

Structured Web Pages Management for Efficient Data Retrieval

D. Taniar
Dept. of Computer Science
RMIT University
Australia
taniar@cs.rmit.edu.au

Y. Jiang
School of Communications
and Informatics
Victoria University, Australia
jiang@matilda.vu.edu.au

J.W. Rahayu, L. Bishay
Dept. of Computer Science
and Computer Engineering
La Trobe University, Australia
{wenny,bishayl}@cs.latrobe.edu.au

Abstract

The widespread use of World Wide Web in recent years has opened a way of universal access to vast amount of information sources. An obstacle that affects the access to Web data is the lack of information structure among and within Web pages. This raises a need for a structured Web pages management for efficient Web information search. Our proposed structured Web pages management is built upon two stages, particularly (i) HTML transformation to XML, and (ii) Navigation Hierarchy. Also, we study how querying Web data can be accomplished to our structure web pages management, in which users may follow a navigation hierarchy to browse both inter-page and intra-page structures of the Web database and specify queries for desired information.

1 Introduction

The widespread use of World Wide Web in recent years has opened a way of universal access to vast amount of data sources. With the growth of Web space, however, locating and retrieving desired information is becoming more and more difficult. This is largely due to the fact that Web information providers, both at organizational and individual levels, are virtually free to structure and organize data as they wish and there is little central control and management of the Web data.

One solution to the problem of information search is to develop more efficient search engines, which have better summarizing and indexing capability [1]. Machine learning techniques are also used to filter out irrelevant information [9]. Another solution that attracts increasing research interest is to introduce some conceptual structure to Web data. It has been realized that much more meaningful search would be possible if at least a partial representation of Web information structure (known as schema in the database lexicon) is available. Web data structuring is first studied as semi-structured databases and then commercially endorsed via the use of XML (eXtensible Markup Language) [5,6]. The recent work on

semi-structured databases has focused on data modeling and developing query languages. Most of the semi-structured data models, such as OEM, are object-oriented; while their query languages are similar to OQL or SQL [12,16]. The work on XML is currently concentrating on tools for extracting data from large XML documents, for translating XML data between different ontology and for integrating XML data from multiple data sources. A number of XML query languages have been suggested but yet to be implemented and evaluated [2,7,8,11].

The aim of this paper is to present a structured Web page management for efficient Web information search. This new approach synthesizes ideas from hyperlinks browsing, XML data schema, and semi-structured database query. Since our approach requires an XML data format, we firstly need to convert any existing HTML pages to XML pages.

Once these pages have been converted, a navigation hierarchy is provided for users to browse both *inter-page* and *intra-page* structures of the Web database management and to specify queries for desired information.

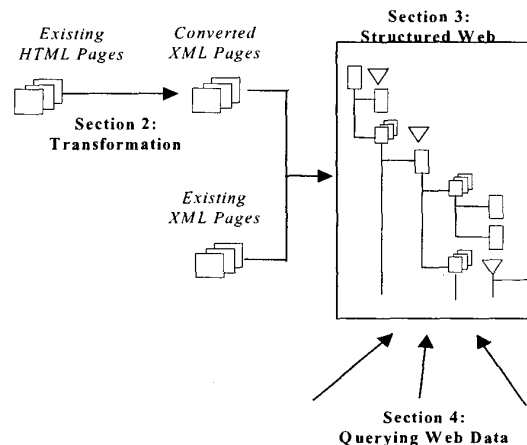


Figure 1. Scope of this paper

Figure 1 gives a graphical illustration on the scope of this paper. Section 2 describes a transformation from HTML to XML, including a semi-automatic case tool for the transformation purposes. Section 3 proposes a navigation hierarchy, where Web pages are related to each other is specified in an inter-page data schema, and the information components within each Web page is declared in an intra-page data schema. Section 4 studies how querying Web data can be carried out. Finally, Section 5 gives the conclusions and explains future work.

2 HTML to XML Transformation

The first stage toward a structured Web pages management and hierarchy is transforming existing HTML pages to XML pages. The transformation comprises three steps, particularly (i) reformatting HTML pages, (ii) deriving Web-schemas, and (iii) using a mapping tool to do the transformation. Certainly such a task of converting these HTML pages manually, will take an enormous amount of time, to the point that it may seem impossible to undertake. Subsequently, there is a great need of semi-automating tool to map their existing HTML pages to XML. For this purpose, we have built a case tool, which takes HTML pages and their Web-schemas and transforms them to XML pages. The process of mapping from HTML pages to XML pages is shown in Figure 2.

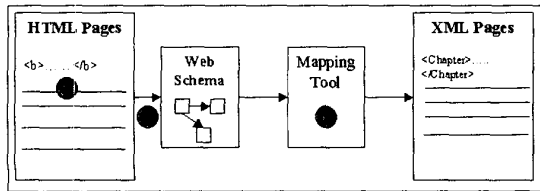


Figure 2. HTML to XML Mapping Process

(i) Step 1: Reformatting HTML Pages

The first step of the process is to assess the HTML pages that will be mapped to ensure that these pages can be mapped smoothly to XML. There are a few requirements, which the HTML files must conform to in order to be mapped to XML and be accepted by the XML Web browser. These requirements are defined by Reding and Vodink [15], which are as follows:

- All elements must have start and end tags,
- All elements must be nested correctly,
- All attribute values must appear with quotation marks, and
- All elements must be self-identifying by ending with ">"

After this step, the HTML page is ready to go through the mapping tool.

(ii) Step 2: Deriving A Web-Schema

The simplicity of this process step highly depends on the structure of the existing HTML site that is being mapped. That is, if the HTML site is in itself well structured and clearly defined then this step will be simple and straightforward. However if there has been numerous administrators creating and modifying the Web site and have not taken into account any guidelines in which the site must abide, then this step will have to 'find the lost schema'.

In order to create efficient XML Web sites the administrator will need to study the entire Web site and extract from it any sort of structure. A way of representing this structure is by sketching a Web-Schema. This notation is used by Mecca, Mendelzon, Merialdo [13] in Efficient Queries over Web Views. These page-schemas are not from a forward engineering phase, but rather from a reverse engineering phase. This aims at describing the structure of an existing site. A human designer, possibly the Web site administrator who knows most about the Web site conducts this analysis. There exist a number of tools, which semi-automatically analyse the Web in order to find regular patterns. Hence the second step of the mapping process is creating a Web-schema of the Web site that will be mapped [13].

Figure 3 shows an example of a Web-schema for a University Web Site from the academic point of view. In this example, each Web page is viewed as an object with a set of attributes.

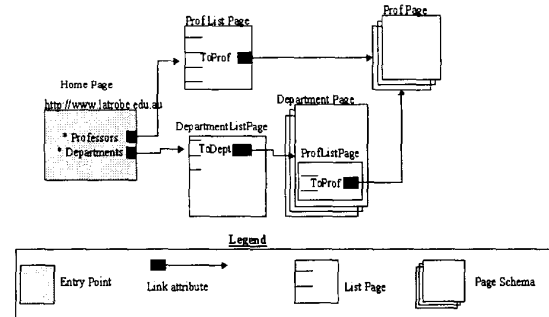


Figure 3. A University's Web Schema

(iii) Step 3: Using the Mapping Tool

The final step in this mapping process is to feed the HTML page into the mapping tool and view the output – an XML page. Here users will define the HTML page which needs to be mapped, the tags which need to be mapped. The tool will then go on

processing the HTML file by prompting the user for the XML tag to be included.

One of the outstanding issues with using XML tags is that there is a high chance of the Web page author being inconsistent with the naming conventions used. That is in one case they would enter 'Last_Name' as the tag name and the next time they would enter 'Surname' for the same type of tag. This would be greatly inefficient for a Search Engine as it will either return irrelevant results or it will need to be very smart as to pick out same type tags and treat them the same.

The mapping tool used here attempts to overcome this problem at least on a smaller scale where the naming conventions are consistent on the business' Web site level. It attempts to do this by first asking the user (who could be the Web page administrator) to enter in all the possible tags that will be used in the mapping process. This way the user really thinks about what the tag names will be and will not add in two or more names for the same type of tags. This will be used in a pull-down list when mapping.

Then the user is asked to select all the HTML tags they wish to convert to XML. After this, the tool simply reads the HTML file specified and prompts the user when it finds one of the tags required to be

replaced. Here the user simply chooses from a pull-down list of tags the tag name they wish to use. The process is repeated until all tags have been processed. Figure 4 and 5 show the tool in action. In Figure 4, it shows that all LI tags are processed. For each LI tag location, there is provided a combo-box, which contains a list of possible XML tags, which are previously specified by the Web administrator. Users then need to go through each of these and choose a relevant XML tag. After the LI tags are chosen, the next step is the program will display the HTML page section that contains an LI tag. The result of this mapping, we would like to show a section of the XML code, which was generated. This can be seen in Figure 5.

The above process describes the mapping process in which the LI tags are mapped in to XML tags. Users can repeat the same process for all other tags, until all tags have been converted.

The details of this tool can be found in Bishay, Rahayu, and Taniar [3]. In this paper, we just explain briefly how the tool works, as the primary intention of this paper is not on the tool itself, but how the tool fits into the proposed structured Web pages management paradigm.

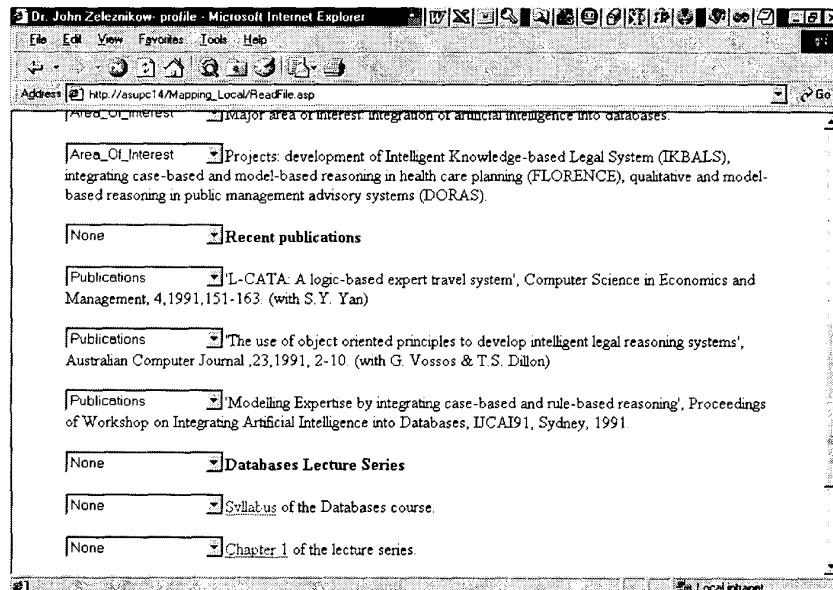


Figure 4. Mapping the HTML with the XML tags

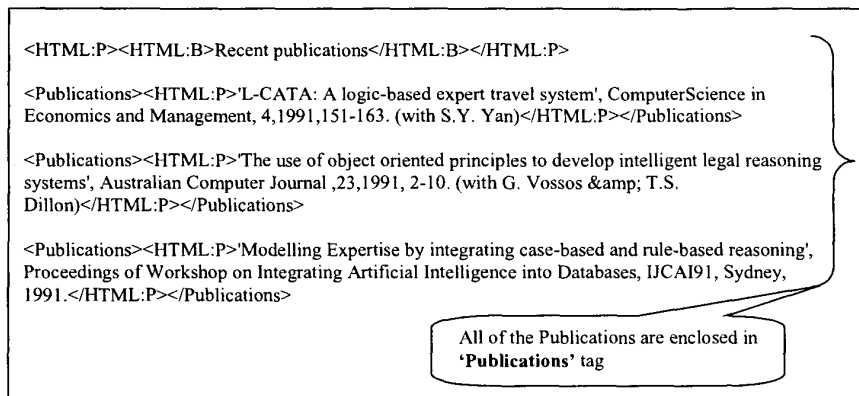


Figure 5. Part of XML code from the HTML page, which was mapped

3 Structured Web Pages Management and Navigation Hierarchy

The proposed structured Web pages database management is designed to provide tools for structuring and managing Web sites of organizations or companies. Since the Web sites at organizational level usually has a controlled point of administration, it is necessary to explore semantic data relationships among individual Web pages and organize them along typical operational hierarchies of the organization [1,10]. Structuring information within each of the Web pages is also important. With a conceptual structure, extracting specific segment of information from a page becomes easy. Moreover, by clustering Web pages with the same or similar structures, a logical data schema, typically loosely-complied, can be formulated and based on that query languages can be used to retrieve Web data. The architecture of the proposed Structured Web Page Management is outlined in Figure 6.

The Web pages stored in the database are XML-enhanced (i.e. XML tags are embedded to specify the information components within the Web pages). In our structured Web pages management, the structure of the information components of a Web page is called an *intra-page schema*. Similarly, the Web pages themselves can be seen as components of the Web site. Therefore, the structure and relationship of the Web pages can also be defined, called *inter-page schema*. The query interface then allows users to browse these schemas through a navigation hierarchy and write queries accordingly. The queries are executed by a query server, producing result Web pages.

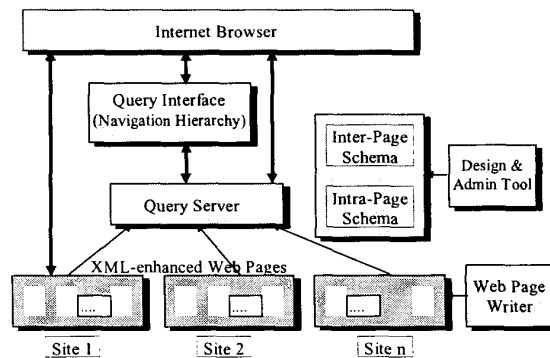


Figure 6. Architecture of Structured Web

3.1 Intra-Page and Inter-Page Data Schema

Web pages are the building blocks in our proposed structured Web page management, which can be any data object in the format, typically *XML*, readable by Internet browsers such as Netscape or Internet Explorer. In order to specify the internal structure of the information stored in the Web pages, however, XML tags must be used. For example, a staff Web page may have any information components defined in the following schema.

```

<Staff>
  <Contact>> </>
  <Teaching> </>
  <Research>
    <Projects> </>
    <Publications> </>
  </>
</>

```

Intra-page schemas are loosely enforced. A Web page does not need to follow a schema or may follow a part of the schema. For example, a staff Web page may have a <Research> component but no further sub-components, i.e., <Projects> and <Publications>, are specified. Nevertheless, this freedom is in a tradeoff with effectiveness of Web data query because the successful queries depend on the degree of uniformity of the information structure.

Besides the XML tags, the objects used in the information segments are not the concerns in page uniformity. In <Contact> segment, for instance, one may have only textual information or additionally, a photograph and a hyperlink to the personal home page. In this context, the freedom of individual Web page construction remains intact.

Inter-page data schema describes how Web pages are logically related to each other. Such a schema can be easily defined using a set of XML tags. Each type of XML tags represents a type of Web pages, while an instance of the tags used in the schema corresponds to an actual Web page. As an example, a schema for a simplified faculty Web site is presented below. The faculty has Web pages for the Dean and several departments. Each department has Web pages for a number of courses offered and for the department staff.

```
<?XML version="1.0" ?>
<faculty: schema>
<Title>SCIENCE</Title>
<Intra_Schema>INTRA_SCIENCE/SCIENCE</Intra_Schema>
<Page>Faculty_Science.xml</Page>
<Dean><Title>DEAN</Title>
  <Intra_Schema></Intra_Schema>
  <Page>Dean_Science.xml</Page>
</DEAN>
<Dept><Title>COMPUTER SCIENCE</Title>
  <Intra_Schema>INTRA_SCIENCE/DEPT</Intra_Schema>
  <Page>Computer.xml</Page>
  <Course><Title>BSC Computer Science</Title>
    <Intra_Schema>INTRA_SCIENCE/BSC-COURSE</Intra_Schema>
    <Page>COMPSCI-BSC.xml</Page></Course>
  <Course><Title>MSC Computer Science</Title>
    <Intra_Schema>INTRA_SCIENCE/MSC-COURSE</Intra_Schema>
    <Page>COMPSCI-MSC.xml</Page></Course>
  . . . . .
  <Staff><Title>HOD</Title>
    <Intra_Schema>INTRA_SCIENCE/STAFF</Intra_Schema>
    <Page>COMPSCI_HOD.xml</Page></Staff>
  <Staff><Title>ADAM</Title>
    <Intra_Schema>INTRA_SCIENCE/STAFF</Intra_Schema>
    <Page>COMPSCI_ADAM.xml</Page></Staff>
```

```
<Staff><Title>BACK</Title>
  <Page>COMPSCI_BACK.xml</Page></Staff>
. . . . .
</Dept>
<Dept><Title>PHYSICS</Title>
. . . . .
</Faculty: schema>
```

In the above schema, each Web page element have three special tags: <Title> being the title of the page, <Intra-Schema> specifying the intra-page schema used and <Page> pointing to the XML file. This information is needed for the navigation hierarchies described later. It may be noted that not every element needs to have an intra-page schema. The Dean's Web page doesn't have an intra-page schema associated with it because it is not necessary. In contrast, staff Back does not specify an intra-page schema although it is available. It indicates that his/her Web page does not follow the standard staff Web page structure. A drawback from lack of an intra-page schema is that the Web page can only be retrieved as a whole and extraction of specific information segments is difficult.




In order to facilitate browsing, all intra-page schemas used by each inter-page schema are usually stored in one file. Since each page element in the inter-page schema has a pointer tag to the corresponding entry in the file, the intra-schema can be easily displayed along with the inter-page schema.

3.2 Navigation Hierarchy and Information Search

Based on the inter-page and intra-page data schemas, a navigation hierarchy can be built for users to search required information on the Web site. The proposed navigation hierarchy is different from existing hyperlink browsing tools and graph-based semi-structured data models such as HyperTree and DataGuide [2,12] because it combines their features together, i.e. it can.

- display inter-page schema of Web pages (also known as logical data view);
- display instances of the Web pages under each schema element (also known as physical data view)
- display intra-page schema which are internal information structure of Web pages;
- support browsing Web pages or part of the pages (page components); and
- enable queries for retrieving specific information.

A navigation hierarchy consists of a set of nodes in a form of tree with a root node representing the top level element of the inter-page schema. There are three types of nodes in the hierarchy as defined below:

- **Web Page Node**, denoted by icon  which represents a Web page. It has a title, a link to the Web page file and optionally a pointer to the intra-page schema. By clicking on the node icon, the nodes at the next level, if exist, will be displayed in the navigation hierarchy.
- **Cluster Node**, denoted by icon  When there are sibling Web page elements in the inter-page schema having the same type of tags, a cluster node is created to represent this set of the pages. Using a cluster node, the actual Web page nodes, which can be many in number, will be hidden from the navigation hierarchy unless users want to look into them. By clicking on the cluster node icon, all the Web page nodes within the cluster are displayed.
- **Page Component Node**, denoted by icon  which represents an information component (tag) in the intra-page schema of the corresponding Web page. By clicking on the icon, the components at the next level, if exist, are displayed.

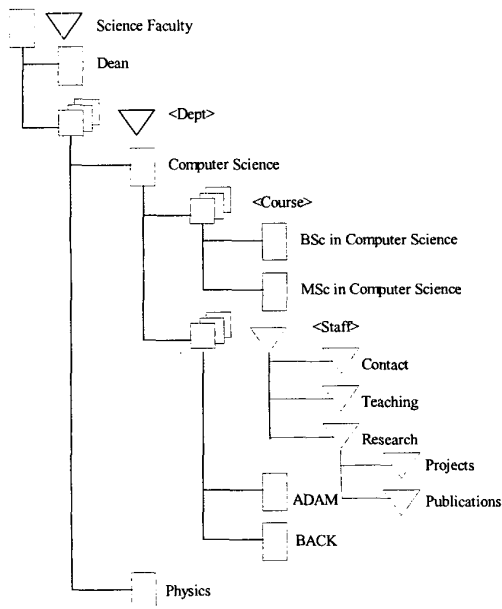


Figure 7. Navigation Hierarchy

The page component icon is also used together with a Web page node or cluster node. In this case, it indicates that this Web page or cluster of Web pages have an intra-page schema available. Clicking on the icon will expand the intra-page schema for display. For sake of simplicity, the page component icon is usually attached only to the

cluster node, assuming all Web pages within the cluster sharing the same intra-page schema. If a Web page does not belong to any cluster, however, a page component icon will be attached to it.

A navigation hierarchy for the example faculty Web site is shown in Figure 7. Initially, only the root node of the navigation hierarchy is shown. By selecting on the root node icon, the nodes at the next level are displayed. Further down traversals are carried out in the same way until the node(s) with required information is found or no further navigation paths are available.

In the example, the Web page node for Dean does not have an intra-page schema icon indicating no such schema is available. The nodes <Dept>, <Course> and <Staff> are cluster nodes with two Web page nodes in each of them. The intra-page schema for staff node is also shown (by clicking on the schema icon).

4 Querying Web Data

Based on the navigation hierarchies, users may explore Web page database in three different ways: (i) Web pages browsing, (ii) Query via navigation hierarchy, and (iii) Query by language. Recall that the interface of the *Web Structure* consists of three sections (frames) which display a list of available navigation hierarchies, the navigation hierarchy selected and the content of the selected Web page or the result of queries.

4.1 Web Pages Browsing

From the displayed navigation hierarchy, users can simply select any Web page node or page component node to bring up the content of the node. The cluster node, however, can not be selected, as it is not associated with actual Web data stored. The general procedure is as follows:

- Choose an appropriate navigation hierarchy to use.
- Double click on relevant node icons to expand and to search down the navigation path(s) until the required Web page or page component is found.
- Single click to select a required Web page node or page component node.
- Press Show button to display the content of the node.

The information surfing via navigation hierarchy is similar to those by hyperlinks within Web pages but much more efficient and fast. By using the navigation hierarchy, one can identify the useful Web pages or even the components of the pages before retrieving them. Moreover, the irrelevant intermediate Web pages can be easily skipped. It should also be pointed out that the

navigation hierarchies are additions to existing Web page surfing because the Web pages and their hyperlinks remain intact. Users may use either physical hyperlinks or logical navigation paths or a combination of them to search the Web page database.

4.2 Query via Navigation Hierarchy

In stead of browsing a single Web page/component at each step, several nodes in the navigation hierarchy can be selected and combined into a new Web page to display. The procedure to do this is the same as the one shown earlier except that

- More than one node in the navigation hierarchy can be selected for display.
- The cluster nodes, which represent a collection of Web pages can also be selected. Selection of a cluster node means that *all* Web pages within the cluster are selected.

The queries via navigation hierarchy can be classified into several types as described below.

(i) *Selecting Web Page Nodes or Page Component Nodes Only*

For example, we may retrieve Web pages for staff ADAM and BACK. By selecting on both ADAM and BACK's nodes, the two Web pages are concatenated together and displayed.

(ii) *Selecting Cluster Nodes Only*

For example, we may retrieve Web pages of all departments in the Science Faculty. This is done by simply clicking on the cluster node <Dept>.

(iii) *Selecting Web Page Nodes plus Their Page Component Nodes*

For example, we may extract the contact information and list of publications from Web pages for staff ADAM and BACK. This is done by selecting Web page nodes for ADAM and BACK followed by clicking on the contact and publications icons from staff intra-page schema.

(iv) *Selecting a Cluster Node plus associated Page Component Nodes*

For example, we may retrieve a full list of publications for all Computer Science staff. This is done by selecting the cluster node for staff and the component node for publications.

Note that selecting a cluster node as well as its associated Web page nodes has ambiguous meanings. It is not clear if all Web pages in the cluster should be chosen or only the ones selected should be used. In this case, we choose to ignore the cluster node selected.

More complex queries can be formulated in the same ways shown above. In fact, a complex query can be most

likely obtained by combining the above types of queries. For example, we may want to retrieve the Web page for Computer Science Department plus a list of contact information of all staff in the Department. This query is obviously a combination of query types (i) and (iv). We need to click the Web page node for Computer Science, the cluster node for staff and the page component node for contact information.

4.3 Query by Language

In our Structured Web Page Management, a query-by-language interface is provided which enables users to write query statements directly. This interface is particularly useful for advanced users to write complex queries in a more explicit and accurate way than using the previous methods. In fact, all the queries generated from query-by-navigation-hierarchy are first translated into query expressions written in an underlying query language, and then are performed accordingly. Since our Structure Web is XML-complied, any XML query languages that have been proposed may be used [2,7,8,11]. We have chosen XML-QL as our underlying query language because of its simplicity and expressive capability [8]. Nevertheless, since the purpose of our queries slightly differs from those upon XML data only, the interpretation of the query syntax is to some extent different as well.

As examples, the queries presented here are written below using XML-QL.

(1) Retrieve Web pages for staff ADAM and BACK who work in Computer Science Department.

```
WHERE <Dept>Computer Science</Dept>
      <Staff> $x </Staff>
CONSTRUCT
  $x
WHERE $x="ADAM" or $x="BACK";
```

(2) Retrieve Web pages of all departments.

```
WHERE <Dept> $x </Dept>
CONSTRUCT
  $x;
```

(3) Retrieve contact information and list of publications from Web pages for staff ADAM and BACK.

```
WHERE <Dept>Computer Science</Dept>
      <Staff> $x </Staff>
      <Contact> $a </>
      <Research>
        <Publications> $b </></>
CONSTRUCT
  $a, $b
WHERE $x="ADAM" or $x="BACK";
```

- (4) Retrieve a full list of publications for all Computer Science staff.

```
WHERE <Dept>Computer Science</Dept>
      <Staff>
        <Research>
          <Publications> $x </></></>
CONSTRUCT
  $x
```

- (5) Retrieve Web page of Computer Science Department plus a list of contact information of all staff in the Department.

```
WHERE <Dept> $x </Dept>
      <Staff>
        <Contact> $y </Contact></Staff>
CONSTRUCT
  $x, $y
WHERE $x = "Computer Science";
```

The above examples are relatively simple queries. By using XML-QL, much more complex queries that involve operations such as joins, sorting and aggregation can be formulated [8]. A full investigation on the query capability and efficient query processing techniques are the future tasks in this project.

5 Conclusions and Future Work

Due to the expansion of the use of the World Wide Web, there needs to be a language, which assists in describing the content of the information at hand, rather than simply the presentation of the information. The new eXtensible Markup Language (XML) was proposed as a replacement of HTML to the WWW Consortium (W3C) in late 1998. After reviewing XML as a markup language, W3C approved the proposal and XML has become the Web's standard markup language.

In this paper, we have presented a tool to map HTML to XML. The transformation methodology consists of three steps: reformatting HTML pages, deriving Web-schema, and using a transformation case tool.

We have also presented the design of a structured Web page management system. The presented system synthesizes ideas from hyperlinks browsing, XML data schema and semi-structured data query. An inter-page data schema is used to specify the logical structure by which Web pages are related to each other, while intra-page schemas are also used to further describe information segments within the Web pages. Based on these schemas, navigation hierarchies are provided for users to browse both inter-page and intra-page structures of the database and specify the queries for desired information. Several query methods based on XML-QL query language are also discussed.

The implementation of the structure Web page management system and its navigational hierarchy is currently under way and some components of the system have been completed. The management of the schemas and navigation hierarchies has yet to be investigated. The issues such as meta-data consistency and integrity will be studied. Complex query processing and query optimisation are important future work.

References

- [1] Aggarwal S., Hung F. and Meng W., "WIRE - A WWW-based information retrieval and extraction system", 1998.
- [2] Arocena, G.O., Mendelzon A.O. and Mihaila G. "WebOQL: restructuring documents, databases and Webs", in *Proc. of Intern. Conf. on Data Eng.*, 1998.
- [3] Bishay, L, Rahayu, J.W., and Taniar, D., "Transformation from HTML to XML: Methodology and Tool", submitted for publication, 2000.
- [4] Bosak, J., *Extensible Markup Language (XML)*, <http://www.w3.org/TR/PR-xml-971208>, 1997.
- [5] Bray T., Paoli J. and Sperberg-McQueen C., (eds.) "Extensible markup language (XML) 1.0, *W3C recommendation*, 1998, <http://www.w3.org/TR/1998>.
- [6] Buneman P., "Semistructured data", in *Prof. of Sixth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, 1997, pp.117-121.
- [7] Ceri, S., et al, "XML-GL: A graphical language for querying and reshaping XML documents", in *Prof. of W3C Query Languages Workshop*, 1998.
- [8] Deutsch A., Fernandez M, Florescu D., Levy A. and Suciu D., "XML-QL: a query language for XML", *Prof. of W3C Query Languages Workshop*, 1998.
- [9] Etzioni O. and D. Weld, "Intelligent agents on the Internet: fact, fiction, and forecast", *IEEE Expert*, August.
- [10] Goldman R., McHugh J. and Widom J., "From semistructured data to XML: migrating the Lore data model and query language", <http://www-db.stanford.edu/lore>, 1998.
- [11] Ishikawa H., Kubota K. and Kanemasa Y., "XQL: a query language for XML data", *Prof. of W3C Query Languages Workshop*, 1998.
- [12] McHugh, J., Abiteboul S., Goldman R., Quass D. and Widom J., "Lore: a database management system for semistructured data", *SIGMOD Record*, 26(3), 1997.
- [13] Mecca, G., Mendelzon, A.O., and Merialdo, P. "Efficient Queries over Web Views", *Advances in Database Techniques - EDBT 1998 - 6th International Conference on Extending Database Techniques*, Valencia, Spain March 23-27 1998 Proceedings, LNCS 1377, Springer Verlag, 1998.
- [14] Mendelzon, A., Mihaila, G. and Milo T. "Querying the World Wide Web", in *International Journal on Digital Libraries*, 1997, pp 54-67.
- [15] Reding, Vodnik, *HTML -complete*, Course Technology 1999.
- [16] Suciu D., "Management of semistructured data", in *SIGMOD Record*, Vol.26, No.4, 1997.
- [17] WC3, *XML FAQ*, <http://www.w3.org/xml>, 1998.