Automated Information Extraction from Web Sources: a Survey

Giacomo Fiumara

Dipartimento di Fisica, Università degli Studi di Messina,
Salita Sperone 31, I-98166 Messina, Italy
`giacomo.fiumara@unime.it`

Abstract. The Web contains an enormous quantity of information which is usually formatted for human users. This makes it difficult to extract relevant content from various sources. In the last few years some authors have addressed the problem to convert Web documents from unstructured or semi-structured format into structured and therefore machine-understandable format such as, for example, XML. In this paper we briefly survey some of the most promising and recently developed extraction tools.

1 Introduction

Although XML can be regarded as a *lingua franca* of the Web, nowadays almost all information available in Web sites is coded in form of HTML documents. This situation in unlikely to change in short or even medium term for at least two reasons: the simplicity and power of HTML authoring tools, together with a valuable inertia to change markup language. From the point of view of anyone interested in extracting information from Web sites, on the opposite, the difference between HTML and XML is evident. Although they are both derived from SGML, HTML was designed as a presentation-oriented language. On the contrary, XML has among its points of strength the separation between data and its human-oriented presentation, which allows data-centered applications to better handle large amounts of data. Another fundamental advantage of XML is the availability of powerful instruments for querying XML documents, namely XQuery/XPath[2], together with the increasing availability of native XML Databases [1], see for example eXist[3] and Monet[4]. Whereas [15] has surveyed the tools for information extraction in the Semantic Web, this survey would like to examine the state of the art of tools addressing the traditional Web. Even though the taxonomy proposed in [15] is largely adopted here, the emphasis is on what can be done in the context of existing, legacy Web sites. Community Web sites that have been serving their users for long time are a particular case in point. This brief survey will focus in Section 2 on the main questions regarding wrappers and their automatic generation and then give an overview of systems in Section 3. Related work will be presented in Section 4. Conclusions and future work will be presented in Section 5.

2 Wrapping a Web page

Information extraction from Web sites is often performed using wrappers. A wrapper is a procedure that is designed to access HTML documents and export the relevant text to a structured format, normally XML. Wrappers consist of a series of rules and some code to apply those rules and, generally speaking, are specific to a source. According to [6, 16] a classification of Web wrappers can be made on the base of the kind of HTML pages that each wrapper is able to deal with. Three different types of Web pages can be distinguished:

- *unstructured pages*: also called free-text documents, unstructured pages are written in natural language. No structure can be found, and only information extraction (IE) techniques can be applied with a certain degree of confidence.
- *structured pages*: are normally obtained from a structured data source, e.g. a database, and data are published together with information on structure. The extraction of information is accomplished using simple techniques based on syntactic matching.
- *semi-structured pages*: are in an intermediate position between unstructured and structured pages, in that they do not conform to a description for the types of data published therein. These documents possess anyway a kind of structure, and extraction techniques are often based on the presence of special patterns, as HTML tags. The information that may be extracted from these documents is rather limited.

Besides the HTML page structure, effective wrappers consider also the structure of hyperlink as it may reveal relevant information. Depending on the type of Web search engine the following kinds of results can be obtained:

- *one-level one-page* result: one page contains all the item descriptions;
- *one-level multi-pages*: a series of pages linked one to another, all containing the item description;
- *two-level pages*: a chain of pages, each containing a shortened description of items, each linking to a detailed page.

3 Information Extraction Tools

In this section a brief overview of some information extraction tools will be given. The idea is to illustrate the main features of tools belonging to the family of the so-called *HTML aware tools* (see [16, 15, 6] for the related taxonomy). Among the large number of information extraction tools we chose Lixto and Fetch as examples of powerful yet commercial semi-supervised wrapper generators, while RoadRunner is a prototype of fully automatic tools. Finally Dynamo will be described as an example of extraction tools which rely on the cooperation between the webmasters of the Web sites which publish information and the user willing to automate the extraction process.

3.1 LiXto

The *LiXto* project was started by Gottlob et al. at TUWIEN and is now developed and sold by the LiXto GMbh software house. *LiXto* [7, 8] is a method for visually extracting HTML/XML wrappers under the supervision of a human designer. *LiXto* allows a wrapper to interactively and visually define information extraction patterns on the base of visualized sample Web pages. These extraction patterns are collected into a hierarchical knowledge base that constitutes a declarative wrapper program. The extraction knowledge is internally represented in a Datalog-like programming language called *Elog* [9]. The typical user is not concerned with *Elog* as wrappers are build using visual and interactive primitives. Wrapper programs can be run over input Web documents by a module in charge of extraction which then translates the output in XML. The latter is done thanks to a *XML translation scheme* with the possibility to construct a Document Type Definition (DTD) which describes the characteristics of the output XML documents. Among the most interesting features is the ability to access Web data even if protected by means of a username/password authentication mechanism, if the user provides them. LiXto has also the possibility to follow links thus collecting information even if spread across several Web pages, the flexibility to output extracted structured information into several formats, namely XML, SQL records and XHTML newly produced Web pages. Finally, the extraction process can be scheduled in order to be repeated at fixed times.

3.2 Fetch Agent Platform

Fetch Agent Platform [21] is another example of commercial information extraction tool.It is based on two major components, the AgentBuilder which provides a visual environment that allows a user to construct web agents, and the AgentRunner which automatically performs the tasks specified by the agent, and produces structured data. The framework also provides a tool able to monitor Web target pages, specifying which data fields are to be checked. The extraction rules are based on landmarks (groups of consecutive tokens) that enable a software agent to locate the start and end of fields within a page. The extraction algorithm that learns these landmarks based on examples labeled by the user and uses the hierarchical structure of the page to constrain the learning problem.

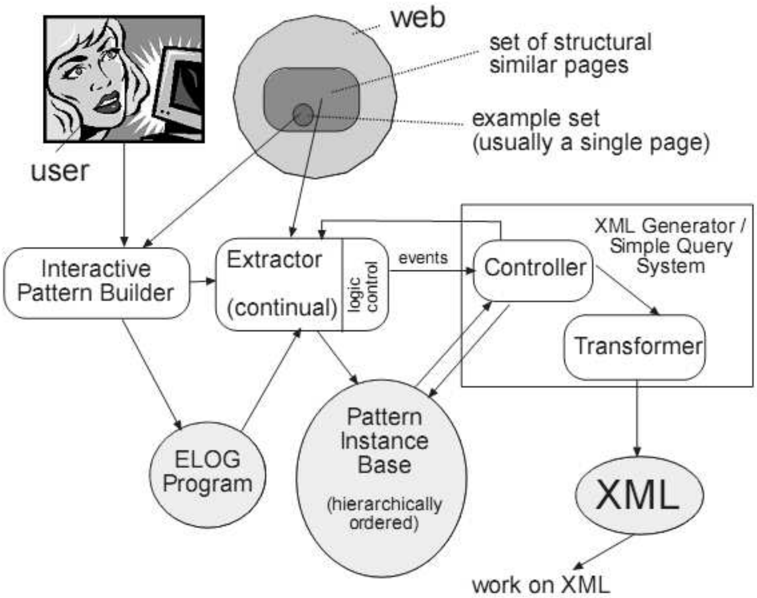
web
set of structural similar pages
example set (usually a single page)
user
Interactive Pattern Builder
Extractor (continual)
logic control
events
XML Generator / Simple Query System
Controller
Transformer
ELOG Program
Pattern Instance Base (hierarchically ordered)
XML
work on XML

Fig. 1. The architecture of *LiXto*, from [7]

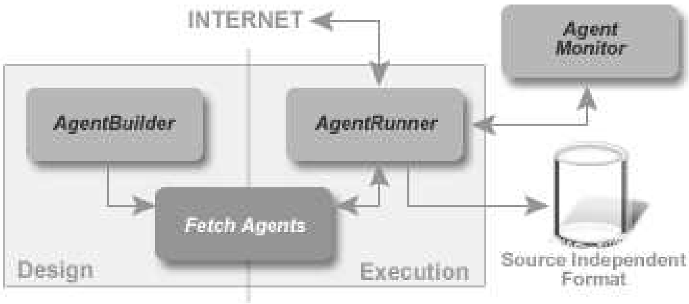

Fig. 2. The architecture of *Fetch Agent Platform*, from [21]

3.3 RoadRunner

RoadRunner [10, 11, 12, 13, 14] was developed at the University of Roma 3 and applies to intensive Web sites, i.e. those sites with large amounts of data and a rather regular structure. RoadRunner works by comparing the HTML structure of a set of sample pages of the same type, and generates a schema for the data contained in the pages. This schema is used as a starting point for the inference of a grammar which is capable to recognize the instances of attributes identified for this schema in the set of sample pages. The extraction procedure is based on an algorithm that compares the tag structure of the set of sample pages and produces regular expressions able to handle structural differences found in the set of sample pages. A peculiar feature of RoadRunner is that this procedure is completely automatic and no user intervention is required.

3.4 Dynamo

The Dynamo Project [18, 19] addresses data extraction and channeling over legacy Web sites in plain HTML. Dynamo is intended to benefit two types of users. First, webmasters may employ it to manage the creation of RSS feeds, thus avoiding to do it by hand or by means of proprietary software. Second, users, i.e., consumers of feeds, may use it to overcome limitations such as i) old feeds may not be consulted and usually are deleted from servers and ii) traditional HTML servers cannot execute advanced queries directly. On the contrary, with Dynamo it becomes possible to:

- automatically and dynamically generate RSS feeds starting from HTML Web pages;
- store feeds in chronological order;
- query and aggregate them thanks to Web Services (WS) acting as agents.

It is important to stress that these results were obtained with a lightweight pull algorithm for retrieving HTML documents by Web servers, thus minimizing the required Web traffic for the updates of news sources [19].

HTML documents contain a mixture of information to be published, i.e., meaningful to humans, and of directives, in the form of tags, that are meaningful to the browsers and determine the appearance on the screen. Moreover, since the HTML format is designed for visualization purposes only, its tags do not allow sophisticated machine processing of the information contained therein.

Among other things, one factor that may prevent the spread of the Semantic Web is the complexity of extracting, from existing, heterogeneous HTML documents machine-readable information. Although the Dynamo project addresses only a fraction of the Semantic Web vision, management of HTML documents needs some technique to locate and extract some valuable and meaningful content. Therefore, a set of annotations, in form of meta-tags, were defined; they are inserted inside HTML in order to highlight informational content that is essential for the creation of a RSS feed. In this application, meta-tags are used as annotations, to describe and mark all interesting information, in order to help in the extraction and so-called XML-ization phases. Notice that with pages that are dynamically generated out of some template (which is the case with practically all on-line fora) Dynamo annotation is done, manually but only once and for all, over the page template.

Once HTML documents are processed by Dynamo, annotated semantic structures are extracted and organized into a simple XML format to be stored and used as a starting point for document querying and transformation. The structure of the XML output resembles the structure of meta-tags previously defined and the RSS XML structure, in order to facilitate transformations from the former to the latter. At the moment, a version of Dynamo is undergoing a phase of testing in several forum of the Milan Community Network (Rete Civica Milanese).

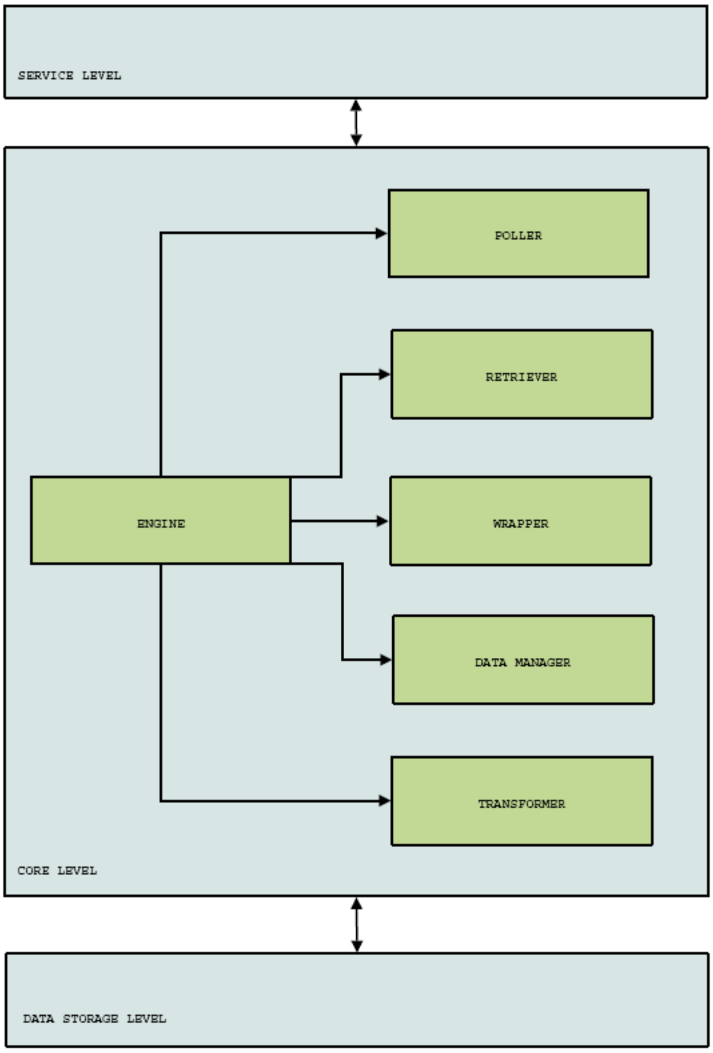

Fig. 3. The architecture of *Dynamo*

4 Related Work

In the past few years, many approaches to the problem of Information Extraction (IE) by means of Wrapper Induction (WI) systems have been tackled. Previously proposed taxonomies will be briefly examined in this section. Hsu and Dung [22] classified wrappers into 4 categories:

- hand-made wrappers using general-purpose programming languages;
- designed programming languages;
- heuristic-based wrappers;
- WI approaches.

A complete categorization was made by Laender et al. [15]. They proposed the following taxonomy:

- languages for wrapper development;
- HTML-aware tools;
- NLP-based tools;
- wrapper induction tools;
- modeling-based tools, and
- ontology-based tools.

They also compared among the tools using these features: degree of automation, support for complex objects, page contents, availability of a GUI, XML output, support for non-HTML sources, resilience and adaptiveness.

Sarawagi [24] classified Web sites wrappers according to the amplitude of the tasks they are able to face. So he distinguishes *record-level wrappers*, capable to extract elements of a single list from a Web page, *page-level wrappers* which extract elements of multiple records and, finally, *site-level wrappers* which can extract and convert into structured format an entire Web site.

More recently, Chang et al. [17] proposed a three-dimensional representation of IE features: the first dimension evaluates the difficulty of an IE task, the second compares the various techniques and the third dimension compares both the training effort of a user and the necessity to port an IE system across different domains.

5 Conclusions and future work

In this paper we presented a short survey of most recent tools for the extraction of information from Web sites. All the tools presented here automatically generate wrappers in order to accomplish their task and all of them provide output data in XML format, thus focusing on the meaning of data rather than on their graphical representation.

There are a series of current and future applications where information extraction tools can fully exploit their power. One of the most promising seem to be the comparison of items, for example in commercial aggregators.

The possibility for a user to compare different offerings of the same object is a feature currently not supported by online auction sites.

Even in the area of communication, the possibility of aggregating and querying information automatically extracted from different Web news sites seems really promising, specially in conjunction with the features offered by XML-based query engines. This, together with more flexible and powerful extraction tools will certainly help paving the road to the semantic web.

References

1. Bourret RP (2005) XML and Databases. http://rpbourret.com
2. W3C (2005) XQuery 1.0. http://w3c.org/TR/xquery
3. eXist (2007) Open Source XML Native Database. http://exist-db.org
4. MonetDB (2007) http://monetdb.cwi.nl
5. Muslea I. (1999) Extraction Patterns for Information Extraction Tasks: A Survey. American Association for Artificial Intelligence
6. Eikvil L. (1999) Information Extraction from World Wide Web - A Survey -. Technical Report 945, Norvegian Computing Center
7. Baumgartner R., Flesca S., Gottlob G. (2001) Visual Web Information Extraction with Lixto. In Proc. of VLDB, 2001
8. Baumgartner R., Flesca S., Gottlob G. (2002) Declarative Information Extraction, Web Crawling and Recursive Wrapping with Lixto. In Proc. of LPNMR, 2002
9. Baumgartner R., Flesca S., Gottlob G. (2002) The Elog Web Extraction Language.
10. Mecca G., Grumbach S. (1999) In search of the lost schema. ICDT(1999)
11. Crescenzi V., Mecca G., Merialdo P. (2001) RoadRunner: Towards Automatic Data Extraction from Large Web Sites. VLDB(2001)
12. Crescenzi V., Mecca G., Merialdo P. (2001) The RoadRunner Project: Towards Automatic Extraction of Web Data. ATEM (2001)
13. Crescenzi V., Mecca G., Merialdo P. (2001) Automatic Web Information Extraction in the RoadRunner System. DASWIS (2001)
14. Crescenzi V., Mecca G., Merialdo P. (2002) Wrapper Oriented Classification of Web Pages. ACM SAC (2002)
15. Laender A.H.F. , Ribeiro-Neto B.A., da Silva A.S., Teixeira J.S. (2002) A Brief Survey of Web Data Extraction Tools. SIGMOD Records 31(2) 2002
16. Flesca S., Manco G., Masciari E., Rende E. and Tagarelli A. (2004) Web wrapper induction: a brief survey. AI Communications 17 (2004) 57 - 61
17. Chia-Hui Chang, Kayed M., Girgis M.R., Shaalan K. (2006) A Survey of Web Information Extraction Systems. IEEE Transactions on Knowledge and Data Engineering, TKDE-0475-1104.R3
18. Bossa S. (2005) Gradation Project in Informatics. University of Messina (in Italian)
19. Bossa S., Fiumara G., Provetti A. (2006) A Lightweight Architecture for RSS Polling of Arbitrary Web sources. Proc. of WOA conference. Available from http://mag.dsi.unimi.it/

20. De Cindio F., Fiumara G., Marchi M., Provetti A., Ripamonti L.A. and Sonnante L. (2006) Aggregating information and enforcing awareness across communities with the Dynamo RSS feeds creation engine: preliminary report. OTM Workshops (1) 2006: 227-236
21. Fetch Technologies, available from http://www.fetch.com
22. Hsu C.-N. and Dung M. (1998) Generating finite-state transducers for semi-structured data extraction from the web. Journal of Information Systems 23(8): 521-538 (1998)
23. Chang C-H., Hsu C.-N. and Lui, S.-C. (2003) Automatic information extraction from semi-structured web pages by pattern discovery. Decision Support Systems Journal, 35(1): 129-147 (2003)
24. Sarawagi S. (2002) Automation in information extraction and integration, Tutorial of VLDB (2002)