

Singapore Management University Institutional Knowledge at Singapore Management University

Research Collection School Of Information Systems

School of Information Systems

11-2002

A visual tool for building logical data models of websites

Zehua LIU

Wee-Keong NG


Feifei LI

Ee Peng LIM

Singapore Management University, eplim@smu.edu.sg

DOI: <https://doi.org/10.1145/584931.584951>

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

 Part of the [Databases and Information Systems Commons](#), and the [Numerical Analysis and Scientific Computing Commons](#)

Citation

LIU, Zehua; NG, Wee-Keong; LI, Feifei; and LIM, Ee Peng. A visual tool for building logical data models of websites. (2002). *Proceedings of the 4th International Workshop on Web Information and Data Management: WIDM '02, McLean, Virginia, November 08, 2002.* 92-95. Research Collection School Of Information Systems.

Available at: https://ink.library.smu.edu.sg/sis_research/968

This Conference Proceeding Article is brought to you for free and open access by the School of Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email libIR@smu.edu.sg.

A Visual Tool for Building Logical Data Models of Websites

Zehua Liu, Wee Keong Ng, Feifei Li, Ee-Peng Lim
Centre for Advanced Information Systems
School of Computer Engineering
Nanyang Technological University
Singapore, 639798

{aszhliu, awkng, asfli, aseplim}@ntu.edu.sg

ABSTRACT

Information sources over the WWW contain a large amount of data organized according to different interests and values. Thus, it is important that facilities are there to enable users to extract information of interest in a simple and effective manner. To do this, We propose the Wiccap Data Model, an XML data model that maps Web information sources into commonly perceived logical models, so that information can be extracted automatically according to users' interests. To accelerate the creation of data models, we have implemented a visual tool, called the Mapping Wizard, to facilitate and automate the process of producing Wiccap Data Models. Using the tool, the time required to construct a logical data model for a given website is significantly reduced.

Categories and Subject Descriptors

H.4.m [Information Systems]: Miscellaneous

General Terms

Design

Keywords

Web Information Extraction, Web Data Model, Supervised Wrapper Generation

1. INTRODUCTION

The World Wide Web has become such a successful channel in delivering and sharing information that people are getting used to searching the Web as the first resort for information. However, users are often not able to retrieve exact information of interest. The amount of data accessible via the Web is huge and growing rapidly. *Information overloading* has been identified as the problem, as opposed to the previous problem of information unavailability. It is usually a time consuming and tedious process for users to finally get the right information that they are looking for.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WIDM'02, November 4–9, 2002, McLean, Virginia, USA.
Copyright 2002 ACM 1-58113-492-4/02/0011 ...\$5.00.

To address this problem, over the past few years, some *Information Extraction* (IE) systems, mostly in the form of wrapper generation systems, have been developed to automatically extract target information. Lixto [2] provides a sophisticated GUI to allow users to interactively construct wrappers based on the browser view of target webpages. XWrap [6] relies on the HTML DOM tree to perform extraction and employs some pre-defined templates to derive the extraction rules. Other related systems include W4F [8], NoDoSe [1], RoadRunner [3] and Wien [4]. For all these systems, their GUI tools mostly concern about information within a single webpage or a set of webpages with similar structure. Even if facilities to navigate through multiple pages are provided, the ability to modelling the overall website is still limited. The resulting data models do not truly reflect most ordinary users' perception of the website.

In this paper, we introduce a software tool called the Mapping Wizard to help users automate and facilitate the process of generating easy-to-understand data models from information sources. The Mapping Wizard has been developed as part of the WICCAP project [5], which aims to solve the information overloading problem by enabling ordinary users to perform web information extraction in a simple and efficient manner. The data models generated by the Mapping Wizard will be used by other components of the WICCAP system to perform the extraction and presentation of information.

2. WICCAP DATA MODEL

When extracting data from websites, it would be useful to have a logical model representing the target website and to specify the target information based on this model, instead of the actual webpages. The core of the WICCAP system is formed based on this. To accommodate this idea, we have proposed a data model called the WICCAP Data Model (WDM). This section provides a brief overview of WDM. A more detailed discussion can be found in [7].

In most current Web Information Extraction systems, when defining a data model, users usually have to directly specify which portion of a webpage is the target information. The separate pieces of target information are later re-organized in some manner to make them more readable and accessible. The data model produced in this manner is usually restricted to the physical organization of the website and not intuitive to other users, especially ordinary users who have no knowledge of the extraction process.

The WICCAP Data Model views information to be extracted from a different perspective. It relates informa-

tion from a website in terms of a *commonly perceived logical structure* instead of physical directory locations. This is based on the observation that different pieces of information in a website are usually related to one another through a certain logical structure that is hidden from the inter-linking of webpages. This hidden logical structure is usually apparent to most users and when users look at a website, they tend to perceive the information as being organized in this logical structure.

For instance, when a newspaper website is mentioned, users generally think of a site that has a list of sections such as the World News, Local News, and Sports News; each section may have subsections or a list of articles, each of which contains information including the title, the abstraction or summary, the article itself, and perhaps other related articles. This hierarchy of information is the commonly perceived structure of a newspaper website that most users are quite familiar. By modeling websites in this way, most users would be able to understand the resulting data models and easily make use of them to retrieve information.

The WICCAP Data Model consists of two components: the WDM Schema and the Mapping Rule. *WDM Schema* defines the basic data elements and how these elements are organized to form the logical view of websites. It provides a set of basic elements that define the basic data structure and at the same time allows extension to these basic elements to achieve more specific and accurate modeling. *WDM Schema* is also flexible enough to represent a group of websites that share similar logical structure, e.g. a group of online newspaper websites.

A *Mapping Rule* refers to a specific WICCAP Data Model that describes a website's logical structure using the WDM corresponding to the type of that website. Mapping Rules can be viewed as instances of a specific WDM, analogous to the Object-Class relationship in OO concept. In WICCAP, the WDM Schema is defined using a XML Schema and the Mapping Rule is described as a normal XML document. The WDM Schema defines how Mapping Rule should look like, thus, the structure of the logical representation.

Each Mapping Rule includes a set of basic elements organized in a tree structure that reflects the website's logical structure. Each element has a *Mapping* attribute that describes how to reach it in the actual webpages. The Mapping attribute of an element may or may not be based on its parent's. Mapping also allows jumping from one page to another. This makes the logical data model (i.e. the structure of the logical elements) totally independent of the physical layout of webpages and the directory structure of websites.

The process of creating a Mapping Rule of a given website involves two main steps. The first is to construct the basic logical structure by defining the set of basic elements and arranging them in the correct structure. The second is to specify the extraction details, i.e. the Mapping attribute, of each logical node. Both steps can be accomplished by using the tools described in the following section.

3. THE MAPPING WIZARD

The current version of the *Mapping Wizard* is implemented using Visual C++ 6.0. This implementation is available on the Windows Operating System platform (Windows 98/ME/2000/XP). The GUI and the set of assistant tools provided by the Mapping Wizard have been briefly described in [7]. Here we examine in detail how the various assistant

tools help accelerate the processing of generating Mapping Rules.

The assistant tools are the critical components that make the Mapping Wizard practically useful. Due to the fact that we are modeling users' perception of the website's logical structure, it is inevitable to have user intervention. In the Mapping Wizard, rather than to try to provide a single tool, we build a set of tools that address different aspects of the problem. We identify the major bottlenecks that tend to slow down the overall process and develop assistant tools to accelerate them. In this way, the efficiency of the overall process is improved and the time required to generate a Mapping Rule of a given website is greatly reduced.

3.1 Pattern Searching Tool

Pattern Searching is similar to the "Search" function that most text editors provide. It allows the searching of certain text pattern within the HTML source and the web browser view. Searching directly on the Mini Web Browser may be more helpful and straightforward to the users.

This tool may not seem important with the existence of the Pattern Induction Tools, which will be described later. However, the patterns derived by the Pattern Induction Tools may not be optimal in some cases. Users may want to modify and optimize the Mapping properties manually. In this case, the Pattern Searching Tool will be useful to test the manually modified patterns.

3.2 HTML Source View Synchronizer

When users locate information of interest, they usually browse through webpages using the browser view. When the user finds the information on the webpage, he or she would like to see the corresponding HTML source codes of that portion. However, to locate the HTML source of certain part of a webpage is not straightforward, especially when the size of the webpage is large. The *HTML Source View Synchronizer* is implemented to accomplish this task of synchronizing the cursor position of the HTML source with the HTML view, and vice versa. The tool automatically highlights, in the HTML Source Viewer, the HTML source of the current selected portion on the browser view, or the reverse if the current selected position is in the HTML source. The result of the synchronization has been proven to be quite accurate, with rare exceptions where complicated non-HTML techniques are used in webpages.

3.3 Pattern Induction Tools

The *Pattern Induction Tools* are the most useful ones among all the assistant tools. They are built based on the two tools described previously and combined with pattern induction algorithms. These tools are developed to release users from manually figuring out the patterns that enclose the target information. Without these tools, after having identified the information, the user has to use the HTML Source View Synchronizer to locate the HTML source codes to analyse the patterns appearing before and after the target text string, and use the Pattern Searching Tool to confirm that the starting and ending patterns are unique, and to finally record these patterns in the Mapping properties of the logical tree node. All these steps can be done automatically with one click with Pattern Induction Tools.

To derive the Mapping properties, the user highlights the information on the browser view and activates the com-

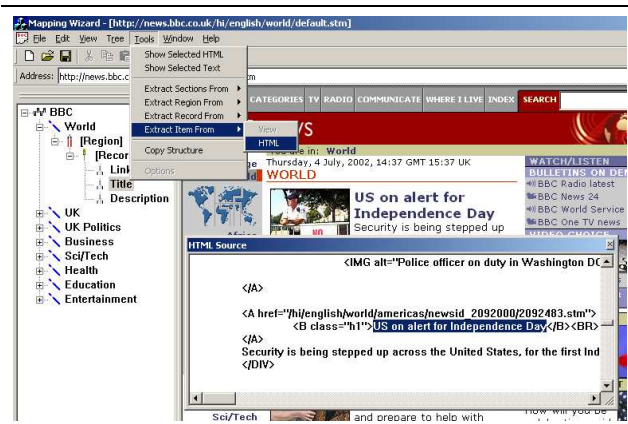


Figure 1: Using the Pattern Induction Tool

mand. In the background, the Mapping Wizard will detect what portion on the browser view is highlighted and invoke the HTML Source View Synchronizer internally to determine the corresponding HTML source. It then applies an algorithm to derive the starting pattern and ending pattern that enclose this part of the HTML source.

The algorithm involves two repeating steps: *generating a candidate* and *testing the generated candidate*. It generates and tests each candidate until a suitable candidate is found. Due to space constraints, we will only illustrate the algorithm with an example. To derive the patterns for an *Item* with type *Link*, i.e. a hyperlink, in the following text string:

```
<A HREF="http://www.cais.ntu.edu.sg:8080/">
```

The final patterns induced are “” for *EndPattern*. The text string located by these patterns is exactly the link that we want to extract.

As pointed out previously, these assistant tools can be used both on the HTML browser view and HTML source. If the user highlights on the browser view, a synchronization between the HTML view and source is performed internally by the Pattern Induction Tool. Due to the highlighting mechanism implemented in the Microsoft Foundation Class (MFC), the user may not be able to select arbitrary portion of the HTML view. This is tolerable for Pattern Induction of rough area, which does not require an exact match and the user can always select a larger area. However, to achieve precise pattern induction for Pattern Induction of exact area, it is usually better to do the highlighting of the target information in the HTML source.

In Figure 1, the user highlights the target string (the title of the news) and activates the command. The induced Mapping will have “<B class=“h1”>” and “” as begin and end patterns.

3.4 Section Extraction Tool

In most Mapping Rules, the first level of tree nodes is usually a list of links pointing to different pages. For example, in BBC Online News, the first level tree nodes are Sections such as “World”, “UK” and “Business”, each of which points to its own webpage. The links of these Sections exist in the homepage of the website and are located close to each other within a small area. The *Section Extraction Tool* is developed to handle this special case by attempting to figure out



Figure 2: Using the Section Extraction Tool



Figure 3: Extracted Section Nodes

all the possible Section nodes in the same level and derive the Mapping properties for them in a single click.

The user highlights the area containing the list of Sections on the browser view and the tool finds out the HTML source using the HTML Source View Synchronizer and applies suitable heuristics, based on common menu layout patterns, to identify the Sections. It then uses a similar induction algorithm to that of Pattern Induction Tools to derive the patterns that locate the links to the Sections’ own webpages.

```
<TABLE>
  <TR><TD><A HREF="worldnews.htm">World</A></TD></TR>
  <TR><TD><A HREF="biznews.htm">Business</A></TD></TR>
</TABLE>
```

For example, a website may have a menu like above. After applying the Section Extraction Tool, two Sections will be identified and inserted into the logical tree in the Logical Tree Editor automatically. These two Sections will have Name attribute of “World” and “Business” respectively. Their Mapping properties will be a Link containing a Locator that has a BeginPattern of “”. Figure 2 and Figure 3 illustrate the use of the section extraction tool.

3.5 Structure Copier

After all the Section nodes have been automatically inserted, the user has to insert the child nodes under each Section and derive their Mapping properties. This process

Name	Website	Description
BBC Online News	news.bbc.co.uk	News from BBC
Amazon	www.amazon.com	Amazon Books
CIA Fact Book	www.odci.gov/cia/publications/factbook/	Countries start with A
DBLP	www.informatik.uni-trier.de/ley/db/	VLDB Papers

Figure 4: Test-sites used for Mapping Wizard

Name	Nodes Derived	Section Extracted	Nodes Copied	Manually Modified	Time (mins)	Time Modeling	Time Mapping
BBC	4	8	28	4 (9.1%)	27	11 (41%)	8 (30%)
Amazon	5	10	54	8 (10.4%)	41	17 (41%)	19 (46%)
CIA	11	20	238	7 (2.5%)	51	14 (27%)	29 (57%)
DBLP	12	9	96	6 (4.9%)	44	12 (27%)	20 (45%)

Figure 5: Evaluation of Mapping Wizard

could very slow and tedious if a website has ten or more Sections at the first level. Fortunately, in most websites, the structure of each Section is nearly identical, because they are normally generated by the same back-end engine with the same formatting styles. To make use of this fact to further accelerate the process, the Mapping Wizard provides another tool, the *Structure Copier*. The basic idea is to copy the structure of the constructed Section to other empty Section nodes, hoping that the same sub-tree hierarchy will fit into those Sections with little modification. It can be seen as a special case of “Copy and Paste”, with a single recursive “Copy” and a batch “Paste”.

3.6 Combining All Tools

Using all the assistant tools together to create a Mapping Rule, the user first uses Section Extraction Tool to generate all the Sections, uses the Pattern Induction Tools to identify sub-tree nodes under the first Section, then activates the Structure Copier to copy the structure of the first Section to other Sections. After some modification on the rest of the Sections, a Mapping Rule is generated.

4. EXPERIMENTAL RESULTS

We have chosen four example websites (Figure 4) (representing four very different categories) to conduct a preliminary evaluation of the Mapping Wizard. Most of the sites have been used for testing purposes by other similar systems. Several users (with 1 to 2 years experience of HTML and some knowledge of XML) were asked to build WICCAP Data Model for each website. Figure 5 summarizes the key statistics of the experimental results. Users are required to record the number of tree nodes that are produced by the various tools and that are manually modified. Time is recorded as well for constructing the logical model and specifying the mapping detail.

From the result, we noticed that, except Amazon, all the three other websites required less than 10% of *manual modification* to the automatically generated models. This figure varies according to the number of nodes copied using the Structure Copier. For CIA Fact Book, as the page for each country has identical structure, simple structure copying worked very well and resulted in the lowest manual modification. The 10.4% manual effort required for Amazon was partly due to the fact that the users were required to model only 10 book categories. If all (35) categories were modelled, this figure will be significantly reduced (estimated 3.2%).

The total time required to construct a data model for each website is within one hour. One of the most recent system, Lixto [2], requires about the same amount of time. However, the time required depends very much on the granularity of

the wrapper constructed (i.e. depth of the tree and number of nodes to be extracted). In addition, the WICCAP Data Model deals with the whole website while Lixto seems to focus on a single webpage in its experiment. To form a view of the whole website, other tools would require extra amount of time to construct multiple wrappers and combine them together into an integrated one.

We also saw a relatively consistent amount of time (from 11 to 17 minutes) spent on figuring out and constructing the logical skeleton of the websites. The time for specifying the mapping details varied depending on the granularity of the data model. The CIA Fact Book required the longest time because there is a huge amount of information for each country, which leads to a large number of nodes.

5. CONCLUSIONS

In this paper, we introduce a new XML data model for mapping websites from their physical structures to logical views. A software tool has been implemented to automate the process of generating a data model. A set of visual tools have been provided to address the bottlenecks of the data model generation process. By combining all the tools, the overall process can be significantly accelerated.

6. REFERENCES

- [1] B. Adelberg. NoDoSE - A Tool for Semi-Automatically Extracting Semi-Structured Data from Text Documents. In *Proc of ACM SIGMOD Conf*, Jun 1998.
- [2] R. Baumgartner, S. Flesca, and G. Gottlob. Visual Web Information Extraction with Lixto. In *Proc of the VLDB Conference*, Sep 2001.
- [3] V. Crescenzi, G. Mecca, and P. Merialdo. RoadRunner: Towards Automatic Data Extraction from Large Web Sites. In *Proc of the VLDB Conference*, Sep 2001.
- [4] N. Kushmerick. Wrapper induction: Efficiency and expressiveness. In *AAAI-98 Workshop on AI and Information Integration*, Jul 1998.
- [5] F. Li, Z. Liu, Y. Huang, and W. K. Ng. An Information Concierge for the Web. In *Proc of DEXA Workshop*, Sep 2001.
- [6] L. Liu, C. Pu, and W. Han. XWRAP: An XML-Enabled Wrapper Construction System for Web Information Sources. In *Proc of ICDE*, Feb 2000.
- [7] Z. Liu, F. Li, and W. K. Ng. Wiccap Data Model: Mapping Physical Websites to Logical Views. In *Proc of the 21st Int'l Conf on Conceptual Modelling (ER2002)*, Oct 2002.
- [8] A. Sahuguet and F. Azavant. Building intelligent Web applications using lightweight wrappers. *Data and Knowledge Engineering*, 36(3):283–316, 2001.