International Journal of Science and Engineering Vol.4 No.4(2014):5-9

Roll Your Own Web Database: An Innovative Approach for Providing Searchable Web Content

Chun-Hsiung Tseng
Department of Information Management, Nanhua University, Chiayi County, Taiwan
lendle_tseng@seed.net.tw

Yung-Hui Chen
Department of Computer Information and Network Engineering, Lunghwa University of Science and Technology, Taoyuan County, Taiwan
cyh@mail.lhu.edu.tw

Han-Ci Syu, Chu-Chun Chuang, Jia-Hua Wu, Yan-Ru Jiang
Department of Information Management, Nanhua University, Chiayi Country, Taiwan

摘要

網際網路是一個巨大的資料庫，與常見的關聯式資料庫類比，要從今天的網際網路取得資訊，現有的搜尋方式並不充分。網頁資料相當於原始資料表，但關聯式資料庫具有View、SQL 語法等等工具，網際網路幾乎只有關鍵字搜尋一項。HTML 最初的設計理念，是針對資料的排版呈現，提供的是呈現層，而非資料層。本研究提出了 Object-Oriented Schema Model (OOSM)，這是一個文法模型。OOSM 可由領域專家或一般的使用者來制定。使用者利用 OOSM 來標記網頁內容，OOSM 核心可以將一般呈現用的網頁轉譯成可供資料處理的內容。本研究除了文法模型，亦實作了一個供使用者建立對應的工具，以便於使用。

關鍵詞：資訊擷取、本體論、標籤

Abstract

The paper is aimed at addressing two issues: first, despite of the importance of semantic information in HTML pages, it is often ignored by search engines due to various technology difficulties; second, the ambiguity problem sometimes makes results returned by search engines much less useful. OOSM, a schema model as well as a set of information processing tools, is proposed in the paper. OOSM develops the concept of coarse mapping, that is, users are allowed (but not restricted) to associate a grammar node to a sub section instead of a single node on a HTML page. AS a result, minor modifications of the annotated HTML page can be tolerated. We believe that OOSM is a right solution for the issues presented in this paper.

Keywords: *information extraction, ontology, labeling*

Introduction

The Web is a huge database, however, compared with relational databases, the methods for searching information on the Web is not sufficient. A HTML page is just like a raw database table. For relational databases, there are an enormous amount of tools such as view and SQL to be utilized. But, for the Web, what we have is keyword search only. A problem of HTML is it is originally designed to support the presentation layer, not the data layer. This makes searching difficult. Modern search engines are capable of extracting text contents from various information sources. Neverthless, from a semi-structured document, such as a HTML page, existing keyword-based approaches are good at processing just syntactic information, not sematic information. For simple search cases such as looking for documents containing someone's name, technologies we possessed today appears sufficient and adequate. The problem is, what if we want to achieve more complex tasks? For example, consumers can benefit from a price list of smartphones if she/he is planning to buy a new one. Search engines are capable of returning lists of HTML pages containing the word "smartphone", however, they have no idea about the real semantics of these pages and thus it is not possible for search engines to extract further metadata such as price from these HTML pages. Furthermore, it is not impossible that some of the returned HTML pages contain the keyword "smartphone" but have no information of specific smartphones!

The scenarios presented above illustrate two issues: first, despite of the importance of semantic information in HTML pages, it is often

*Corresponding author: lendle_tseng@seed.net.tw

DOI：10.6159/IJSE.2014.(4-4).02

ignored by search engines due to various technology difficulties; second, the ambiguity problem sometimes makes results returned by search engines much less useful. To overcome these difficulties, making semantic information accessible for search engines is required. Trying to achieve this without causing severe impact to the Web infrastructure today is very challenging. In this research, we propose OOSM, which is in fact a grammar model. With OOSM, domain experts and ordinary users can define grammars and then end users use these grammars to annotate HTML pages. Evaluating annotations, domain objects conform to grammars will be obtained. Semantic information carried by domain objects can then be utilized for acquiring better search results. To reduce possible impacts, the proposed mechanism adopts external annotations. That is, existing HTML pages are left as-is, and an additional external mapping file is used to maintain the annotation information of an HTML page. Definitely, the design can raise additional issues such as the maintainability of connections between annotations and HTML pages after the original annotated page is modified. To alleviate the problem, OOSM develops the concept of coarse mapping, that is, users are allowed (but not restricted) to associate a grammar node to a sub section instead of a single node on a HTML page. AS a result, minor modifications of the annotated HTML page can be tolerated. Of course, with coarse mappings, post-processing will be needed. In the proposed system, JSON-encoded domain objects are then imported into NoSQL-style databases for further post-processing.

To sum up, the OOSM system contains the following components:

1. a grammar model
2. an annotation processing component
3. an annotating tool, the OOSM mapper
4. a database layer that is based on NoSQL-style databases and is capable of processing JSON-encoded objects

Related Works

Crowd search is highly related with social networking [1]. The opinions collected within friends and expert/local communities can be ultimately helpful for the search task. For example, the question "find all images that satisfy a given set of properties" can be difficult for machines to proceed, but with the help of human intelligence, answering the question becomes simpler [5]. A special query interface that let users pose questions and explore results spanning over multiple sources was proposed in [2]. Another type of crowd search and crowd sourcing is social bookmarking. As shown in Heymann's research work [3], social bookmarking is a recent phenomenon which has the potential to give us a great deal of data about pages on the web.

Considering the Web as a huge database with plenty of information, existing query techniques are far from perfect. Today, the most widely adopted query technique is the keyword-based search. The research of Konopnicki and Shmueli [4] examined some trends in the domain of search, namely the emergence of system-level search services and of the semantic web. In the research, a SQL-like solution was surveyed. In [2], a YQL (Yahoo! Query Language [6]) is implemented. The framework consists of some interaction primitives and is aimed at supporting users in finding responses to multi-domain queries.

The Schema Model

Extracting information from relational databases is usually much easier than extracting information from Web pages since the formers are structured information sources while the latters are not. There are various information extraction technologies that are designed for the Web. However, most of them suffered from unstability and limited-adoption issues. To ease the data processing task, the proposed platform provides schema sub components that are used for designing and maintaining schemas. Here, the researcher would like to avoid complex schema structures such as XML schema and DTD files. Instead, the researcher proposes an Object-Oriented Schema Model (OOSM). An OOSM is defined as the following:

(ROOT_ELEMENT, (RULE)*)

That is, an OOSM consists of a root element and a list of rules. A ROOT_ELEMENT is represented as a namespace URI and a local element name pair and is used to group OOSM instances into categories. Furthermore, a RULE is represented as the following:

(ELEMENT, {CONSTRUCT1, CONSTRUCT 2, ……, CONSTRUCT n})

That is, a RULE consists of a root ELEMENT and a set of unordered schema CONSTRUCTs. Again, an ELEMENT is represented by a namespace URI and a local element name.

A schema CONSTRUCT is further defined as the following:

CONSTRUCT ::= ARRAY(ELEMENT) | ELEMENT

That is, a schema CONSTRUCT is either an array of instances of an ELEMENT or an ELEMENT alone.

Keeping simple is essential for OOSM to work. Note that the order of child COSTRUCTs of an ELEMENT is not strictly enforced. They are just regarded to as properties of an object according to the convention of the object-oriented paradigm. Furthermore, there is no complex structure defined in OOSM. Different from complex standard such as XML schema, which defines quite a few structures: sequence, choice, and all, etc., there are only two types of CONSTRUCTs defined in OOSM: array of ELEMENTs and a single ELEMENT. The reason of sticking to simplicity is that most Web users are not domain experts of the grammar-definition field, thus complex grammar model may become an obstacle of wide adoption.

Additionally, in OOSM, the content of an ELEMENT is not strictly typed. An ELEMENT simply attaches some form of meaning to the content surrounded by it. Because of the diversity of Web contents, the researcher would like to claim that a strictly typed grammar model can be difficult to adopt. On the other hand, an ELEMENT itself, that is, the namespace URI and local name pair, already provides a good amount of information for the content associated with it.

The Mapper Tool

OOSM Mapper is used for annotating a HTML source against a selected schema. The figure below shows the user interface of OOSM Mapper:

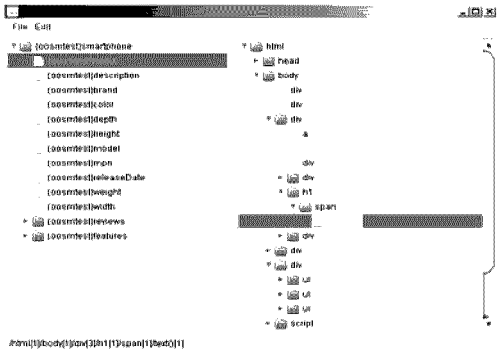

Figure1. OOSM Mapper Application.

OOSM Mapper is designed around the project-centric concept. That is, the adopted schema, the target HTML document, and the created annotations are stored in a project. The design makes annotations more manageable. To create a project, at first, one should open the create project dialog:

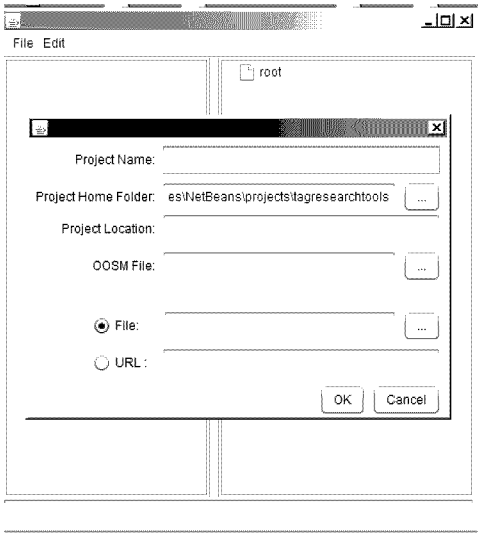

Figure 2. Project Creation.

A project contains an OOSM file and a HTML document. Users either specify the file path or the URL of the HTML documents to be annotated. If a URL is specified, OOSM Mapper will automatically fetch the document from the specified URL. Note that, a normalization process will be performed against the retrieved document to avoid common HTML errors.

To annotate the target HTML document, one simply selects a schema node from the left panel and a HTML node from the right panel, and then triggers the binding dialog shown in figure 3. In the binding dialog, there are three input fields: ID, Definition, and Target. The ID field shows the id of the current binding, which is managed by OOSM Mapper automatically. The Definition field shows the schema node used for annotating the selected HTML node. The Target field shows the specification of the annotated target node. By default, the Target field shows the xpath of the selected HTML node. However, users can enter any valid xpath expression here, including xpath functions. The design will be useful when it is inadequate to annotate a whole node. For example, users may want to annotate a schema node to only a sub string of the selected HTML node.

Fig 3. The Binding Dialog.

Finally, to view the extracted results, one simply open the show result dialog shown below:

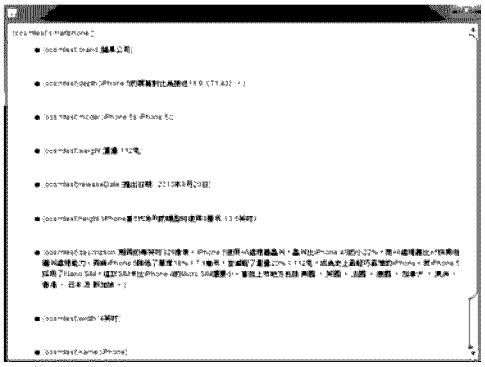

Figure 4. Show Result Dialog.

The Database

What is a Web database? Throughout this paper, it is referred to as the combination of the followings:

1. a virtual or physical medium that is capable of storing contents from the Web
2. a system for managing the above contents; the system should support the following operations:
   i. create/update/delete data sets
   ii. create/modify the schema of the data sets
   iii. define data to be included in the data sets
   iv. aggregate or transform existing data

   note that the operations supported should not include create/update/delete of the data itself since the data is collected from the Web and should be read-only; otherwise, the system becomes a possible source of fraud
3. a search tool; in this research, the MongoDB database is chosen as the underlying database. MongoDB accepts JSON-style documents and provides a rich set of query functionalities that can fit our requirements.

A lean approach for defining databases is required. The concept of this research is to propose the idea of domain-specific databases. Such databases have the advantage of accuracy and efficiency. By limiting the domain of a specific Web database, extracting semantic information from the database becomes much easier. The code snippets below demonstrate how to define a Web database:

```
{
    "database-name":{
        "namespace-uri": "sample",
        "local-name": "3c-accessories"
    },
    "datasets":[
        {
        "oosm-schema":{
            "namespace-uri": "sample",
            "local-name":
"smartphone"
},
"urls":[
    "http://en.wikipedia.org/wiki/Smartphone",
    "http://cellphones.about.com/ "
],
“bindings”:[
    “http://sample1.ilab.twgogo/binding1”,
    “http://sample1.ilab.twgogo/binding2”
```

```
]

}]}
```

As shown above, a database definition contains two major parts: database-name and datasets. To preserve uniqueness, a fully-qualified name with a namespace uri is required. The datasets part represents a list of dataset sections. Each dataset section contains a schema reference, a list of urls, and a list of corresponding binding definitions. According to the database definition, data stored in these urls is then extracted to json-format strings and then stored into the underlying MongoDB.

Conclusions & Future Work

In this paper, an innovative approach for providing searchable Web content is proposed. Our approach provides a way for users to define advanced information sources, which are just like views of relational databases. Based on advanced information sources, some more feature-rich search operations are also provided. The researcher claims that the proposed mechanism is more efficient and powerful than ordinary keyword-based search. The current implementations are mostly desktop-based GUI applications. In the future, it is planned to port the current implementations to be Web-based for easier integration and adoption.

ACKNOWLEDGEMENT

First, I have to acknowledge NSC's support for the completion of this research. Furthermore, the paper was prepared in collaboration with my lab members in Department of Information Management, Nanhua University, Taiwan. I would like to acknowledge the following persons who have made the completion of this research possible: Chu-Chun Chuang, Jia-Hua Wu, Han-Ci Syu, and Yan-Ru Jiang.

參考文獻

[1] Bozzon, A., Brambilla, M., and Ceri, S. Answering search queries with CrowdSearcher. In Proceedings of the 21st international conference on World Wide Web, 2012, 1009-1018.

[2] Bozzon, A., Brambilla, M., Ceri, S, and Fraternali, P. Liquid query: multi-domain exploratory search on the web. Proceedings of the 19th international conference on World wide web, 2010, 161-170.

[3] Heymann, P., Koutrika, G., and Garcia, H. Can social bookmarking improve web search? In Proceedings of the 2008 International Conference on Web Search and Data Mining, 2008, 195-206.

[4] Konopnicki, D. and Shmueli, O. Database-inspired search. In Proceedings of the 31st international conference on Very large data bases, 2005, 2-12.

[5] Parameswaran, A., Garcia-Molina, H., Park, H., Polyzotis, N., Ramesh, A. and Widom, J. CrowdScreen: algorithms for filtering data with humans. In Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, 2012, 361-372.

[6] Yahoo, Yahoo! Query Language, http://developer.yahoo.com/yql/