Toward an Ontology-based Web Data Extraction

Hicham Snoussi[1], Laurent Magnin[1] and Jian-Yun Nie[2]

[1] Centre de recherche informatique de Montréal, 550 rue Sherbrooke, suite 100, Montréal, Canada H3A 1B9
{hsnoussi, lmagnin}@crim.ca, http://www.crim.ca/
[2] Université de Montréal, C.P. 6128, succ CENTRE-VILLE Montréal, H3C 3J7 Canada
nie@iro.umontreal.ca

Abstract. Many web sites provide regularly updated data in a fixed structure. These data are very useful for some applications with autonomous agents (e.g. to determine the exchange rates). However, data extraction from these sites is non-trivial because of the great variations from one site to another. In this paper, we propose an approach based on ontology, which facilitates the formalization and the extraction of data from different sources. The extracted data are converted into a coherent structure so that users and agents can query them regardless of their origin. The ultimate goal of this tool is to extract reliable information from web pages.

1 Introduction

The Internet contains more and more Web pages with dynamic and frequently updated data. If a search engine provides useful help for users to identify relevant information, it cannot be used to "understand" the semantics of the results and to obtain reliable data. This is mainly due to the lack of precision and standard formalism in presenting data and because HTML is a formatting language. In addition, current search engines are more focused on static data on the web rather than dynamic data that constantly change, such as weather forecasts, stock exchange information, etc. On the other hand, such data are more and more required by automated processes such as software agents, and the need is growing to find ways to extract data so they can be fully exploited by agents. Our goal is to develop a method to extract reliable data from web pages so that software agents can be used.

1.1 Problem Description

Data on the Web are usually included in HTML pages, and they do not correspond to a known schema. While a human user can understand the data in a page, it is impossible to do so by a machine. Therefore, extracting data from web pages requires knowledge of both their structure and contents. There are mainly three approaches to deal with this problem of data extraction from web pages:

- The first approach relies on natural language processing (NLP). It is known that current NLP is not accurate and powerful enough to recognize the contents of unrestricted web pages. Therefore, this approach has only been used in some limited areas;
- The second approach tries to associate a web page with some semantic markers (or tags) when it is created. For example, one may use personalized markers. The limitations of such an approach are well known: since the markers are personalized, they can hardly be generalized [1]. Currently, an initiative of Semantic Web [17] is geared towards the creation of a web structure that more readily recognizes the semantics of Web pages. The method currently under investigation consists in defining a general ontology of meta data on semantic contents. However, few actual Web pages use such markers;
- There is a third manner to solve the problem: As the original data are structured in different ways, it is more suitable to restructure them according to a common model that is independent of the information sources [9] [10]. Thus, extracting and combining data from different sources will be much easier and more reliable.

Our approach is based on the third idea: we will make use of ontology to model the data to be extracted. In particular, we will focus on data extraction from web pages that present constantly changing data, but with a fixed structure (e.g. stock exchange quotes). The data in a web page is first converted into XML, then mapped with the data model. The definition of the data model and the mapping are done manually. Then an automatic process is carried out to perform the real extraction task. The final result is an XML document that contains a standardized and queriable data set.

1.2 Some Related Work on Data Extraction

The work of [10] on WMA (Web Mining Agent) and that of [9] (project Ariadne) are close to our approach. Ariadne has a graphical interface to assist the users in selecting the data to be extracted from the Web pages [9]. Ariadne uses techniques of machine learning to generate extraction rules from examples. Although we also use a graphical interface to help the user identify the data to be extracted, we use in addition ontology in our extraction process. The advantage of this is an increased coherence in the organization of the extracted data. This is a fundamental difference compared to Ariadne's approach.

WMA uses a proper description language to represent and identify data inside HTML pages. XML documents are used as templates for identifying data to be extracted. The extracted data are then stored in a relational database. To be able to use this tool one has to learn WMA description language. Also, software agents are required to interact with databases to obtain data. In our approach, data are extracted directly from Web pages and can be queried directly.

Our extraction process is primarily aimed at web sites where information is regularly updated, but their organization remains the same (e.g. sites that provide exchange rates). It is this type of web site that we can expect precise recognition and extraction of information. The information provided by such websites is valuable for automatic agents. For example, one can imagine an agent that automatically converts one currency to another using the information we extract from a money exchange site.

2 Ontology as a model for Data Extraction

Web page information is not necessarily presented in the same way. Due to this fact, data extraction and exchange are not an easy task if different actors (producers or consumers of information) have not agreed on the semantics of data. By using a common model, we can provide a way to ensure a good interpretation and understanding of exchanged data. Ontology can be used as a common model for our purpose. Ontology is a way to decompose a world into objects, and a way to describe these objects. It is a partial description of the world, depending on the objectives of the designer and the requirements of the application or system. For each domain, there may be a number of ontologies [2]. The use of ontology differs from one application to the next, as do its design and its formalism of representation.

Our intended use of ontology is to describe a data model and to give semantics to data stored in web pages, rather than knowledge. Therefore, it is not necessary now to include inference and reasoning mechanisms to produce new knowledge. We will use a modeling of ontology close to object-oriented (OO) modeling. With the OO paradigm, we can express an ontology in an explicit way and generate software elements that are easily exploitable by other applications. We propose a design of ontology that uses a 3-level model: basic objects, model and meta-model. The meta-model layer has been introduced and experimented in [15] where the goal is to describe axioms of ontology. We use the same concept here, but in a more general framework. We use this approach to express specific design needs of the user, as in [15][5].

We consider four types of objects: entity, attribute, relation and constraint. These basic objects are used in the definition of the model. The meta-model will define inherent constraints of each object.

In figure 1, we show a simple example of an ontology model in finance. We consider financial domain as an example. Suppose that the entity STOCK is defined in the meta-model in relation *HAS-A* with entities STOCK_DESCRIPTION, DATE, etc.. The fact that STOCK is added in this model also imports all the definitions made in the meta model. This definition in the meta-model corresponds to inherent properties that we want to impose on the entity.

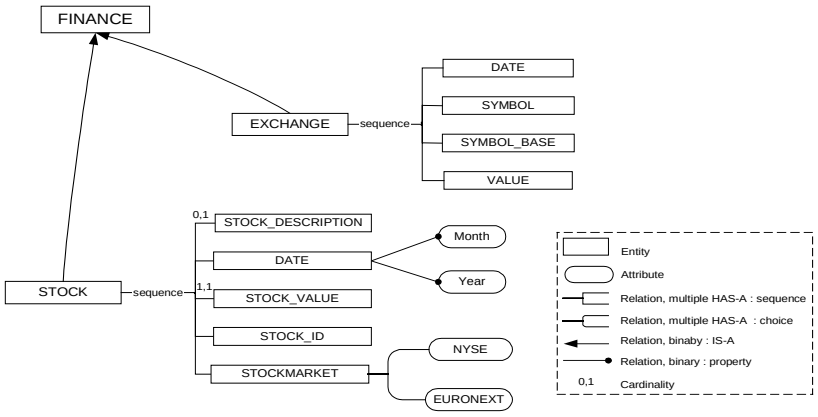

Figure 1. Example of a simplified model

SOX (Schema for Oriented-Object XML) [14] is used in our case for ontology definition and data modeling. It is developed by *Commerce One* in order to use XML in E-commerce [3]. The choice of SOX is motivated by the fact that one can use it to express the requirements that we defined previously. It is close to the OO paradigm and introduces concepts of OO-programming into XML documents.

3 Data Extraction Process

There are a few existing tools for data extraction, e.g. W4F (*World Wide Web Wrapper Factory*) [12] and JEDI (*Java Extraction and Dissemination of Information*) [7]. W4F is a development environment that allows users to construct a *Wrapper*, to compile it as a Java component and to include it in applications. A *Wrapper* is an interface to access the contents of web pages. It downloads documents from the Internet, corrects them and extracts data from them. Extracted data are connected with predefined variables [13].

JEDI is a set of tools for generating *Wrapper*s in order to extract data from textual sources. This may be applied to web pages as well as other textual files. The goal of a *Wrapper* is to indicate how to generate a representation in XML for the extracted data. Usually, a *Wrapper* is a text that contains a set of extraction transformation instructions, including rules and codes of control. A rule contains a syntactic constraint describing the data (character strings) to be extracted.

However, both W4F and JEDI are difficult to use. One has to learn the specific syntaxes of the extraction languages. In addition, it is also difficult to construct a tool to generate *Wrapper*s, due to the complexity of the language. For these reasons, we do not use this type of language. We designed a similar, but much simplified approach (in terms of use) that offers possibilities for future improvements.

Our hypothesis is that many websites do not often change their organization of information. New information is published with a rather steady structure. Then if we can recognize how information is organized, a precise data extraction can take place. Therefore, instead of relying on identification of boundaries of character strings within HTML documents, as is the case in TSIMMIS [6], we manually construct a description that shows how data can be found and extracted from a given Web site. The edition of the description is done manually, with the help of an assistance tool. Once the description is created, it can be integrated with autonomous agents to extract specific data.

The extraction process will actually contain two steps: we need to access the appropriate portion of data in the source document, and we also need to re-organize the extracted data into a useable form.

Documents in HTML do not allow for direct querying. In addition, there may be errors in HTML structures[1]. Therefore, we first convert the HTML document into XML by following their hierarchical structure [16]. Possible errors are corrected using HTML TIDY [11] and W4F [13] (correction and transformation module of W4F). Once a web page is transformed into XML, portions of data can be easily accessed using a DOM (Document Object Model) parser [4].

---

[1] The most frequent errors met in documents HTML are : missing of closing tags, incorrect fitment of tags and missed attributes

We can now start the extraction process from better structured XML documents. As we explained earlier, our general approach is based on a description of the domain in form of ontology. In fact, the extracted data will be linked to the corresponding elements of the ontology. Figure 2 shows the construction of the element <STOCK>. The latter is part of the element <FINANCE>. The figure also shows the data in the Web page that correspond to the elements of the ontology.

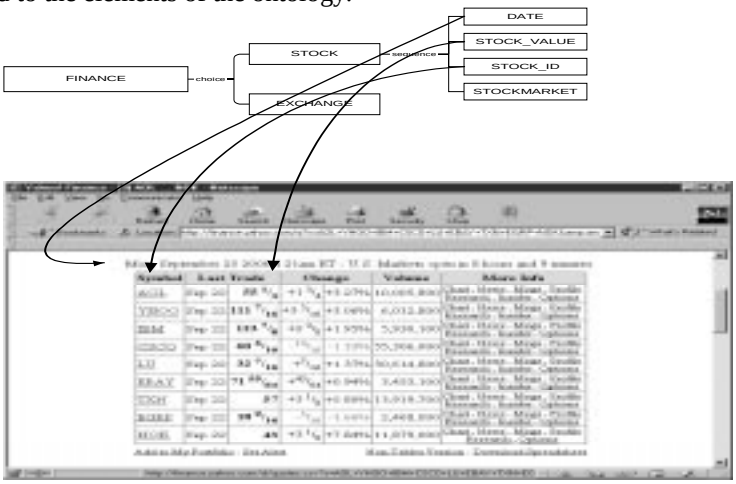

Figure 2. Mapping with the ontology

Once the correspondence between a raw data in XML and a node in the model is created, a query is associated to the node in order to extract the desired data.

4 The WeDaX Tool

The main steps of extraction are summarized in Figure 3. The lower part is that which can be integrated in a software agent. It is an automatic step since the agent uses only a description (generated before) to recover the data. Since the extracted data belongs to the same ontology, the results of the extraction of several sources have the same structure. A simplistic manner to combine them would be to put them in the same XML document.

We developed a specific tool called WeDaX (for Web Data extraction). The extraction follows several steps: download a web page and convert it into XML, construct a data model using the ontology, and map the XML document with the elements in the ontology. The HTML-XML conversion is performed using the tools described in section 2. This step can be done through our graphical interface. In the construction of the data model, the user chooses the elements of the ontology that reflect to his interests. This corresponds to determining "what to extract". The process of mapping and the underlying querying process answer the question "how to extract".

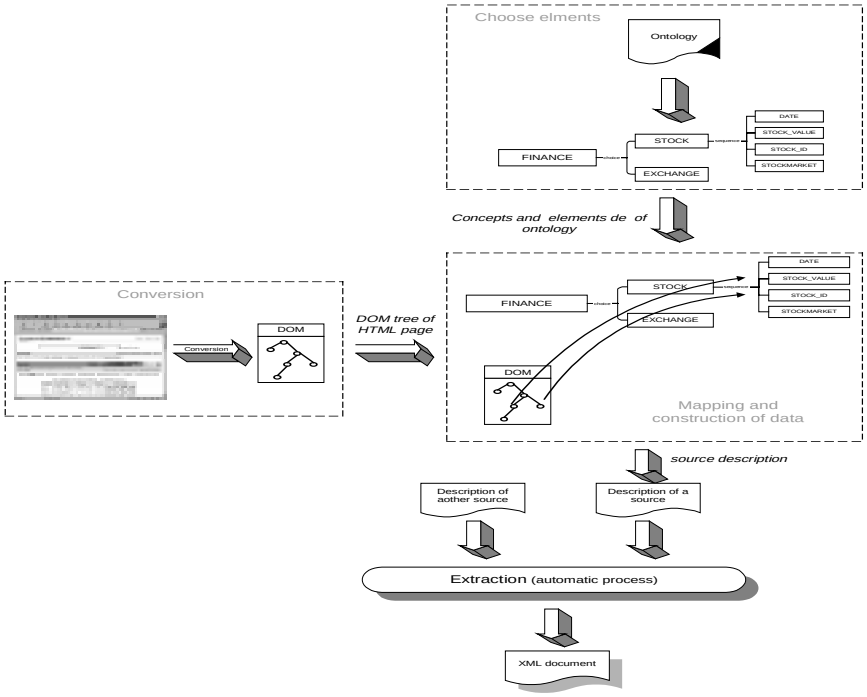

Figure 3 : Global view of the extraction

Figure 4. Screen snapshot - search and query DOM tree view of a web page

Figure 4 shows a screen snapshot while the user searches and queries the tree view (an XML DOM tree, panel 2) of a web page. Panel 1 displays the project structure with the associated sources and their descriptions. The user navigates the tree to locate data of interest or searches a specific word and obtains the node containing it. Data can also be

queried using an XQL query on the DOM tree and viewed in panel 3. Our tool helps the user (especially those who are not familiar with XQL language) to construct XQL queries. The user chooses some data on the page, the system generates the relevant queries and proposes a more general query to get data of the same kind (like all the prices) in the page. The user retrieves data for each element of the ontology. These data are combined together in a manner that conforms the structure given in the ontology.

The result is a description that contains all the processes in the form of a specification (description) for a given source. The specification contains, among other things, the names of entities/attributes, the queries, the transformations and the information about the structure of the final result (XML document). The advantage of making a specification is that it is reusable along time by an extraction program or an autonomous agent.

This tool has been completed and successfully integrated into a small agent-based application. The agent uses our data extraction tool to follow the course of stock indices and informs the user of the evolution of operations and trends. The integration of the tool with the agent is easy due to the standard model we created. This application shows the feasibility of our approach, and the utility of the extraction tool. We believe that this same approach also applies in larger applications.

5. Conclusions and discussions

In this paper, we dealt with the problem of data extraction from the web. In particular, our goal was to find a way to extract reliable data, and to convert them in a useable standard form. Data extraction consists of two steps: converting documents from HTML to XML and using a specific XQL query to extract appropriate portions of data from XML documents. The extraction process is defined by the user through a graphical tool. As the user has a tight control over the extraction process, the extracted data are of high quality and thus can be exploited by other programs or software agents. Integration of data from different sources is possible because they respect the same ontology.

The necessity of manual preparation may be a critical aspect. However, we believe that the current organization of the HTML documents do not allow for a completely automatic extraction for high-quality data. Some manual intervention is the price to pay for quality. Furthermore, the manual preparation is greatly facilitated by our assistance tool. The most critical point is that our approach assumes that a website has a fixed structure. If an information site is restructured, we have to construct a new extraction process. However, one has to notice that:

- A number of Web pages do provide dynamic data in a fixed page structure.
- Many Web pages are automatically generated from data stored in a database. The structure of this kind of data does not change frequently.

Indeed, during the several months of our project, the websites we used did not change their structure. This confirms the stability of structures of many information sites.

This study shows that it is possible to exploit and use automatically the data presented in some web pages. However, some manual preparation is necessary. We argue that this manual step is always necessary if one's intention is to extract reliable data to be exploited by other programs or software agents. Despite the manual preparation, we believe that this approach is appropriate for extracting data to be used by software agents.

Acknowledgement

This work has been supported by a scholarship of AUPELF to Hicham Snoussi, and a complementary scholarship of the NSERC.

References

[1] Atzeni, P., Mecca, G. and Merialdo, P., To Weave the Web - In Proceedings of the 23rd International Conference on Very Large Databases (VLDB'97), 1997

[2] Bezivin, J., Les nouvelles convergences : Objets, composants, modèles et ontologies, JICAA'97, Roscoff France, Mai 1997.

[3] Commerce One, http://www.commerceone.com/

[4] Document Object Model,http://www.w3.org/DOM/

[5] Gruber, T., Toward principles for the design of ontologies used for knowledge sharing, The International Workshop on Formal Ontology, March 1993.

[6] Hammer, J., Garcia-Molina, H., Cho, J., Aranha, R., and Crespo, A., Extracting Semistructured Information from the Web". In Proceedings of the Workshop on Management of Semistructured Data. Tucson, Arizona, May 1997.

[7] Huck, G., Fankhauser, P., Aberer, K. and Neuhold, E.J., JEDI: Extracting and Synthesizing Information from the Web, Conference on Cooperative Information Systems CoopIS'98, New York, August, 1998, IEEE Computer Society Press.

[8] Ishikawa, H., Kubota, K. and Kanemasa, Y., XQL: A Query Language for XML Data, Query Languages'98 (QL'98) workshop, Boston, Massachussets, December 1998.

[9] Knoblock, C. A., Minton, S., Ambite, J. L., Ashish, N., Modi, P. J., Muslea, I., Philpot, A. G. and Tejada, S., Modeling Web Sources for Information Integration. Proceedings of the National Conference on Artificial Intelligence, Madison, 1998.

[10] Ouahid, H and Karmouch, A., An XML-Based WEB Mining Agent, Proceeding of MATA'99, Ahmed KARMOUCH and Roger IMPEY eds., World Scientific, Ottawa, 1999.

[11] Raggett, D., HTML Tidy, http://www.w3.org/People/Raggett/tidy/

[12] Sahuguet, A. and Azavant, F., Building light-weight wrappers for legacy Web data-sources using W4F, International Conference on Very Large Databases (VLDB), Edinburgh - Scotland – UK, September 7 - 10 1999.

[13] Sahuguet, A. and Azavant, F., Looking at the Web through XML glasses, Conference on Cooperative Information Systems CoopIS'99, Edinburgh Scotland, September 2-4 1999.

[14] Schema for Oriented-Object XML, http://www.commerceone.com/xml/cbl/docs/

[15] Staab, S., Erdmann, M. and Maedche, A., An extensible approach for Modeling Ontologies in RDF(S), 12th International Workshop on Knowledge Engineering and Knowledge Management (EKAW'2000), Juan-les-Pins, French Riviera, October 2-6, 2000.

[16] W3C, http://www.w3.org

[17] W3C-Semantic Web, http://www.w3.org/2001/sw/