubMonth	☑

Figure 3: Defining the Query.


Netscape - [W3QS Server]
File Edit View Go Bookmarks Options Directory Window Help
Location: http://www.cs.technion.ac.il/~W3QS/cgi_bin/frames.cgi#center
Help on the Current Screen
In-Depth Help
General Info
Simple Query
Complex Query
PERLCOND
Perl Language
Add Query
Simple
Complex
Use W3QL
View Query
List
W3QS
W3QS - The WWW Query System
Developed by:
David Konopnicki and Oded Shmueli © Copyright 1995-1997
Insert the query :
Submit
Reset
Edit your query

The name: query01

The text:

```
Select
From Page1,Link1,Page2 Where
Page1 in {
http://www.cs.huji.ac.il/~yakov/index.html };
Link1 in { /bib/ };
Page2: PERLCOND 'Page2.content =~ /Milo/';
Using ISEARCHd
```

☐ Edit only !!

Figure 4: Searching for Data Sources Using W3QL.

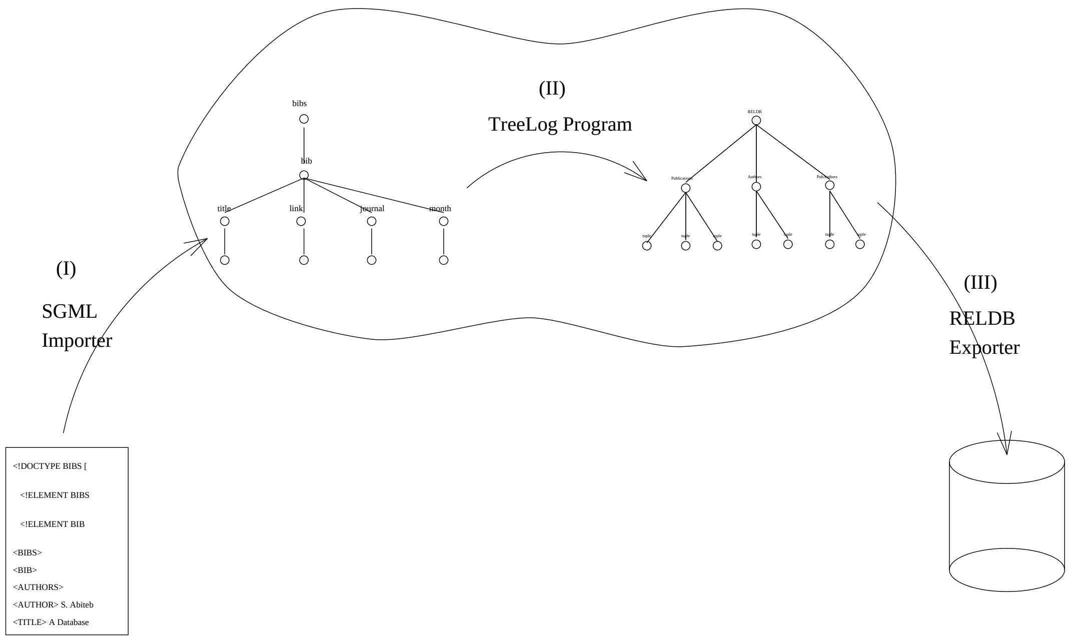
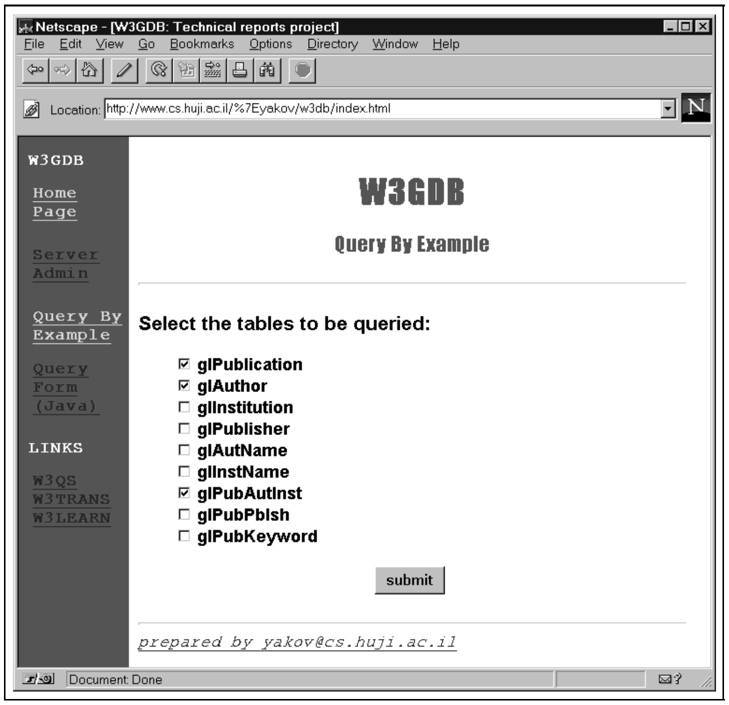


Figure 1: The Translation Process


Netscape - [W3GDB: Technical reports project]
File Edit View Go Bookmarks Options Directory Window Help
Location: http://www.cs.huji.ac.il/%7Eyakov/w3db/index.html
W3GDB
Home Page
Server Admin
Query By Example
Query Form (Java)
LINKS
W3QS
W3TRANS
W3LEARN
submit
Document: Done

W3GDB

Query By Example

Select the tables to be queried:

- [✓] glPublication
- [✓] glAuthor
- [ ] glInstitution
- [ ] glPublisher
- [ ] glAutName
- [ ] glInstName
- [✓] glPubAutInst
- [ ] glPubPblsh
- [ ] glPubKeyword

prepared by yakov@cs.huji.ac.il

Figure 2: Selecting Relations for the Query.

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

[24] Konopnicki D., and O. Shmueli. W3QS: A Query System for the World-Wide Web. In *Proceedings of 1995 VLDB Conference,* , Zurich, Switzerland, September 1995.

[25] 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.

[26] A. Y. Levy, A. O. Mendelzon, Y. Sagiv, and D. Srivastava. "Answering Queries Using Views," *Proc. Fourteenth ACM SIGACT-SIGART-SIGMOD Symp. on Principles of Database Systems,* 1995.

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

[28] A. Mendelzon, G. Mihaila, and T. Milo. Querying the world wide. In *Proc. of PDIS'96*, 1996.

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

[30] S. Mukherjea and J.D. Foley. Visualizing the World Wide Web with the Navigational View Builder. *Computer Networks and ISDN Systems*, 27:1075–1087, 1995.

[31] 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`.

[32] Y. Papakonstantinou, H. Garcia-Molina, and J. Widom. Object exchange across heterogeneous information sources. In *International Conference on Data Engineering*, 1995.

[33] J.E. Pitkow and K.A. Bharat. WebViz: A tools for World Wide Web access log analysis. In *Proceedings of the First International World Wide Web Conference*, Geneva, Switzerland, May 1994. `http://-www1.cern.ch/PapersWWW94/pitkow-webvis.ps`.

[34] D. Quass, A. Rajaraman, Y. Sagiv, J. D. Ullman, and J. Widom. "Querying Semistructured Heterogeneous Information," *Deductive and Object-Oriented Databases* (Proceedings Fourth Int. Conf., DOOD'95, Singapore, December 1995, Tok Wang Ling, Alberto O. Mendelzon, and Laurent Vieille (Eds.)), LNCS 1013, Springer, pp. 319–344, 1995.

[35] A. Rajaraman, Y. Sagiv, J. D. Ullman. "Answering Queries Using Templates With Binding Patterns," *Proc. Fourteenth ACM SIGACT-SIGART-SIGMOD Symp. on Principles of Database Systems,* 1995.

[36] K. Shoens, A. Luniewski, P. Schwartz, J. Stamos, and J . Thomas. The rofus system: Information organization for semi-structured data. In *Proc. of the 19th Int. conf. on Very Large Databases, VLDB93*, pages 97–107, 1993.

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

[38] 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.

[39] J. Walker. HTML Converters. In *http://www2.ncsu.edu/bae/people/faculty/walker/hotlist/htmlconv.html*, 1994.

[40] Yahoo Inc. Yahoo: Main page. 1996. `http://www.yahoo.com`.

[4] Paolo Atzeni, Salvatore Labonia, Alessandro Masci, Giansalvatore Mecca, Paolo Merialdo and Elena Tabet. "The ARANEUS Project," http://poincare.inf.uniroma3.it:8080/Araneus/araneus.html

[5] G. Bell, A. Parisi, and M. Pesce. The Virtual Reality Modeling Language: Version 1 Specification. May 1995. http://www.virtpark.com/theme/vrml/.

[6] P. Buneman, S. Davidson, K. Hart, C. Overton, and L. Wong. A data transformation system for biological data sources. In *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pages 158–169, Zurich, Switzerland, 1995.

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

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

[9] V. Christophides, S. Abiteboul, S. Cluet, and M. Scholl. From structured documents to novel query facilities. In *Proc. ACM Sigmod, Minneapolis*, 1994.

[10] T.-P. Chang and R. Hull. Using witness generators to support bi-directional update between object-based databases. In *Proc. ACM SIGMOD/SIGACT Conf. on Princ. of Database Syst. (PODS)*, San Jose, California, May 1995.

[11] M. Consens, T. Milo. Optimizing Queries on Files, *ACM SIGMOD Int. Conf. on Management of Data*, Minneapolis, Minnesota, 301-312,May 1994.

[12] Cosmo Player. Silicon Graphics. Netscape plugin for VRML 2.0. May 1996, Mountain View, CA. http://vrml.sgi.com/cosmoplayer/help/index.html.

[13] F. Das Neves. The aleph: A tool to spatially represent user knowledge about the www docuverse. In *to appear in Proc. ACM Hypertext '97*, 1997.

[14] Peter Doemel. WebMap – a graphical hypertext navigation tool. In *Proceedings of the Second International World Wide Web Conference*, Chicago, October 1994. `http://www.ncsa.uiuc.edu/SDG/IT94/-Proceedings/Searching/doemel/www-fall94.html`.

[15] Gershon Elber. IRIT solid modeller. IRIT 6.0 User's Manual, Technion, March 1996.

[16] Excite Inc. Excite: Main page. 1996. `http://www.excite.com`.

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

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

[19] 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 in a special issue of the *International Journal of Intelligent Information Systems;* extended abstract appeared in the *Second Int. Workshop on Next Generation Information Technologies and Systems,* 1995.

[20] M. Hemmje. Lyberworld - a visulalization user interface with full text retrieval. In *Proceesings of SIGIR 94*, 1994.

[21] Iris Performer Programmer's Guide. Document Number 007-1680-030, Silicon Graphics, May 1996, Mountain View, CA.

[22] M. Kifer, G. Lausen, and J. Wu. "Logical Foundations of Object-Oriented and Frame-Based Languages," *JACM,* Vol. 42, No. 4, pp. 741–843, 1995.

fruitful research direction, that of "wrapping applications" out of basic tools that will be available across the network.

The bibliography scenario is an example of a particular kind of cooperation. A W3GDB task is carried out with the aid of W3QS (WWW searching), W3LEARN (source utility determination and tagging) and W3TRANS (format conversion). This type of cooperation is by no means the only one possible. We list several possible modes of cooperation among the tools:

1. Direct W3GDB and W3QS cooperation. W3GDB can utilize W3QS querying capabilities directly for obtaining data from sites that are known to be relevant for a task. It may use W3QS's view mechanism to make sure that certain sources are accessed at pre-determined time intervals and are monitored for changes. It may also utilize W3LEARN to extract certain pieces of information, without using W3TRANS's format transformation capabilities.

2. W3GDB and W3QS may utilize W3TRANS directly, without W3LEARN, in accessing known structured sources or (relational or object oriented) databases.

3. PIM can utilize each of these tools in obtaining and organizing data.

4. W3QS may use W3LEARN for examining information sources, as part of W3QL queries. Here, W3LEARN offers certain services in classifying the types of sources which W3QS can use. Currently, W3QS can recognize only certain information source types (e.g. Postscript or Latex files) on its own. An ability of recognizing and parsing a larger set of source types (e.g. articles on distributed databases) will greatly enhance W3QL's querying ability.

WebSuite includes one more component that we have not discussed in this paper. The user interface to Web data is rather limited. The navigational nature of most Web browsers provides a rather narrow visual window, showing only one page at a time. Other visualization paradigms (e.g. [20, 13, 30, 33, 14]) enable the user to work with groups of documents and highlight issues such as the network structure and the documents content. However, they are still rather isolated from the local user environment and the other applications the user is working with. To overcome these limitations, we have developed a graphical interactor, VRG, that uses virtual reality as a user interface for viewing and interacting with data from the Web as well as from other sources. VRG related material may be found at `http://www.cs.technion.ac.il/~anda/files/WebSuite.html`.

Acknowledgments

Tova Milo thanks Serge Abiteboul and Sophie Cluet for their contribution to the development of the formal foundation of the data translation system W3TRANS.

References

[1] S. Abiteboul, S. Cluet, T. Milo. Querying the file. *Proc. of Intl. Conf. on Very Large Data Bases*, Dublin, 1993

[2] S. Abiteboul, S. Cluet and T. Milo, A database interface for files update, *Proc. ACM SIGMOD Int. Conf. on Management of Data*, May 1995.

[3] S. Abiteboul, S. Cluet, and T. Milo. Correspondence and Translation for Heterogeneous Data. In *Proc. Int. Conf. on Database Theory (ICDT) 1997*, pages 351-363. 1997.

that a folder does not correspond to a physical file, since a message can be included in any number of folders. Task queues are also folders; this illustrates one way of sharing data across clients. A message is identified as requiring an action, and is put into a task queue, possibly after a comment on the required action has been added. Other sources and consumers of structured data are the address book, meeting calendar manager, task queues, and the bibliographic manager.

Of course, some of the data contains (long) text fields, for example (parts of) papers. We posit that even these should be viewed as having some structure in addition to the textual part. These additional fields could be the language (e.g., Latex), an id, and a few more fields that can be useful for version management. A pure text file, suitable for editor operation, can easily be generated as needed, and after changes are complete, it can be checked back into the version manager.

An important point is that one can associate a specific behavior with a folder. The simplest example is displaying the fields that are defined for the folder. Different folders may be associated with different file organizations. In many cases, it may be possible to make a good guess about a candidate folder from information in the incoming message, so that an appropriate suggestion and format of presentation are selected by the mailer. Such information may be useful also for deciding the relative significance of a message, since a mailer can be programmed to reorder incoming mail by significance, and even to not display mail about some topics. While some such information may be gleaned from the headers, in most cases it will depend on the contents, and it is typically highly dependent on the user. For this purpose, the W3LEARN module is a natural candidate for performing preliminary analysis of incoming messages. We intend to explore this avenue.

We briefly discuss some other potential clients. Most scientists have a personal bibliography database, say in Bibtex format. As text files, simple queries using grep are supported, but this format has fields, so it supports complex queries. A bibliography manager will allow one to collect/enter new items (for example by using WGDB, W3QS, W3LEARN and W3TRANS, as explained in the previous section), to create an initial bibliography file for a paper by posing queries, collecting the results in temporary files, browsing to delete some items and merging into a permanent file to be stored by the version manager. Such a manager can be extended to support a view of items as objects, with specific behavior. For example, a conference series might support a specific style of printing the conference reference, with the date and location added as appropriate.

Another important client is the task queue. In principle, there can be more than one task queue. Queue management should go beyond insert and delete, to supporting reordering, and to recording relevant actions. A meeting calendar is essentially such a queue, with facilities for displaying or printing.

8 Conclusions

In this paper we presented a system for searching, collecting, integrating and managing Web data. The system consists of five tools, each is aimed at solving one fundamental problem encountered when attempting to utilize 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. 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. We believe that, in general, systems that are developed in such a modular fashion can be more powerful than those developed in a monolithic centralized manner. 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 (e.g. in our system, AI and Databases) to enrich the system capabilities. This effort points to a

7 The Personal Information Manager (PIM)

7.1 The Main Idea

The Personal Information Manager (PIM) is used to manage data collected from the Web as well as from other sources. One can view PIM as residing in a layer *above* the previously described tools, taking advantage of the many services provided by WebSuite's, as well as other, tools.

Currently, a significant part of the daily work of scientists and other professionals revolves around their workstation/PC. Many tools now support graphical window-based interfaces. There seems to be a much slower convergence for the data storage and handling component. A user normally views directly the file system and has to be aware of, and responsible for, the location of directories and files. Many tools do not exploit the potential of structured data, and when they do, it is often a tool-specific view of the structure. Different tools tend to share only the byte-oriented view of the data. The main idea underlying PIM is to explore Database ideas and technologies so that first, that much of the schema and storage organization tasks can be automated, and second to facilitate the sharing of data and its structure (where available and relevant) across all tools and applications employed by a user.

PIM consists of modules of two kinds: services and clients. The services include a suite of tools for data management: a schema manager, a browser, a query language, a buffer manager, indexing tools, and so on. Clients are the tools used for everyday work: a mailer, a bibliography manager, an address book, a meeting calendar, a task queues, and a version manager for papers and software.

Many projects that deal with data from multiple sources face the issue of deciding on a data model. Choices include the relational model [23], or a light weight object model [34]. We have opted for flexibility. Our query language supports a rich object model, that includes the relational and nested relational models as special cases. That is, structures can be nested, and both values and objects are supported. This allows a user to use a relational model for one application, and an object model for another. Additionally, we allow individual records in a file to have fields that are not in the file's schema; these should be supported by the query language.

The basic facilities offered by the data services layer are as follows. One can compose and execute queries; query results are held in temporary files, which can be browsed and updated, and can be made permanent. The query processor relies on the schema manager to supply the information needed to process queries.

Personal information (as handled today) consists of thousands of files; yet, most files are quite small. That is, from a database perspective, the volume of data is relatively small, and for reasonable processing strategies, performance is not expected to be an issue. On the other hand, since the data involves many subjects and possibly many formats, schema and storage management need to provide enhanced support. In particular, one should be able to browse and update the schema. To support many file formats, a suitable DDL will be developed. The W3TRANS module will be used to enable the query processor to read from and write to files of various formats.

7.2 An Example of an Implemented Client: The Mailer Client

A lot of the data relevant to everyday work originates in e-mail messages. Except for messages that are discarded right away, most messages need to be stored in a way that facilitates subsequent retrieval and manipulation. Now, a message has a few fields that are of interest: sender identity and e-mail address, time of arrival, subject and body. Additional fields are added when the message is received: an id; an expiry date (to allow the system to purge *old* messages automatically); recording of a response (or responses) and a pointer to it; one or more folders; keywords; and to-do fields. Most of these can easily be added by choosing from menus, and many have default values. Note the Web pages satisfying the conditions of the query). Next, each document in the result is evaluated by the statistical-learning tool (W3LEARN) to determine whether it consists of a listing of 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). Note that the statistical-learning tool must have been previously trained on a large sample of documents in order to acquire the capability of discerning between documents that contain listings of publications and those that do not. If the document consists of publications, W3LEARN further analyzes the document and adds tags, according to a predetermined format, for indicating author names, titles, etc.

Now the user should activate the data translation (W3TRANS) tool in order to translate each tagged document into a collection of local relations (representing the publications found in that document). W3TRANS is activated through the interface shown in Fig. 5, in which the user is required to specify the formats of the data source and the desired output, and the translation program between those formats. In this particular example, all these three parameters are known in advance and can be selected from the appropriate menus. Note that this is not a coincidence. Once W3LEARN has been trained to tag a document, the set of possible tags is known in advance, and therefore the local relations into which the document is to be translated can also be determined in advance (since null values do not present any problem in our system); similarly, the translation program can also be determined in advance once the data-source and output formats are known.

So, after choosing the appropriate formats and translation program, W3TRANS is activated for the given tagged document, and the output is the URL of a file that contains the new local relations (created from that tagged document). Now, these new local relations have to be added to the global database (W3GDB). To accomplish this task, the form of Fig. 6 is used. In that form, the user specifies the name of the new collection of local relations, and the URL of the file containing those relations. Next, the user has to define rules that map the local relations to the global ones; the form of Fig. 7 is used for that purpose. Some of the rules used in our case are as follows.

$$\begin{aligned}
&glAuthor(AutId, AutName, \_AutLink, \_AutEmail, \_AutPhone, \_AutFax) :- \\
&\qquad\qquad tauAuthors(AutId, AutName). \\
&glPubAutInst(PubId, AutId, InstId, \_GrantNo) :- \\
&\qquad\qquad InstId =' TAU', tauPubAuthor(AutId, PubId).
\end{aligned}$$

Note that variables in the head that do not appear in the body will later be replaced by null values, as described earlier. Since the same local schema is used whenever possible, the rules are almost the same in most (if not all) cases. However, information that is unique to a local source can be inserted through the rules rather than directly into the local relations. For example, the institute ID is inserted through the second rule. The advantage of this approach is that it is much easier to modify the rules for a specific local database than to modify W3TRANS.

After incorporating the new local relations in W3GDB, the original W3GDB query has to be re-submitted, and the result is shown in Fig. 8 as a list of tuples. Note that an attribute that appears without a value has a null value.

6 Cooperative Publications Querying—WGDB, W3LEARN, W3TRANS and W3QS

So far we presented four tools, each aiming at solving one specific Web-related problem (the fifth tool is described in the following section). By combining the tools in different ways, one can build a wide range of applications and overcome many of the limitations in existing systems for utilizing Web data.

To illustrate this, we describe below an application that was built using the tools. The application supports queries about publications of Computer Science researchers. The global database (W3GDB) provides a schema containing information about publications, their titles, authors, etc., and the user can pose queries against that schema. The data for the global database is derived from many local databases, where each local database represents some information source found on the Web. However, the global database does not contain all the relevant information sources that exist on the Web, and part of the application consists of searching for new information sources and incorporating them into the global database. The application user interface is implemented as a sequence of HTML forms. The user interface controls the interaction with the global database, and is also used to connect to the other components of the system (each of those components has its own user interface).

Typically, a user of this particular application starts by posing a query about publications. The query is phrased with respect to the relations of the global database schema. The user interface for phrasing the query is in the style of Query-By-Example. In the first form (Fig. 2) shown to the user, the list of the relations of the global database schema is presented, and the user is asked to check the relations that will participate in the query. The query is the natural join of the relations checked by the user, and in addition to the natural join, the user is also allowed to specify a selection and a projection, as shown in the form of Fig. 3. Note that in that form, the user checks the attributes that appear in the result (i.e., specifies the projection), and also specifies a selection for each attribute. In this particular example, there is one selection, namely `PubArea = 'Databases'`. Once the definition of the query is completed, the query is submitted for execution by clicking on the `Submit` button.

The query is executed by translating it into queries on the local relations (using the appropriate rules stored in W3GDB), and the user can view the result as a set of tuples. Suppose that the result does not contain sufficient information (e.g., the user was looking for publications coauthored by Milo, but could not find any). In this case, the user should activate other tools of the system in order to find new information sources on the Web that may contain the information he/she is looking for. The search for new information sources starts with a W3QL query. One specific query is shown in Fig. 4. This particular query starts a search in a specified page (i.e., specified by its URL) and looks for pages connected to the first one via a single hyperlink that contains the character string `bib`. Moreover, the pages reached in this way should contain the character string `Milo`. Thus, pages are retrieved based on how they are reached as well as their contents. The query (in this example) may seem a bit contrived, but the principle should be clear. The user may combine any condition on the contents of Web pages (e.g., appearance of some keywords in those pages) as well as any condition on the location of those Web pages and how they are connected to other pages on the Web (e.g., search only Web pages that are "close by", i.e. that can be reached by following at most two hyperlinks from some page located on a Web server of a Computer Science Department). Thus, the user is able not just to incorporate knowledge about the contents of the pages he/she is looking for, but also incorporate knowledge on how those pages are interconnected.

The W3QL query is submitted to, and executed by, W3QS. The result is a list of URLs (of

```
n2: PERLCOND 'n2.author.content =~ /Jones/i && n2.author.content =~
/Smith/i';
```

The PERLCOND program extracts the author field of the Latex file and verifies that it contains the strings "Jones" and "Smith".

The resulting query is:

```
Select
From n1,l1,n2
where
n1 in {
http://www.west.edu/datamining/publications.html
};
n1: PERLCOND '(n1.content =~ /Jones/i) &&
             (n1.content =~ /Smith/i)';
n2: PERLCOND ' (n2.format =~ /Latex/i)';
n2: PERLCOND 'n2.author.content =~ /Jones/i && n2.author.content =~
/Smith/i';
using ISEARCHd
```

Finally, we may know a starting point at West University but not the exact location of the Data Mining Group. We can modify the first few lines of the above query to search through the West site. The expression (nx,lx) denotes an *unbounded* path of length 1 or more. The query needs to be further modified if we stipulate that the Data Mining group is accessible from West's starting page with a path of length less than 4. For clarity, this further modification is omitted.

```
Select
From n0,l0,(nx,lx),l1,n1,l2,n2
n0 in {http://www.west.edu};
nx in {/www.west.edu/};
n1: PERLCOND '(n1.content =~ /Jones/i) &&
             (n1.content =~ /Smith/i)';
n2: PERLCOND ' (n2.format =~ /Latex/i)';
n2: PERLCOND 'n2.author.content =~ /Smith/i && n2.author.content =~
/Jones/i';
using ISEARCHd
```

The above are merely simple examples of W3QL's capabilities. For example, not always does one know the search starting points, or cares to specify them. In such cases, one may obtain starting points from an established search service (e.g. Infoseek at www.infoseek.com), and proceed from these points. This entails specifying, within a W3QL query, the search service http address and how the query forms (of the search service) are to be filled. Thus, one can use existing search services as database-like indexes. W3QS offers intuitive interfaces to partially automate query writing.

Among the more important features of W3QL are the ability to specify arbitrary boolean conditions on nodes and links, the ability to use previous form filling activities to navigate through forms (and to prompt the user on encountering new forms), the ability to define (refreshed) views, and the ability to analyze file formats (e.g. checking the author of a Latex file). One may view W3QL as a mechanism for specifying information retrieval agents (or robots). W3QL searches can be more sophisticated than those provided by search services such as AltaVista or Yahoo. Such services may be used from within W3QL queries.

client. The client can either fetch an existing translation program from the server and activate it locally, or write a new program. Once the translation is done, the client asks the server to export the translated data to the appropriate target source. As assistance to the user, both the source and the target data (in their middleware format) can be displayed on a graphic window. Some benefits of the architecture are:

- For security reasons, it is better to grant access to the external sources (part of which are databases), only to one user - which is the W3TRANS server.
- The most extensive computation - the parsing and activation of the TreeLog program, is done at the client. Thus, avoiding a bottleneck at the server.

The architecture also complies with the current Java requirement that all the communication of a Java client (Applet) is done through the server from which the client applet was loaded.

5 W3QL and W3QS

W3QL [24] 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 related material may be found at `http://www.cs.technion.ac.il/~konop/w3qs.html`. W3QS is accessible at `http://www.cs.technion.ac.il/~W3QS`.

W3QL is a SQL-like query language. It enables specifying patterns over the WWW (viewed as a "directed graph") as well as specifying content conditions. Unlike SQL, the W3QL "From List" is a list of paths. Paths may be *unbounded*, i.e. the path's edges may be left unspecified. W3QL allows specifying a limit on the length of unbounded paths. The current W3QL implementation allows only a single path in the "From List".

One can write W3QL queries to specify useful tasks. To get a taste of W3QL's querying abilities, consider the following problem. We'd like to search through a site, say the Data Mining group publications at West University (`http://www.west.edu/datamining/publications.html`) and look for papers in Latex format that are authored by Smith and Jones. Here is our first cut into the problem:

```
Select
From n1,l1,n2
where
n1 in {
http://www.west.edu/datamining/publications.html
};
n1: PERLCOND '(n1.content =~ /Jones/i) &&
             (n1.content =~ /Smith/i)';
n2: PERLCOND ' (n2.format =~ /Latex/i)';
using ISEARCHd
```

This specifies a pattern consisting of page `n1` pointing via link `l1` to page `n2`. In addition, `n2` should be a Latex file. Also, page `n1` should contain the strings Jones (`/i` means case independent matching) and Smith. This is not yet what we want, since currently we know that page `n1` contains both Jones and Smith, but there is no guarantee that page `n2` (the Latex file) is actually co-authored by them. The following condition ensures co-authoring:

A difference is that we view the children of each vertex as ordered. This is crucial to describe lists, an essential component of the DX formats.

Each data source that is to be exposed to the Web community is expected to provide a mapping to/from the middleware format. The representation of each source inside the middleware is very close to the structure of data in the source, so the implementation of such a mapping is fairly easy. For example, a SGML document can be represented by the parse tree of the document. A relational database can be represented by a tree whose root represents the whole database, and having one child for each relation (labeled with the relation name). These nodes will have children representing the tuples in the relation, and having a child for each of the attributes.

The fact that the data representation is close to the source format means that the representation of the various sources inside the middleware can be different. So the translation scheme consists of 3 phases:

(i) In the first phase, data is imported from the source to the middleware data model to yield a forest-like representation that is hiding details unnecessary for the restructuring (e.g., tags or parsing information).

(ii) Then, the data is translated from one abstract representation to the other.

(iii) Finally, the data is exported to the target.

Fig. 1 displays our translation scheme.

For part (ii) of the translation scheme (translations inside the middleware), we use the *TreeLog* language [3, 29]. It is a rule-based language, with a fixpoint semantics, in the flavor of Datalog. 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.

For lack of space we cannot give here a formal definition of the middleware model and the TreeLog language, nor can we demonstrate how the language is used for defining useful translations. More details and examples of translation programs can be found in [3, 29].

Nevertheless, we would like to point out one significant advantage of the language with respect to updates. Assume some data has been extracted and translated from one format to the other, and then was updated in the target system. To propagate the update back to the source, one needs to know the correspondence among the various pieces of data in the two systems, and also to have an inverse translation, mapping data from the target format to the source. Interestingly it turns out that in most practical cases, translation, correspondences between data elements, and inverse translations, can all be automatically derived from one TreeLog program [3]. This is an important result. It saves on writing different specifications for each sub-component of the data integration task, and also helps in avoiding inconsistent specifications.

4.2 Implementation

The ideas above have been implemented and tested in a prototype system, called *W3TRANS* (see `http://www.math.tau.ac.il/~pinim/w3trans-home.html`). It currently supports the HTML, SGML, RELDB (relational format), and O2 (an object oriented system), data formats.

The system is written in Java and built in a client-server architecture. The server stores a set of common translation programs, and is in charge of communicating with the various data sources for importing/exporting data. When a client wants to perform a translation, it connects to the server, and asks for some data to be imported. The server, on behalf of the client, sends the request to the relevant data source. The data (after being mapped to the middleware format) is then sent to the

```
prize-collecting salesmen,
<LINK>http://www.math.tau.ac.il/~azar/kmst.ps.gz
<JOURNAL>Proceeding of 27th STOC,
<MONTH>
<YEAR>1995
```

The important ingredient of this tool is that it can handle erroneous, noisy, and irregular input. However, it can also make errors in its labeling, classification and segmentation. It is thus important that the other components of the system, or other users of these tools, are robust enough to deal with statistical tagging mistakes.

4 The Translator (W3TRANS)

A primary motivation for new database technology is to provide support for the broad spectrum of multimedia data available on the network. These data are stored under different formats: a specific database vendor format, SGML or Latex (documents), DX formats (scientific data), Step (CAD/CAM data), etc. Their integration is a very active field of research and development, (see for instance, for a very small sample, [9, 6, 7, 10, 8, 18, 31, 32, 1, 11, 2]).

A key observation is that, often, the application programs used by organizations can only handle data of a specific format. (e.g. Web browsers, like Netscape, expect files in HTML format, and relational databases expect relations). To enable a specific tool to manipulate data coming from various sources, a translation phase must take place–the data (in the source format) need be mapped to the format expected by the application.

Currently two kinds of tools are available for mapping from one format into another one: (i) specialized tools that perform a default mapping from one format to another (e.g., from Latex to HTML[39]) and (ii) more general tools allowing the user to specify a mapping (e.g., ICA[27]). The tools of the former kind are easy to use but are very limited since they do not allow the user to customize mappings. In fact, the user does not know *a priori* how the data will be translated [36]. Although more satisfactory, the tools of the latter kind still have many drawbacks: (i) they are very difficult to use, (ii) they often do not keep the correspondences between the various components of the source data and those of the translated data, and they do not provide good support for inverse translations; this makes it very difficult to propagate updates (on the translated data) back to the source, (iii) they do not use a declarative language and thus can hardly be analyzed for optimization or update purposes [17].

The goal is to build a translation tool that will overcome the above shortcomings, and allow easy specification of a mapping via a declarative language.

4.1 System Overview

At the heart of the solution are (1) a middleware data model to which various data sources are mapped, and (2) a declarative language, called TreeLog, for specifying data translation within the middleware model.

The choice of the *middleware* data model is clearly essential—it serves as a common ground for all the data sources, and has to provide a clean abstraction of the different formats in which data are stored. The model we use consists of *ordered labeled trees*. It is simple, yet general enough to capture the essence of the formats we have mentioned. Our model is similar to the one used in [7] and to the OEM model for semi-structured data (see, e.g., [19, 32]). (This is not surprising since the data formats that motivated these works are part of the formats that we intend to support.)

The input file to this tool can be (but is not always only) a HTML file which contains bibliography items, such as:

```
<LI> Synchronization power depends on the register size.  Y. Afek and G. Stupp,
<A HREF="ftp://ftp.math.tau.ac.il/pub/afek/focs93.ps.gz"> In Proceeding of
the 34th Ann IEEE Symp on Foundations of Computer Science, pages 196-205,
November, 1993.

<LI> March, 96.  D. Aiger and D. Cohen-Or, Real-Time
Ultrasound Imaging Simulation,
<A HREF="ftp://ftp.math.tau.ac.il/pub/daniel/ultra.ps.gz">

<LI> N. Alon, Y. Matias and M. Szegedy,
<A HREF="ftp://ftp.math.tau.ac.il/pub/noga/AMS.ps.Z"> The
space complexity of approximating the frequency moments, Proceeding of the
28th ACM STOC, 1996.

<LI> Improved approximation guarantees for minimum-weight
k-trees and prize-collecting salesmen, Proceeding of 27th STOC, 1995,
277-283.  B. Awerbuch, Y. Azar, A. Blum, S. Vempala, <A HREF="
http://www.math.tau.ac.il/~azar/kmst.ps.gz">
```

The output of the segmentation tool on this file portion looks as follows (and is in the form requested by the W3TRANS tool).

```
<BIB>
<AUTHORS> <AUTHOR>Y. Afek <AUTHOR>G. Stupp
<TITLE>Synchronization power depends on the register size.
<LINK>ftp://ftp.math.tau.ac.il/pub/afek/focs93.ps.gz
<JOURNAL>In Proceeding of the 34th Ann IEEE Symp on Foundations of Computer
Science,
<MONTH>November
<YEAR>1993

<BIB>
<AUTHORS> <AUTHOR>D. Aiger <AUTHOR>D. Cohen Or
<TITLE>Real-Time Ultrasound Imaging Simulation,
<LINK>ftp://ftp.math.tau.ac.il/pub/daniel/ultra.ps.gz
<JOURNAL>
<MONTH>March
<YEAR>96

<BIB>
<AUTHORS> <AUTHOR>N. Alon <AUTHOR>Y. Matias <AUTHOR>M. Szegedy
<TITLE>The space complexity of approximating the frequency moments,
<LINK>ftp://ftp.math.tau.ac.il/pub/noga/AMS.ps.Z
<JOURNAL>Proceeding of the 28th ACM STOC,
<MONTH>
<YEAR>1996

<BIB>
<AUTHORS> <AUTHOR>B. Awerbuch <AUTHOR>Y. Azar <AUTHOR>A. Blum <AUTHOR>S.Vempala
<TITLE>Improved approximation guarantees for minimum-weight k-trees and
```

3.1 Statistical file categorization

The first issue addressed by statistical learning techniques is automatic file (URL) categorization, based on initial training data provided by the user. We developed a new algorithm for extracting efficient statistical signatures that enable sequential hypothesis testing of two (or more) file sources [37].

The method is based on automatic identification of a minimal set of strings that provide maximal statistical discriminability of the given categories. In the training phase the algorithm generates a "dictionary" of strings, often but not always words, which are statistically discriminative among the different categories. The discrimination power of each such string is determined by the discriminating information, i.e., the cross-entropy of the string distributions in the two categories,

$$D[p_1|p_2] = \sum_x p_1(x) \log \frac{p_1(x)}{p_2(x)} , \tag{1}$$

where x is a string in the dictionary and $p_i(x)$ its empirical probability in the i-th category. The pairwise measure is extended to several categories by Bayes law.

A small example of such a dictionary is given in the table below, which is a small part of the Computer Science bibliography tagging dictionary. The training complexity amounts to two passes through the training strings, keeping only strings which are sufficiently informative. The cross-entropy, estimated from the training data, also provides estimates of the test size and level of confidence of the categorization.

String	Distribution
(a)	0.004132 0.491736 0.475207 0.004132 0.004132 0.004132 0.004132 0.004132 0.004132 0.004132
(abstract)	0.045455 0.590909 0.045455 0.045455 0.045455 0.045455 0.045455 0.045455 0.045455 0.045455
(acm)	0.007143 0.007143 0.935714 0.007143 0.007143 0.007143 0.007143 0.007143 0.007143 0.007143
(adaptation)	0.357143 0.071429 0.071429 0.071429 0.071429 0.071429 0.071429 0.071429 0.071429 0.071429
(almost)	0.071429 0.357143 0.071429 0.071429 0.071429 0.071429 0.071429 0.071429 0.071429 0.071429
(alcom)	0.023810 0.023810 0.785714 0.023810 0.023810 0.023810 0.023810 0.023810 0.023810 0.023810

The above table shows a short segment of the tagging dictionary. Each column corresponds to a specific category. Each string is shown with its empirical probability in the different tagging categories.

3.2 Automatic statistical tagging

The second technique we developed is automatic segmentation of text into fields with statistically distinguishable signatures, such as bibliography items. The method of the classification is similar to the supervised hypothesis testing technique of the previous section, with additional information that comes from the temporal order of the items and from knowledge of the possible field separators. 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. Here we use, for the first time, the technique of hidden Markov models (HMM) for this task, 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.

2.4 A Brief Comparison with Related Work

Our approach is similar to that of the Information Manifold [25] 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 [19] and HERMES [38]) concentrate on translating queries directly rather than by mapping local relations to global ones. But there are some important differences between our approach and the Information Manifold. We can easily handle null values, and therefore, our approach makes it quite easy to use the same local schema for information sources of a similar nature. The local schema does not have to be an exact reflection of the structure of the data found at an information source. Instead it may be just an "upper bound" of what we expect to find there. Null values are filled in automatically when a tuple for a given relation has only partial information. Similarly, the mapping between the local and global relations 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.

A second important difference is in the mapping rules. In our approach, 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. In the Information Manifold, local relations are expressed as *views* of the global relations, and therefore, evaluating a query requires sophisticated algorithms for view usability [26] that are either inefficient (i.e., exponential) or nonexistent in the general case. The Information Manifold uses a heuristic approach to solve the view-usability problem.

3 Statistical Categorization and Tagging (W3LEARN)

This tool 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.

A database query is usually formulated in terms of concepts (e.g., relation names or class names) with a precise formal meaning that is not fully apparent to a casual user. Moreover, the terminology and dictionary used are often very specific to a particular database and context. Thus, making effective queries requires an understanding of the concepts and the precise words that describe those concepts to a degree that is beyond the capabilities of a casual, non-expert user. This presents a severe problem when trying to build a query language for the large and diversified amounts of data on the Internet.

So far, there has not been much work on the applicability of the emerging technology of machine learning to this problem. What seems to be needed is moving, as much as possible, of the learning of concepts from the user to the machine. The methodology we apply to this task is that of statistical pattern recognition, or 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. Much is known by now about the quality of this approximation for many rules and hypotheses. In particular, under certain technical conditions we can guarantee good performance (low classification error) *outside* of the training sample, on arbitrary data taken from the same distribution as the training data. This essential property in machine learning is known as *generalization*. We have so far applied the statistical learning techniques to two generic and fundamental problems: automatic file (URL) categorization, and statistical segmentation (tagging) into semantic/syntactic fields. While these are clearly general and very common problems, we have demonstrated our solution on classification and tagging of bibliography URL's, see Section 6.

or different information source).

$$\begin{aligned} g_1(X,Z) &\ :-\ \ l(X,Z,Y). \\ g_2(Z,Y) &\ :-\ \ l(X,Z,Y). \end{aligned}$$

Given the above rules, the problem is how to represent a tuple of l that has only partial information. As an example, suppose that some tuple of l has values for the first and third columns, but not for the second one. We can handle missing information by means of null values that are expressed as logical terms (similarly to the way of defining logical object id's in F-logic [22]). So, for example, $l(a, f_2(l(a, \mathit{null}, b)), b)$ is a tuple of l with the values a and b in the first and third columns, respectively, and the null value $f_2(l(a, \mathit{null}, b))$ in the second column. Note that the term representing the null value is generated by applying the functor f_2 to the tuple itself, and f_2 is a functor that is associated exclusively with the second column of l. This guarantees the uniqueness of null values (note that *null* is considered a constant symbol), and assures that, when a query is evaluated, different null values are not "accidently" equated to yield superfluous tuples that should not be in the answer to the query.

A second feature of our approach is the ability to map some information into the global relations by encoding it directly into the mapping rules, without having to insert it first into the local relations. Typically, we use this feature to insert information that is source dependent. For example, we may want to have for each publication an attribute *Institute* that specifies the institute in which that publication was found. The information about the institute's name is not written explicitly in listings of publications; rather, it is found in a URL or in some other Web page. Therefore, the easiest way to do it, in our system, is through the mapping rules, as in the following example.

$$g_1(X, Z, Y, \mathit{Institute}) \ :-\ \ l(X, Z, Y),\ \mathit{Institute} = \text{`Tel-Aviv University'}.$$

Inserting information directly in the mapping rules enables us to apply the same translation program of W3TRANS for generating l, regardless of the institute in which the information source was found.

2.2 Experience with the System

The above approach has proven to be very useful in the application of querying publications (which is described in Section 6). We defined one local schema that contains all the information we expect to find about publications. Therefore, the data translation tool (W3TRANS) had to be programmed only once (i.e., 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).

2.3 Implementation

The global-database tool (W3GDB) is implemented in Sicstus Prolog as a server that is available to clients via the TCP/IP protocol. Currently, local relations are transferred to W3GDB as text files, and they are not maintained to reflect updates in the information sources. The special terms denoting null values are added by W3GDB, and not by W3TRANS.

It should be emphasized that W3GDB is not restricted just to the approach described above. It can also incorporate the approach based on view usability, as done in the Information Manifold. In fact, a hybrid approach that combines the two (i.e., view usability and the one described above) is also possible. Similarly, W3GDB can also incorporate algorithms that take into account binding patterns [35].

find new information sources and translate their data into appropriate local databases.

Thus, there is a single conceptual schema that is used to formulate queries, but there are many different local databases that are needed to answer those queries. One task of the W3GDB tool is to translate queries on the global database into queries on the local databases and evaluate them. But W3GDB also serves as a natural entry point into the system and as a coordinator that invokes other tools when needed. The reason is that using the global relations to formulate a query is the easiest way of finding information, and only if this approach does not yield the desired information, should the user start the more elaborate process of looking for additional information sources, translating them into local databases, mapping those databases into the global one and, finally, resubmitting the query to W3GDB in order to get a more accurate answer to his/her query.

In summary, the functionalities provided by W3GDB are as follows. First, the ability to formulate queries against the global relations (i.e., the conceptual schema). Second, an evaluation of those queries with respect to the local relations representing the various information sources. Third, the ability to define a local database for a new information source; that is, define the local relations of that local database and, by invoking other tools of the system, create actual data for those relations from the information source. Fourth, the ability to define the mapping between the new local relations and the global relations.

Providing the above functionalities, while at the same time cooperating smoothly and easily with the other tools in a way that covers all aspects of *querying and integrating* heterogeneous information sources, is a source of some conflicting requirements. For one, the whole idea behind the two-level approach of local and global relations is to offer maximum flexibility in describing information sources vs. the rather rigid and coherent global schema that is more convenient for a casual user. But if this flexibility is used liberally, it may mean that a lot of effort would have to be invested into translating information sources into local relations. The reason is that in the W3TRANS tool, a separate translation program would have to be written for each output format. In our case, an instance of an output format is just an instance of a local schema (i.e., a schema of a local database). So, if there is a lot of variations among the local schemas, then many different translation programs would have to be written. Similarly, for each local database, there is a need to define a mapping between the local relations and the global ones. That mapping depends on the local schema. So, once again, many different mappings would have to be defined if there is a lot of variations among the local schemas. This is not just time consuming, but also error prone. In what follows we discuss remedies for these problems.

2.1 Mapping Local Relations to Global Relations

Our approach to the definitions of local relations and their mappings to the global relations is characterized by the ability to reuse those definitions from one information source to another when those sources are of a similar nature (e.g., contain listings of publications). In that respect, our approach is different, for example, from the approach of the pioneering Information Manifold work [25] that advocates local relations that fit as much as possible the structure of the data found at the information source. In particular, in the Information Manifold, local relations should be defined so that they will not have null values. For W3GDB, we have developed an approach that permits null values, and consequently, there is more flexibility in using the same local schema for different information sources.

We use Datalog rules in order to map local relations to the global ones. For example, the following rules define the global relations g_1 and g_2 in terms of the local relation l; note that there may be additional rules for g_1 and g_2 that define them in terms of other local relations (of the same the required format. This mechanism is more general than most of the special-purpose wrappers developed for specific Web applications [19, 23].

To illustrate the advantages of such a combined system, we present in Section 6 an integration scenario showing how the first four tools could be used to build a global database of publications. A user may ask relational queries about publications through W3GDB, and he/she may also search the Web for new information sources about publications and add those sources to the global database. Ideally, the process of searching for new information sources and adding them to the global database should be completely automatic. In reality, a considerable amount of human intervention is needed in all systems proposed thus far for the task of searching for and integrating heterogeneous information sources. We believe that our approach drastically reduces the amount of human intervention needed for that task, and that reduction is achieved through several means.

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. In Section 5, we give some examples of concrete W3QL queries that could be used to search the Web for lists of publications. W3LEARN serves as a completely automatic filter that determines whether a given file consists of a list of publications. 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. Our approach for creating wrappers is both novel and powerful. It is based on a combination of W3LEARN and W3TRANS, and it provides the ability to use the same wrapper for two information sources that are statistically similar rather than precisely similar (i.e., have the same format). Thus, the number of different wrappers that a user has to write is drastically reduced.

After building a wrapper for a given information source, the user has to map the output of that wrapper into the global database. Our approach for doing that is described in Section 2, where we explain why our approach makes it easy to take mapping rules that were written for one information source and use those rules, with only minor modifications, for another information source.

The rest of this paper is organized as follows. In the next 4 sections, we briefly describe each of first 4 tools, focusing on the functionality and novelty of each tool. Then in Section 6 we illustrate how the four tools can be combined together to construct useful Web applications, and provide a detailed example of such a construction. Section 7 then describes the fifth tool and explains how it can also be integrated with the rest. Finally, the last section presents our conclusions and future research directions.

2 The Global Database (W3GDB)

The global-database tool (W3GDB) provides a single, unified view of Web-resident data. Naturally, the global database cannot encompass all the data on the Web, and it is more realistic to assume the existence of a global database for each coherent slice of the Web. The application described in Section 6, for example, has a global database with information about publications. The purpose of the global database is to provide the capability of posing queries as if the user has to deal with a *single* database system and not with a myriad of information sources all over the Web. The W3GDB tool provides a conceptual schema, cast in the relational model, and the user can formulate any relational query using the relations of the conceptual schema (also referred to as the *global relations*). However, the global relations are *not* materialized. Instead, each *information source* on the Web is represented as a separate *local database,* and the relations of that local database, called *local relations,* are the ones that are actually materialized. In fact, other tools of our system are used to

- Due to the huge amount of data, it is very difficult to locate relevant information, and in particular to focus on the more "reliable" ones.
- The user is exposed to a large variety of data ranging from Web pages and files to databases and e-mail messages, and may want to utilize this variety of information in many different ways. This causes a big data organization problem, for which existing Web tools give very little support.

Every developer of a Web application has to deal with, at least, some of the above issues [4, 9, 6, 7, 10, 8, 18, 31, 32, 1, 11, 2]. When building a specific application, one often cannot afford to provide a fully general solution for the Web problems being encountered, but rather settle for solving the special aspect for the task at hand. Of course, in this case, it is unlikely that the specific solution can be reused when developing other applications for some other aspects of the problem.

In this work, we took a different approach. Rather than focusing on building one specific Web application, we focused on building a *set of tools*, where each tool aims at solving one *typical* WWW problem. Each tool can be used in a stand-alone mode, in combination with the other tools, or even in conjunction with other systems. Specific Web applications can then be built by using a "mix and match" approach, by combining some of the tools together.

The contribution of the work is two fold. First, each tool being developed is a state of the art tool in its area, providing novel features and functionalities. Second, we show that the combination of these general tools enables us to build powerful complex Web systems in a very flexible way. The five tools developed in this work are the following (the order of tool listing corresponds to the problem listing above):

- A global database server, named W3GDB, that provides a global database view of Web data where data from various sources is integrated. It allows users to formulate queries against the integrated Web data in a standard database style.
- A statistical-learning mechanism, called W3LEARN, for classifying, analyzing and tagging Web data.
- A data translator, named W3TRANS, that supports data translations among a broad spectrum of both standard and ad-hoc data formats.
- A query engine, called W3QS, and a query language, called W3QL, for querying the Web and focusing on significant data.
- A personal information manager, PIM, that one can use to manage the data collected from the Web as well as from other sources.

By combining these general tools in different ways, one can build a wide range of applications and overcome many of the limitations in existing systems for utilizing Web data.

For example, the Web query engine W3QS can utilize the capabilities of the statistical-learning classification mechanism (W3LEARN), and thus provide better data filtering and selection than most Web search engines (e.g. [40, 16, 28]). This powerful combined search engine can, in turn, be used by the global database server, W3GDB, to locate the data sources needed by the global database server, and thus provide more reliable information.

As another example, a powerful wrapper [19, 23] can be obtained by combining the data translator and the statistical-learning mechanism. The statistical-learning mechanism can be used to tag significant components of HTML documents, and then the translator can map this tagged data into

Contact Author: Oded Shmueli
email: oshmu@CS.Technion.AC.IL

WebSuite—A Tool Suite For Harnessing Web Data

Catriel Beeri, Gershon Elber, Tova Milo, Yehoshua Sagiv, Oded Shmueli, Naftali Tishby,
Yakov Kogan, David Konopnicki, Pini Mogilevski, Noam Slonim

All Figures Appear at the End.

Abstract

We present a system for searching, collecting, and integrating Web-resident data. The system consists of five tools, where each tool provides 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 wide range of capabilities that overcome many of the limitations in existing systems for harnessing Web data. The paper describes each tool, possible ways of combining the tools, and the architecture of the combined system.

1 Introduction

The *Internet*, and in particular the *World-Wide Web* (WWW), provides a readily-available domain of heterogeneous information sources. WWW technologies, including Web browsers and index servers, facilitate a hypertext based navigation through the Internet as well as the ability to search and access different sorts of data. The rapid growth of the Web and the large amount of information accessible on it, make it a very appealing platform for the development of new applications. However, there are many problems one encounters when trying to utilize Web data.

- When finding some potentially interesting data, it is difficult to verify (automatically) that the data is indeed of the required type, and to extract the relevant pieces of information from it. This is highly influenced by the loose structure of information on the Web. Documents are often only weakly structured using HTML tags, and the inner logical organization is user dependent. For example, researchers have distinct ways for formatting bibliographical information in their homepages. Thus, automatic extraction of, say, article titles, becomes a difficult task.

- There may be several sources containing significant information on some specific topic. The sources may contain complementary data, have some overlapping information, or may even have contradictory information. So, not only locating the significant data sources is a problem, but also the utilization and integration of these sources in an appropriate way is a major obstacle.

- Information sources may contain data in different formats. Some data (as in the bibliography example above) may be in some user defined format. Other data may obey more rigid standards like that of a particular database vendor, SGML or Latex (documents), DX formats (scientific data), Step (CAD/CAM data), etc. Applications need to handle these different formats, and often may need to map from a certain format to some other standard or application-dependent format.