Wiccap Data Model: Mapping Physical Websites to Logical Views

Zehua Liu, Feifei Li, and Wee Keong Ng

Centre for Advanced Information Systems, School of Computer Engineering
Nanyang Technological University, Singapore, 639798, SINGAPORE
`wkn@acm.org`

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 interests in a simple and effective manner. To do this, information from the Web sources need to be extracted automatically according to users' interests. However, the extraction of information requires in-depth knowledge of relevant technologies and the extraction process is slow, tedious and difficult for ordinary users. We propose the Wiccap Data Model, an XML data model that maps Web information sources into commonly perceived logical models. Based on this data model, ordinary users are able to extract information easily and efficiently. To accelerate the creation of data models, we also define a formal process for creating such data model and have implemented a software tool to facilitate and automate the process of producing Wiccap Data Models.

1 Introduction

1.1 Motivation

In the past decade, the World Wide Web has completely reshaped the Internet and led to a wide range of applications and services that are available to people surfing the Internet. Net-surfers do not face the problem of information unavailability. Instead, *information overloading* is becoming a bottleneck for obtaining information from the Internet. People are not able to retrieve exact information of interest. Normal Internet search engines usually return massive irrelevant information along with some useful ones. Even in a given website, useful information is normally split into pieces and is highly coupled with other unrelated data such as advertisements and formatting styles. It's a time consuming and tedious process for users to finally get the right information that they are looking for. In addition, it is not easy for applications to access information from the web, because websites are designed specifically for human browsing.

One of the solutions to this information overloading problem is *Information Extraction* (IE) from the Web. Over the past couple of years, some IE systems, mostly wrappers and software agents, have been built to automatically retrieve information from various sources and extract only those that are of the users' interests.

However, the extraction of information requires a process of specifying data models or extraction rules that define the mapping between the actual information on the Web pages and the pieces that the users are interested. This process usually requires *in-depth knowledge* of relevant technologies, which ordinary users do not possess. As a result, ordinary users who do not possess such technical knowledge are prevented from using IE systems or are limited to use wrapper programs created by other people which may not extract the exact information that the users want.

Another common problem that most Information Extraction systems face is the *slow speed* at which the data models are created. The rate at which new websites are appearing and old websites are changing is much faster than what the current IE systems can handle. Thus, to have a tool to help users to quickly create the required data model for extracting information has become the top priority of IE system for the Web.

1.2 The WICCAP Approach

In this paper, we propose a data model, called the Wiccap Data Model (WDM), to map information from the Web into commonly perceived organization of logical concepts. The aim of WDM is to enable ordinary users to easily understand the data models and use them to perform extraction tasks without worrying about technical details. WDM is designed as part of the *WWW Information Collection, Collaging and Programming* (WICCAP) system [11], which is a Web-based information extraction system to enable people to obtain information of interest in a simple and efficient manner.

To allow ordinary users to perform information extraction, the WICCAP system explicitly separates the tasks of information modeling and information extraction. In WICCAP, expert users construct the WICCAP Data Model and specify how to extract the information; ordinary users indicate what to extract based on the given WDM and use the tools provided by the system to extract and view the information.

Besides being easy-to-understand and easy-to-use to ordinary users, WDM is designed to be *flexible* enough to work with heterogeneous categories of websites and *open* enough to allow other systems and applications to understand and make use of it.

To cope with the rapid changes of old websites and establishment of new websites, we formalize the process of creating a Wiccap Data Model for a given website to permit as much automation as possible. A software tool, called the Mapping Wizard, has been implemented to assist users to quickly generate the data model of a given website.

1.3 Paper Overview

The remaining sections of this paper are organized as follows. In Section 2, we give the definition of Wiccap Data Model and describe how it can be used to model websites. In Section 3, we define the formal process for creating WDM.

In Section 4, we present a software tool that helps to automate the data model creation process. We discuss some of the related work in Section 5. Finally, some concluding remarks are given in Section 6.

2 Wiccap Data Model

In most current Web Information Extraction systems, when defining data models, users usually have to directly specify which portion of a web page within a website constitutes the target information. The isolated pieces of target information are later re-organized to make them more readable and accessible. As a result, the data model produced is usually not intuitive to other users and is very specific to the particular website.

In WICCAP, we try to derive a logical data model (WDM) of the target website first and then extract information from the website based on this model. The role of the WICCAP Data Model is to relate information from a website in terms of *commonly perceived logical structure*, instead of physical directory locations. The logical structure here refers to people's perception on the organization of contents of a specific type of website.

For example, when a newspaper website, such as BBC Online News [16], is mentioned, people generally think of a site that consists of sections such as World News, or Sports News; each section may have subsections and/or a list of articles, each of which may have a number of attributes such as title, summary, the article itself, and maybe links to other related articles. This hierarchy of information is the commonly perceived structure of a newspaper website, which most users are quite familiar with. If we model and organize the information from a newspaper website in this way, the resulting data model will appear familiar to and be easily accepted by most users. In addition, since the model is applicable to the whole category of websites, e.g. all newspaper websites, the logical structure created can be re-used in the future when creating data model for websites of the same type.

WDM is designed to be flexible such that it can be used to model not only a single webpage, but also a set of webpages. This set of webpages may or may not have similar layout structures in terms of HTML syntax. It could be a small collection of several web pages located in the same directory, the whole website, or pages across several websites, so long as all pages together form an unified logical view.

The final logical view of a website is usually a tree structure. In the case of BBC Online News, as shown in Figure 1, we have a root node, under which there are different sections, such as "World" and "Business". There may be subsections under each section. Each node (Section or Subsection) may have a list of articles, denoted by the node "ArticleList" with a child node "Article". Attributes and contents of the article are represented as child nodes of "Article". The mapping information is stored as attributes of the elements to hide the technical details from ordinary users and to maintain the logical overview of the data model.

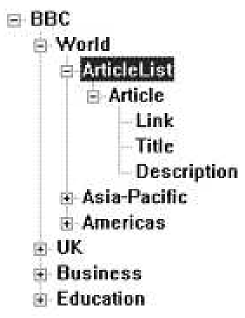


Fig. 1. Logical View of BBC Online News

2.1 WDM Schema and Mapping Rules

The WICCAP Data Model consists of two main components: WDM Schema and Mapping Rule. WDM schema provides the skeletons of the logical view while Mapping Rule offers a detailed definition of the mapping between logical elements and the actual webpages.

A WDM *Schema* concerns not only about a single website but a category of websites. It defines the basic data elements and how these elements are organized to form a logical view of a website. The WICCAP system categorizes websites into different types, such as online newspaper, digital library, and product information. For each category of website, a specific WDM schema is defined. Each WDM schema is defined using a XML Schema. Since a XML schema is itself an XML document, the WDM schema is well structured, interoperable, and easy to process.

A *Mapping Rule* of a website refers to a specific WICCAP Data Model that describes that particular website's logical structure using the WDM schema of the category of that website. A Mapping Rule is defined as a normal XML document, with the corresponding XML schema as its definition.

A Mapping Rule based on a WDM schema could be viewed as an instance of that schema. The relationship between WDM Schema and Mapping Rule is similar to the Class-Object relationship in object-oriented concept. This relationship is reinforced by using the WDM schema as the XML Schema of Mapping Rule's XML document.

2.2 WDM Elements

We define a few *Basic WDM Schema Elements* in WICCAP to form the base of the WICCAP Data Model concept. Additional WDM schema elements that are specific to certain categories of websites can be defined to achieve more accurate modeling of a particular type of website. A WDM schema generally consists of the basic predefined elements and some other extended elements. Each WDM schema describes a category of websites that share the same or similar logical structure. Mapping Rules for specific websites can be created according to the WDM schema of the website.

The WDM schema defines several basic elements, including *Locator*, *Link*, *Form*, *Mapping*, *Item*, *Record*, and *Region*. These basic elements themselves do not have much logical meaning. However, they are essential to WDM schema because they provide the mechanisms for mapping from other elements in the logical data model to the actual Web pages in the website. This seciton describes the basic elements and Section 2.3 illustrates the use of these elements with an example.

Locator A *Locator* is the fundamental element in WDM. It locates a portion within a webpage. The current WICCAP system only implements a ByPattern Locator that relies on the starting pattern and ending pattern to locate a specific portion that resides between these two text patterns. Since HTML pages are merely ASCII texts, with a Locator, we are able to retrieve any part within a given webpage, regardless of whether it is a link, attribute, element name, or text. The definitions of Locator and some of its relevant children are as follows.

```
<xsd:complexType name="LocatorType" content="elementOnly">
    <xsd:choice>
        <xsd:element name="LocatorPattern" type="LocatorPatternType"/>
    </xsd:choice>
    <xsd:attribute name="Type" use="required">
        <xsd:simpleType base="xsd:NMTOKEN">
            <xsd:enumeration value="ByPattern"/>
            <xsd:enumeration value="ByPath"/>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:complexType>
<xsd:complexType name="LocatorPatternType" content="elementOnly">
    <xsd:sequence>
        <xsd:element name="BeginPattern" type="BeginPatternType"/>
        <xsd:element name="EndPattern" type="EndPatternType"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="BeginPatternType" base="xsd:string">

         ⋮   ⋮   ⋮
```

Link A *Link* corresponds to the concept of a hyperlink in the Web context. It gives WDM the ability to following hyperlinks in Web pages; thus, traversing the entire website. When a Link appears in a Mapping Rule, it indicates that there is a hyperlink in the actual web page that should be followed. When the extraction agent follows a Link in a Mapping Rule, it attains the same effect as a user clicking on a hyperlink in a browser, and is brought to the new page pointed by that hyperlink.

A Link can be *Static*, *Dynamic* or a *Form*. A static link is just a simple fixed URL. It is used to indicate the base address of a Mapping Rule. For example, in the root element of the BBC Mapping Rule, we specify a static Link with a value "http://news.bbc.co.uk/". A Dynamic type of Link's link value must be extracted by the extraction agent at run-time from the Web page. The Link element contains a Locator that locates the hyperlink to be extracted. The agent will follow this link to reach another webpage to continue the extraction. This Dynamic type of Link is critical to a flexible and reliable extraction rule because the links in a webpage are likely to change while the positions where these links appear remain relatively stable. A good example of dynamic Link is the links of news headlines that point to the detailed news.

The definition of Link is shown as follows:

```
<xsd:complexType name="LinkType" content="mixed">
    <xsd:choice>
        <xsd:element name="Locator" type="LocatorType"/>
        <xsd:element name="Form" type="FormType"/>
    </xsd:choice>
    <xsd:attribute name="Type" use="required">
        <xsd:simpleType>
            <xsd:restriction base="xsd:NMTOKEN">
                <xsd:enumeration value="Dynamic"/>
                <xsd:enumeration value="Form"/>
                <xsd:enumeration value="Static"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:attribute>
</xsd:complexType>
```

Form *Form* caters to a special group of HTML tags: FORM and other related HTML elements. Form is defined as a type of Link because a FORM ACTION causes the server to perform some operation and return another page; this has the same effect as clicking on a hyperlink. Typical examples where the Form element may be useful are the search function and user login authentication.

When the type of a Link is Form, the URL is dynamically constructed by concatenating all the name and value pairs of the input elements contained in the specified form. The extraction agent will post this URL to the remote web server and obtain the returned webpage. This ability to handling Forms is important for a Web IE System due to the extensive use of Forms in websites. It is also one of the characteristics that distinguish WDM from other similar systems, as currently only few of these systems [2] [17] take Forms into consideration or directly incorporate them into their systems.

Mapping *Locator* allows us to access anywhere within a single page while *Link* enables us to jump from one page to another. *Mapping* combines these two basic elements to allow us to navigate throughout an entire website, or even the whole World Wide Web. It is the element that a Mapping Rule uses to relate the logical element to the actual physical webpage.

A Mapping is a simple container element that holds either a Link, a Locator or both. A Mapping may contain a child Mapping element. This recursive definition allows WDM elements to jump through multiple levels of links and essentially makes it possible to completely separate the logical structure from the physical structure defined by the hierarchy of links and physical directories. Every main element, which is shown in the logical tree view as a node, should have a Mapping as its attribute to indicate how to get to this node in the actual website.

Item The elements introduced before are more for mapping purpose. They are not the main elements that constitute the logical tree. The following three elements, Item, Record and Region, are the logical elements that represent information to be extracted.

An *Item*, which represents a piece of information that the user is interested in, is the basic main element of WDM. It is the atomic unit of all the information in the logical data model. Item has a Mapping attribute that indicates where data is to be extracted, and other attributes that describe its semantic meaning.

Strictly speaking, Item is not a basic WDM element as generally different types of websites will have different Item Types. The definition of Item element varies in different WDM schema. As they have the similar definition and only differ slightly, we treat it as a basic WDM element.

Record In most cases, we do not just specify a single Item to extract in an entire logical tree. Within a website, there is likely to be a lot of Items that the users are interested. Thus, we need to group related Items together, using *Record*.

A Record contains a group of Items, or a tuple of attributes. It is usually used to repeatedly retrieve data in a table or a list when its "Repeative" attribute is 'True'. It makes WDM more flexible and expressive by allowing it to describe iterative structure, which appears frequently in complex logical data model. As with other main elements, there is a Mapping attribute to map the Record to the actual webpages.

An example of Record is the Article element in the BBC News Mapping Rule, with the different attributes of the article, such as Title, Link, Short Description and Content, forming the group of Items under this Record. This article record is iterative since there is a number of articles under each section.

Region A *Region* is defined to help to identify the area where an iterative Record repeats itself. A Region typically contains only one Record if that Record is repeative. It can be considered as the rough area that we are interested within a webpage.

Complete WDM Schema With all the basic elements, we are now ready to define WDM Schemas for different types of websites. The process of producing a WDM schema consists of creating customized elements and assembling these new elements and the basic elements to form an organized structure that reflects the logical view of the website.

New WDM elements have to be defined for the type of website that the user is interested. What elements should be added depends on the specific type of Web site that the user is dealing with. All the basic and customized WDM elements will be organized into a certain hierarchical structure. This hierarchy of elements must reflect the logical structure of the Website that the user perceives or understands.

2.3 An Example on BBC Online News

In this section, we illustrate with an example of how WDM schemas and the Mapping Rules translate physical structure into logical view. The website that we will be looking at is the BBC Online News [16].

```
<!-- WiccapNewsBBC.xml -->
<?xml version="1.0" encoding="UTF-8"?>
<Wiccap Name="BBC" Group="News"
            xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
            xsi:noNamespaceSchemaLocation="WiccapNews.xsd">
    <Mapping><Link Type="Static">http://news.bbc.co.uk</Link></Mapping>
    <Section Name="World">
        <Mapping>
            <Link Type="Dynamic"><Locator Type="ByPattern">
                <LocatorPattern>
                    <BeginPattern>
                        <![CDATA[<TD CLASS="rootsection" BGCOLOR="#FFFFFF"><A HREF="]]>
                    </BeginPattern>
                    <EndPattern><![CDATA[" CLASS="index">]]></EndPattern>
                </LocatorPattern>
            </Locator></Link>
        </Mapping>
        <Region Name="ArticleList">
            <Mapping><Locator Type="ByPattern">
                <LocatorPattern>
                    <BeginPattern><![CDATA[<SPAN CLASS="sectiontitle">]]></BeginPattern>
                    <EndPattern><![CDATA[<SCRIPT LANGUAGE=]]></EndPattern>
                </LocatorPattern>
            </Locator></Mapping>
            <Record Name="Article" NumOfItem="3">
                <Mapping><Locator Type="ByPattern"><LocatorPattern>
                    <BeginPattern><![CDATA[<DIV CLASS="]]></BeginPattern>
                    <EndPattern/>
                </LocatorPattern></Locator></Mapping>
                <Item Type="Link" TagFilter="Off" Description="This is the link to the news">
                    <Mapping><Locator Type="ByPattern"><LocatorPattern>
                        <BeginPattern><![CDATA[<a href="]]></BeginPattern>
                        <EndPattern><![CDATA[">]]></EndPattern>
                    </LocatorPattern></Locator></Mapping>
                </Item>
                <Item Type="Title" Description="This is the title of the news">
                    <Mapping><Locator Type="ByPattern"><LocatorPattern>
                        <BeginPattern><![CDATA[<B class="h***]]></BeginPattern>
                        <EndPattern><![CDATA[</B>]]></EndPattern>
                    </LocatorPattern></Locator></Mapping>
                </Item>
                <Item Type="Description" Description="This is the description of the news">
                    <Mapping><Locator Type="ByPattern"><LocatorPattern>
                        <BeginPattern><![CDATA[</A>]]></BeginPattern>
                        <EndPattern><![CDATA[<BR]]></EndPattern>
                    </LocatorPattern></Locator></Mapping>
                </Item>
            </Record>
        </Region>
        <Section Name="Asia-Pacific">
         :  :  :
        </Section>
        <Section Name="Americas">
         :  :  :
        </Section>
    </Section>
    <Section Name="Business">
     :  :  :
</Wiccap>
```

Fig. 2. Logical View of BBC Online News

The WDM schema for this website is used as an example throughout the paper. Most of the definitions of the basic and customized elements are described in previous sections. This WDM schema simply redefines the Item type and add in two more new elements: Section and Wiccap. Item type is tailored to this specific type of website. Wiccap is the root node of the entire Mapping Rule XML document. Section, as described previously, is the key element in this logical structure because it not only defines a section of a newspaper, but also allows the existence of subsection by using recursive definition. This enables the WDM schema to represent the logical view of most of the online newspaper websites.

The Mapping Rule for BBC Online News, shown in Figure 2, is based on the logical data model established by the WDM schema defined above. Due to the limitation of space, only details of one Section and one subsection of the Mapping Rule are shown.

In the root element, the XML Schema file is specified; this is also the file that contains the definition of the WDM schema for online newspaper websites. The Mapping of the root element Wiccap is a static Link that points to "http://news.bbc.co.uk". This is true for most Mapping Rules because a website needs to have a base URL or home URL for any navigation to start from.

Under the root node "Wiccap", there are a few Sections. In this example, we specify "World", "Business" and so on. Again, the "World" Section has a Mapping, which extracts the link to the page containing the actual "World" section on the website. An ArticleList (Region) is included under the "World" Section to indicate that we are interested in some content in this section. The Article (Record) and Item elements under this ArticleList node give details of the contents to be extracted.

Two Sections "Asia-Pacific" and "Americas" are inserted after the ArticleList. The internal definition of these two sub-Sections are similar to the Section "World". All the Sections and ArticleLists together form a tree structure of the logical view (Figure 1), which is quite close to our common understanding of newspaper: a hierarchy of Sections and Sub-Sections with Articles in each section.

3 WDM Creation Process

As discussed in the introduction section, the manual process of creating data model for Web information sources, including the WICCAP Data Model, is tedious and slow. To allow the WICCAP system to handle large amount of new websites and changing websites, we define a formal process for the generation of Mapping Rule so that it can be made as automatic as possible and have implemented the Mapping Wizard as a tool to help facilitate and automate this process. The formal process of Mapping Rule generation using Mapping Wizard consists of four stages, as illustrated in Figure 3.

Basic Logical Structure Construction To start building a logical view of a website, the user supplies the home URL of the target website to the Mapping

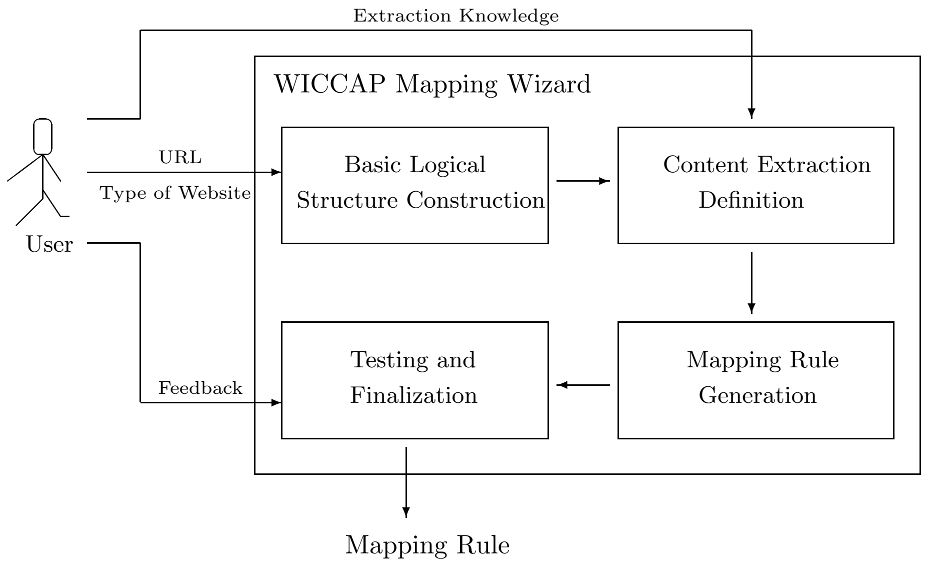


Fig. 3. Formal Process of Mapping Rule Creation.

Wizard and use the Mini Web Browser to navigate the website. The user has to decide on the type of the website and select an appropriate WDM schema either by using his knowledge and understanding of the website or by navigating the website. A logical tree can then be constructed with the understanding of the organization of the contents provided by the website.

In this stage, it is important that the user concentrates on building the overall logical data model instead of details of how to map the logical tree nodes into the actual website. This is because confirming the logical structure first gives the user an overview of the whole website and makes it easy to find the target information that is distributed among a set of webpages.

The process of building up the logical structure of the website is usually quite straightforward. In the case of an online news website, what the user needs to figure out are the sections under the root node and the articles or subsections under those sections. This can be easily done by looking at the index pages of the main sections and subsections. Therefore, the user only needs to insert “World”, “UK”, “Business”, etc, one at a time as the children of the root node, and does the same for any subsection of these sections.

Content Extraction Definition After the basic logical tree is made clear, the user continues with the *Content Extraction Definition* stage, in which he or she spells out the specific information that is of interest and relates the nodes in the logical tree to the actual website.

As described earlier, Mapping includes Link and Locator and eventually points to certain portion of a webpage. To specify the Mapping attributes, it is the same as figuring out the patterns that enclose each piece of information.

Without the help of the Mapping Wizard, the user has to look at the HTML source to decide those patterns are.

The Mapping Wizard is equipped with the ability to figuring out the mapping information with certain manual guidance. This is done by using the assistant tools, which will be discussed in Section 4. The Mapping Wizard also provides a Graphical User Interface (GUI) for the user to modify all the properties to fine-tune and optimize the mapping information.

One point that is very important to the first two stages is that a certain amount of human intervention or feedback is crucial to the Mapping Wizard. Due to the complexity of HTML syntax, we do not believe that there is a single complete algorithm that is capable of fully automating the process of generating the Mapping Rule. What the Mapping Wizard can do is to try to reduce the human intervention as much as possible.

Mapping Rule Generation Once all the information is confirmed, the Mapping Wizard generates the Mapping Rule according to the tree structure and all the corresponding properties. This process is fully automated. The Mapping Wizard validates the information that has been specified and produces the Mapping Rule according to the syntax defined in the WDM schema.

Testing and Finalization In this stage, the user may test the generated Mapping Rule to see whether it represents the correct logical structure and whether it locates all the required information.

The testing is a trial and error process, similar to the process of debugging a program. the Mapping Wizard performs the actual extraction using the generated Mapping Rule. Details of how to perform the extraction using a Mapping Rule is discussed in [10]. If the user is satisfied with all the information retrieved from the website, he or she can finalize the Mapping Rule. Once finalized, the Mapping Rule can either be stored into some repository or be delivered to users of the extraction agent for the extraction of information.

4 Implementation

The current version of the *Mapping Wizard* has been implemented using Visual C++ 6.0. The programming of the Graphical User Interface (GUI) makes use of Microsoft Foundation Classes (MFC). This implementation is available on the Windows Operating System platform (Windows 98/ME/2000/XP). This section briefly discusses the assistant tools provided by Mapping Wizard and its GUI.

4.1 Assistant Tools

Mapping Wizard provides a set of assistant tools to help to automate the process of generating a Mapping Rule. These tools are the critical components that make the Mapping Wizard practically useful. Without the help of these automation tools, the Mapping Wizard is only something like a blind combination of the Internet Explorer, Notepad and an XML editor.

We believe that, in practice, it is not possible to have a single tool that fully automates the whole Mapping Rule generation process because this process is usually complex and requires human intervention from time to time. In the Mapping Wizard, instead of trying to provide a single tool, we build a set of tools that address different aspects of the problem. We identify the main difficulties and bottlenecks that slow down the overall process and develop assistant tools to accelerate those parts one by one. 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. The following gives an overview of the tools provided.

The **Pattern Searching Tool** is similar to the "Search" function that most text editors provide. It allows 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.

The **Pattern Induction Tools** are the most useful ones among all the assistant tools provided. They are developed to release users from manually studying the HTML Source to figure out the patterns that enclose the target information. To derive the Mapping properties using these tools, the user only needs to highlight the information on the Mini Web Browser and activates the command. In the background, the Mapping Wizard detects what portion on the Mini Web Browser is highlighted and determines 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 is developed based on those described in [9].

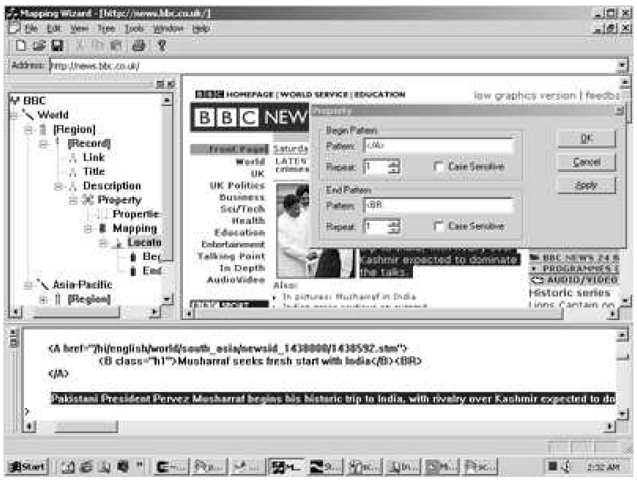
The **Section Extraction Tool** is developed to figure out all the possible Section nodes in the same level and derive the Mapping properties for them in a single click; while the **Structure Copier** is to copy the structure of the constructed Section to other empty Section nodes, hoping that the same Region, Record and Items structure will fit into those Sections with slight modification.

Combining all the assistant tools to create a Mapping Rule, the user first uses Section Extraction Tool to generate all the Sections, uses the Pattern Induction Tools to identify Region, Record and all Items 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.

4.2 Graphical User Interface

Figure 4 shows the main GUI of Mapping Wizard. The **Mini Web Browser** on the right panel is a small web browser for users to navigate the whole website. It has nearly all the functions of a normal web browser. Most assistant tools rely on the Mini Web Browser to perform operations.

The **HTML Source Viewer** displays the HTML source code of the web page that the user is viewing in the Mini Web Browser. It may yield higher accuracy in selecting the desired text string. However, it requires the user to have relevant knowledge, such as HTML and JavaScript.



```
<A href="/hi/english/world/south_asia/newsid_1438000/1438592.stm">
          <B class="h1">Musharraf seeks fresh start with India</B><BR>
</A>

Pakistani President Pervez Musharraf begins his historic trip to India, with rivalry over Kashmir expected to do
```

Fig. 4. Main GUI of Mapping Wizard with Property Dialog

The panel on the left is the **Logical Tree Editor**, which displays the logical model of the website that the user is modeling. The editor allows users to perform normal tree manipulation on the logical tree, including insertion and deletion of tree nodes and modification of their properties, using the **Property Dialog**.

5 Related Work

We compare our work with others related systems in several aspects. In the aspect of information modeling, most existing IE systems, including Tsimmis [6], Wien [9] and W4F [17] do not model information from the logical point of view. The organization of the extracted information has close tie to the physical structure of the website. The data models are schema-less and are difficult to understand. Other systems like Araneus [2] [14] and Stalker [15] build data models that separate the physical and logical structure to certain extent. But their data models still do not truly describe the logical structure that is hidden behind the physical appearance of websites and can not be understood easily by users other than the data model creator. On the contrary, the WICCAP Data Model models information in the way that most users perceive the websites and the resulting logical models are very close to these users' understanding.

In the aspect of data model creation, *manual generation* of data models or wrappers, used in Jedi [7], Tsimmis and Araneus, does not offer GUI support and require the use of programming language like grammar. These systems tend to be very difficult to use by ordinary users. Other systems, including Wien, Stalker and NoDoSe [1], take the *machine learning* approach to learn the extraction rules by examples. Some of these systems provide visual support to allow easy selection and labelling of training examples. However, the requirement for expensive computation power and large number of examples are their drawbacks. A third category of IE systems use *supervised wrapper generation.* Lixto [3], XWrap [12] and W4F provide GUI to interact with users and make use of the HTML DOM tree to locate the information. These systems are closest to WICCAP in terms of the data model generation process. However, they are still not intuitive to ordinary users and require users to have some technical background.

One important distinguishing characteristic of the WICCAP architecture is the separation of the tasks of information modeling and information extraction. This is essentially enable ordinary users to use the system without having special knowledge on Information Extraction.

In addition, systems such as Embley et al. [5], Lixto, RoadRunner [4], Wien and W4F deal with only a single webpage or a set of webpages with similar structure. This greatly limits the ability to modeling a website as a whole. Meanwhile, only few systems (Lixto and Wien) offer the similar functionality as Mapping Wizard to allow users to directly operate on the browser view of webpages instead of the HTML sources.

There are other works that are related in other aspects. Ariadne [8] focuses on modeling and extracting information for further integration and querying. XGrammar [13] takes a similar approach to WDM to use XML Schema to describe data semantics. SiteLang [18] is a language that allows specification of information services based on the concepts of story and interaction spaces.

6 Conclusion

In this paper, we investigated the problems with current Web Information Extraction systems as a solution to the information overloading problem. A new XML data model has been proposed to map websites from their physical structures to logical views. Based this intuitive data model, ordinary users are able to perform information extraction tasks that were previously only possible for people with in-depth technical knowledge. A formal process is presented to standardize the creation of Wiccap Data Model and a software tool has been implemented to automate the process. The process of generating a data model has been significantly accelerated.

References

1. Brad Adelberg. NoDoSE - A Tool for Semi-Automatically Extracting Semi-Structured Data from Text Documents. In *ACM SIGMOD International Conference on Management of Data*, pages 283–294, Seattle, Washington, June 1998.
2. Paolo Atzeni, Giansalvatore Mecca, and Paolo Merialdo. To Weave the Web. In *Proceedings of 23rd International Conference on Very Large Data Bases (VLDB 97)*, pages 206–215, Athens, Greece, August 25-29 1997. Morgan Kaufmann.
3. Robert Baumgartner, Sergio Flesca, and Georg Gottlob. Visual Web Information Extraction with Lixto. In *Proceedings of 27th International Conference on Very Large Data Bases (VLDB 2001)*, pages 119–128, Roma, Italy, September 11-14 2001. Morgan Kaufmann.

4. Valter Crescenzi, Giansalvatore Mecca, and Paolo Merialdo. RoadRunner: Towards Automatic Data Extraction from Large Web Sites. In *Proceedings of 27th International Conference on Very Large Data Bases (VLDB 2001)*, pages 109–118, Roma, Italy, September 11-14 2001. Morgan Kaufmann.
5. David W. Embley, Yiu-Kai Ng, and Li Xu. Recognizing Ontology-Applicable Multiple-Record Web Documents. In *Proceedings of 20th International Conference on Conceptual Modeling (ER 2001)*, Lecture Notes in Computer Science, pages 555–570, Yokohama, Japan, November 27-30 2001. Springer.
6. Joachim Hammer, Héctor García-Molina, Junghoo Cho, Arturo Crespo, and Rohan Aranha. Extracting Semistructured Information from the Web. In *Proceedings of the Workshop on Management of Semistructured Data*, Tucson, Arizona, May 1997.
7. Gerald Huck, Peter Fankhauser, Karl Aberer, and Erich J. Neuhold. Jedi: Extracting and Synthesizing Information from the Web. In *Proceedings of the 3rd IFCIS International Conference on Cooperative Information Systems (CoopIS 98)*, pages 32–43, New York City, New York, USA, August 20-22 1998. IEEE-CS Press.
8. Craig A. Knoblock, Steven Minton, Jose Luis Ambite, Pragnesh Jay Modi Naveen Ashish, Ion Muslea, Andrew G. Philpot, and Sheila Tejada. Modeling Web Sources for Information Integration. In *Proceedings of Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, pages 211–218, Madison, Wisconsin, July 1998.
9. Nicholas Kushmerick. Wrapper induction: Efficiency and expressiveness. In *AAAI-98 Workshop on AI and Information Integration*, pages 15–68, Madison, Wisconsin, July 1998.
10. Feifei Li. Network Extraction Agent for WICCAP System. Technical report, Nanyang Technological University, November 2001.
11. Feifei Li, Zehua Liu, Yangfeng Huang, and Wee Keong Ng. An Information Concierge for the Web. In *Proceedings of the First International Workshop on Internet Bots: Systems and Applications (INBOSA2001), in conjunction with the 12th International Conference on Database and Expert System Applications (DEXA'2001)*, pages 672–676, Munich, Germany, September 3-8 2001.
12. Ling Liu, Calton Pu, and Wei Han. XWRAP: An XML-Enabled Wrapper Construction System for Web Information Sources. In *Proceedings of 16th International Conference on Data Engineering (ICDE 2000)*, pages 611–621, San Diego, California, USA, 28 February - 3 March 2000. IEEE Computer Society.
13. Murali Mani, Dongwon Lee, and Richard R. Muntz. Semantic Data Modeling Using XML Schemas. In *Proceedings of 20th International Conference on Conceptual Modeling (ER 2001)*, Lecture Notes in Computer Science, pages 149–163, Yokohama, Japan, November 27-30 2001. Springer.
14. Giansalvatore Mecca and Paolo Atzeni. Cut and Paste. *Journal of Computer and System Sciences*, 58(3):453–482, 1999.
15. Ion Muslea, Steve Minton, and Craig Knoblock. A Hierarchical Approach to Wrapper Induction. In *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, pages 190–197, Seattle, WA, USA, 1999. ACM Press.
16. BBC Online News. http://news.bbc.co.uk/.
17. Arnaud Sahuguet and Fabien Azavant. Wysiwyg web wrapper factory (w4f). In *Proceedings of World Wide Web Conference*, Orlando, October 1999.
18. Bernhard Thalheim and Antje Dústerhóft. SiteLang: Conceptual Modeling of Internet Sites. In *Proceedings of 20th International Conference on Conceptual Modeling (ER 2001)*, Lecture Notes in Computer Science, pages 179–192, Yokohama, Japan, November 27-30 2001. Springer.