A Framework for Extracting, Classifying, Analyzing, and Presenting Information from Semi-Structured Web Data Sources.

Article *in* Journal of Next Generation Information Technology · November 2010

DOI: 10.4156/jnit.vol1.issue3.12 · Source: DBLP

CITATIONS
2

READS
639

4 authors:

Mahmood Shakir Hammoodi
University of Babylon
15 PUBLICATIONS **51** CITATIONS
SEE PROFILE

Hamidah Ibrahim
Universiti Putra Malaysia
230 PUBLICATIONS **1,380** CITATIONS
SEE PROFILE

Aida Mustapha
Universiti Tun Hussein Onn Malaysia
383 PUBLICATIONS **3,677** CITATIONS
SEE PROFILE

Lili N. Abdullah
Universiti Putra Malaysia
61 PUBLICATIONS **354** CITATIONS
SEE PROFILE

A Framework for Extracting, Classifying, Analyzing, and Presenting Information from Semi-Structured Web Data Sources

Mahmoud Shaker[1], Hamidah Ibrahim[2], Aida Mustapha[3], Lili Nurliyana Abdullah[4]
Department of Computer Science
Faculty of Computer Science and Information Technology
Universiti Putra Malaysia, 43400 Serdang, Selangor D. E., Malaysia
[1]mah222254@yahoo.com, [2]hamidah@fsktm.upm.edu.my, [3]aida@fsktm.upm.edu.my,
[4]liyana@fsktm.upm.edu.my
doi:10.4156/jnit.vol1. issue3.12

Abstract

Extracting information from the web data sources becomes very important because the massive and increasing amount of diverse semi-structured information sources in the Internet that are available to users, and the variety of web pages making the process of information extraction from web a challenging problem. This paper proposes a framework for extracting, classifying, analyzing, and presenting semi-structured web data sources. The framework is able to extract relevant information from different web data sources, and classify the extracted information based on the standard classification scheme of Nokia products, which has been chosen as the case study.

Keywords: *Information Extraction, Semi-Structured, Web Data Sources*

1. Introduction

Nowadays, many users use web search engines to find and gather information. User faces an increasing amount of various semi-structured information sources. The issue of correlating, integrating and presenting related information to users becomes important. When a user uses a search engine such as Yahoo and Google to seek specific information, the results are not only information about the availability of the desired information, but also information about other pages on which the desired information is mentioned. The number of selected pages is enormous. Therefore, the performance capabilities, the overlap among results for the same queries and limitations of web search engines are an important and large area of research.

At the present time, the Internet is general and many people use the Internet to find information. A variety of web pages and the frequently changing of information in web data sources make searching and extracting information very difficult. When Internet users want to get information about Nokia products for example, they first visit search engines such as Yahoo and Google, and then visit all web sites suggested by the search engine.

Many researchers [1], [2], [4], [5] work on extraction of information from web data sources in different domain (traveling, products, business intelligence) but these researches deal with limited web data sources and the user still need to use the search engines such as Yahoo and Google to gather more information.

In this paper we present our proposed framework for extracting, classifying, analyzing, and presenting information from different web data sources. A brief description of the framework can be found in [15]. The components of the proposed framework include *query interface* (QI) which is used to accept user's queries and search web pages through search engine based on the user's queries; *information extraction* (IE) which is used to extract and classify the web pages obtained from QI and convert the extracted and classified web pages into text form; and *relevant information analyzer* (RIA) which is used to determine the relevant information extracted from IE.

The rest of the paper is organized as follows. In section 2, the previous works related to this research are reported. In section 3, we present the proposed framework. Conclusion is presented in the final section, 4.

2. Related works

Many researchers have proposed approaches for searching information as discussed below.

Bernard, Danielle, and Amanda (2007) [7] proposed a model to improve Web search engines by classifying user searcher based on intent in terms of the type of content specified and operationalize these classifications with defining characteristics. The limitation of this study is that they assigned each query to one and only one category.

Shady, Fakhri, and Mohamed (2007) [14] proposed a novel sentence-based representation, called Conceptual Ontological Graph (COG) for enhancing search engine quality. It captures the semantic structure of each term within a sentence and a document, rather than the term frequency within a document. This research needs to extend, to link the presented work to web document retrieval and apply the same method to text classification.

Hongkun, Weiyi, and Clement (2007) [12] proposed a new technique to automatically extract the precise search result records template for any search engine. There is one limitation in this research. The statistics-based solution does have an inherent weakness in dealing with attributes that have the same or nearly the same values (data units) in all search result records. These data units will be mistakenly recognized as template texts.

Another stream of researchers proposed tools for extracting information from semi-structured information web data sources as discussed below.

The Information Systems Universal Data Browser (IS UDB) (2006) [1] which has been proposed by Guntis and Girts is used for searching, extracting, analyzing, classifying, translating, storing, integrating, and browsing semi-structured data. The IS UDB deals with limited semi-structured data sources (web pages), thus user needs to use search engines such as Yahoo and Google to get the required information.

Srinivas, Fatih, and Hasan (2007) [4] work on information extraction from web pages using presentation regularities and domain knowledge. They argued that there is a need to divide a web page into information blocks, or several segments before organizing the content into hierarchical groups and during this process (partition a web page) some of the attribute labels of values may be missing.

Paul and Mukund (2005) [8] present a classification algorithm based on discriminatively trained context free grammar (CFG) to extract information from semi-structured text, and propose that the effective search and reuse of extracted information requires field extraction. The challenge is in converting the semi-structured information of customer (which is already available in an unstructured form on web sites and in email) into the regularized or schematized form required by a database system.

There are many researches focusing on information extraction for business intelligence to collect available information for companies and easement the efforts concerned in gathering, merging, and analyzing information. Horacio, Adam, Diana, and Kalina (2007) [5] proposed the MUSING project (Multi-industry, Semantic-based next generation business intelligence), that needs to cover many semantic categories including locations, organizations and specific business events to help companies that want to take their business overseas and concerned in knowing the best place to exploit.

Fei, Hong, Zhuang, and Zhao (2005) [13], proposed an information extraction system that aims to automate the tedious process of extracting large collections of facts from large-scale, domain-independent, and scalable manner. The biggest of these challenges stems from the fact that search engines only make a small fraction of their results accessible to users.

Another stream of researchers work on extraction of information with agent. Sung, Kyung, Tae, and Hyuk proposed an Intelligent Traveler Support System (ITSS) (2001) [2] for helping traveler to find important information about traveling more easily and effectively. There are many limitations in the traveler supporting system. One of this limitations, the system deals with limited web pages which are related to destinations and weather. Thus, travelers need to search through the numerous web pages to gather all the necessary information using search engines such as Yahoo and Google.

Hongkun, Weiyi, and Clement (2006) [11] proposed a method to automatically generate wrappers for extracting search result records from all dynamic sections on result pages returned by search engines. There are many limitations in this research. The lack of a general pattern for the sections on a result page makes it more difficult to extract the sections. Some consecutive sections with the same format may be mistakenly extracted as records while some large records may be incorrectly extracted as sections.

Utku and Torsten (2006) [10] proposed a complete system for semi-automatic wrapper generation that can be trained on different data sources in a simple interactive manner. This method typically requires the labeling of a single tuple, followed by a selection of a tuple set from a ranked list where the desired set is usually among the first few, plus the labeling of another tuple in the rare case when the desired set is not found in the list.

Gilles (2006) [9] proposed a new method for extracting information from the web using wrappers and this method (wrapper construction) is based on different techniques: labeled page based induction, relation extraction, structure discovery and most recently a new approach based on the generalization of the contexts of a small set. The limitation is to build a wrapper for a data source which can extract a relation it contains. The description of the relation to extract is given in the form of a set of example instances.

3. The proposed framework

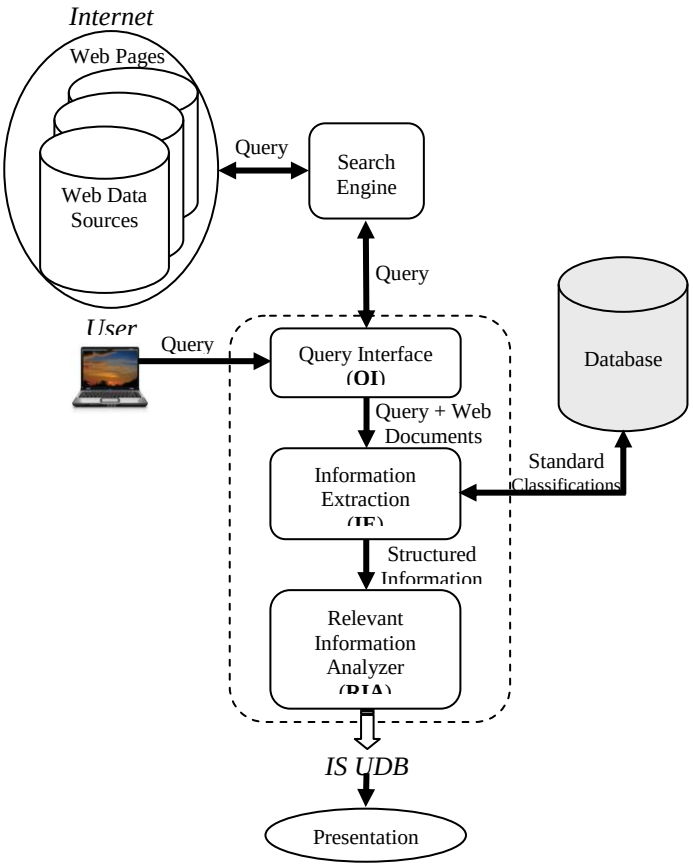


Figure 1. The proposed framework

The major tasks of the proposed framework are to extract relevant information from various web pages (WPs), classify the extracted information, analyze the extracted and classified information, and simplify the analyzed information. Figure 1 illustrates the proposed framework. The framework consists of the following main components: Query Interface (QI), Information Extraction (IE), and Relevant Information Analyzer (RIA). In the following we discuss each of these components in details and we use the notations Attr (WP), Sub_Attr (WP), and G_Sub (WP) to denote the attributes of WP, sub-attributes of WP, and group of sub-attributes, respectively. We use Attr (DB), Sub_Attr (DB), and G_Sub (DB) to denote the attributes, sub-attributes, and group of sub-attributes, respectively that are saved in a database which represent the attributes, sub-attributes, and group of sub-attributes,

respectively of a standard classification scheme (SCS). For the chosen case study, we use the SCS proposed by Guntis and Girts (2006) [1].

3.1. The Query Interface (QI)

The Query Interface (QI) is the key entry to the web and tool for accessing information. A user writes a product name (query) in the query interface, and the query is send to a search engine to search the web data sources. The results of the query and web documents are saved in folders by the query interface as HTM files. Example of a simple query is shown in Figure 2.

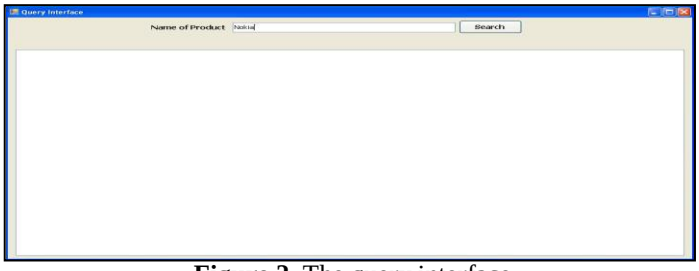


Figure 2. The query interface

3.2. Information Extraction (IE)

The Information Extraction (IE) component is the major component in the proposed framework. The aim of this component is to extract relevant information from the web pages obtained from QI based on standard classification scheme (SCS), classify the extracted information, and convert the extracted and classified information into structured information. Figure 3 illustrates examples of the results of a query and the web documents which are stored in folders. The IE extracts and classifies the web pages that are stored in the folders, and converts them into text form. The details steps are discussed below.

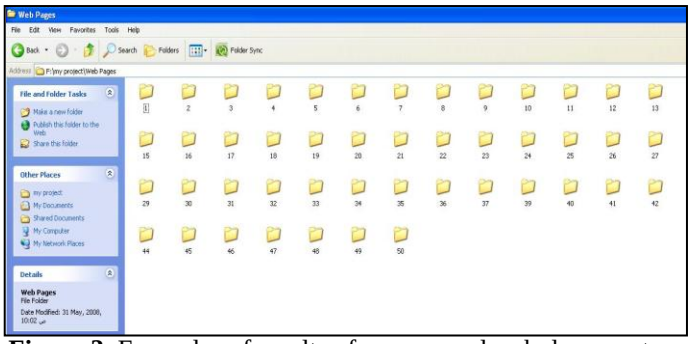


Figure 3. Examples of results of a query and web documents

Step 1 Pre-processing: The process of pre-processing consists of two steps, namely: (i) determine relevant web pages and (ii) prepare tokens.

(i) Determine relevant web pages: Not all of the web pages that are received from QI are related to user's request. Therefore, IE determines relevant web pages by analyzing the title of the web pages. IE checks the title of each web page (HTM files as shown in Figure 4) by comparing the tokens which are found between the tag <TITLE> and </TITLE> against the query as submitted by the user (i.e., keywords). The tokens are saved in an array and the HTML codes are ignored in this step.

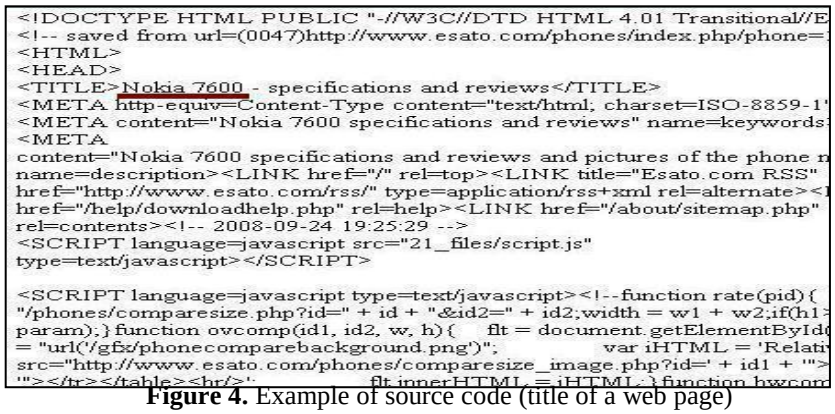


Figure 4. Example of source code (title of a web page)

(ii) Prepare tokens: After IE checks the title of a web page, IE saves the tokens that are found between the tag <TABLE> and </TABLE> in an array for matching them with SCS. Many of the web pages that the corporations used to announce their products are programmed using the tag <TABLE>. The tag <TR> denotes the row of <TABLE>, the tag <TD> denotes the field of <TR>, and the tag <TH> denotes the table header. If there is more than one tag <TD> and there is no HTML codes between the tags <TD> and </TD>, then IE saves the set of tokens that appears between the first tags <TD> and </TD> and prefix this set with the symbol "-" which denotes a sub-attribute (WP), and symbol ":" which denotes the value of a sub-attribute (WP) for the set of tokens that appears between the second tags <TD> and </TD>. If there is only one <TD> in one of <TR> or <TH> and there is no HTML codes between the tags <TD> and </TD>, then IE saves the set of tokens that appears between the tags <TD> and </TD> and prefix this set with the symbol "*" which denotes an attribute (WP). Figure 6 illustrates the sub-attributes and values of the sub-attributes found in Figure 5 that have been saved with the symbols "-" and ":" in an array.

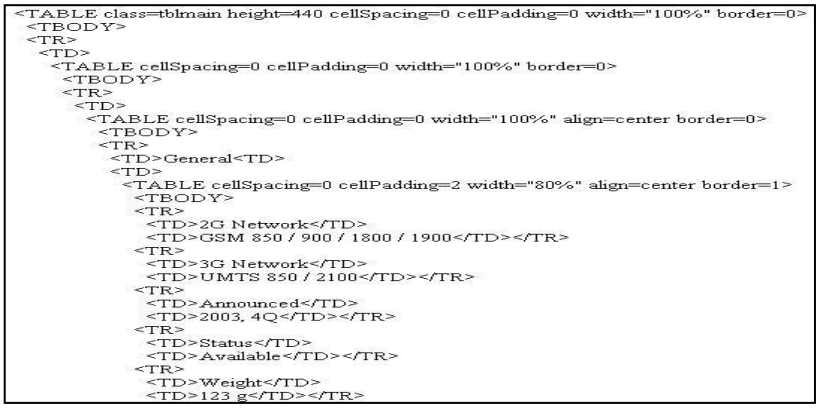


Figure 5. Example of the source code of a web page

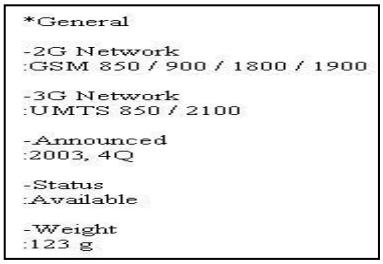


Figure 6. The sub-attributes (WP) and values of sub-attributes (WP) shown in Figure 5 saved in an array

Step 2 Extraction: The process of extracting information consists of two steps, which are (i) extract based on the attribute and (ii) extract based on the sub-attribute.

(i) Extract based on attribute: Information Extraction matches the tokens that are saved in the array with Attr (DB) and matches the Sub_Attr (WP) with Sub_Attr (DB) for extracting these tokens. IE saves the Attr (WP) in a text file as well as the Sub_Attr (WP) with its value. In this step, three cases are considered, namely *exact match*, *overlapping*, and *mutually disjoint*. There are cases where no match is found between the set of tokens saved in the array and Attr (DB). In this case, IE matches the set of tokens with Sub_Attr (DB) as shown in the next step.

(ii) Extract based on sub-attribute: In this step, two tasks are performed, namely: (i) extract set of tokens by matching with Sub_Attr (DB) and (ii) extract G_Sub (WP) by matching with each G_Sub (DB). In some of the web pages, the sub-attribute appears as attribute. Therefore, IE matches the set of tokens with Sub_Attr (DB). If there is a match, then IE saves the set of tokens as a sub-attribute together with its value in a text file. If there is no match among the set of tokens saved in the array and DB (Attr (DB) and Sub_Attr (DB)), then IE matches G_Sub (WP) with each G_Sub (DB) for extracting G_Sub (WP) as well as Attr (WP).

Step 3 Classification: After performing the extraction process, IE classifies the extracted information. Two steps are performed in this process, namely: (i) identify the index number of Attr (DB) that is matched and (ii) group the extracted attributes and sub-attributes based on the index number.

(i) Identify the index number of Attr (DB) that is matched: Information Extraction extracts the Attr (WP), Sub_Attr (WP), and value of Sub_Attr (WP) and later identifies the index of Attr (DB) that is matched. Figure 7 illustrates the example of the attributes that are saved in database with the index number *Index_no*.

	Attribute	Index_no
⊞	General	1
⊞	Size	2
⊞	Display	3
⊞	Ringtones	4
⊞	Memory	5
⊞	Data	6
⊞	Features	7
⊞	Battery	8

Figure 7. Attr (DB) saved in database, *Index_no* denotes the index of Attr (DB)

Figure 8 illustrates the example of sub-attributes and values of the sub-attributes, where each line begins with the index of Attr (DB) that is matched. For example, IE saves the sub-attribute weight with the index of the attribute *Size*.

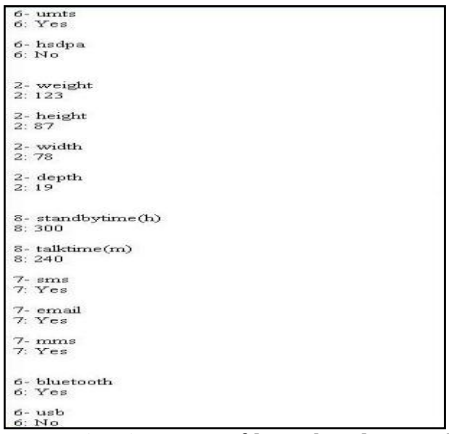
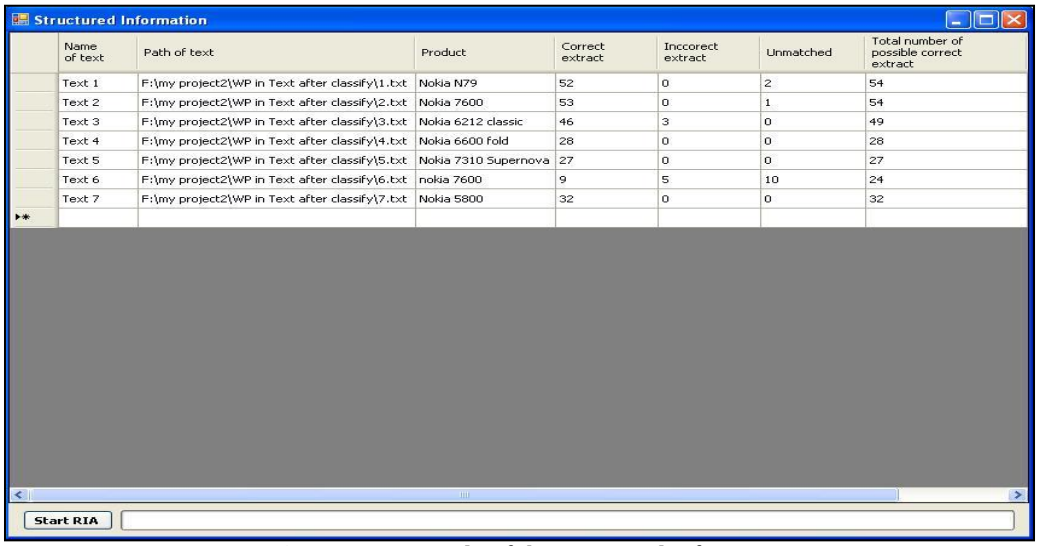


Figure 8. Attr (WP) in a text file with index number of Attr (DB)

(ii) Group the extracted attributes and sub-attributes based on the index number: The matched attributes and sub-attributes are then grouped based on the index number. For example, the lines with the index 6 are grouped together as attribute *DATA*. Figure 9 illustrates the example of the extracted attributes and sub-attributes that are shown in Figure 8 after grouping them based on the index number. In Figure 9, the symbol "*" denotes Attr (WP), the symbol "-" denotes Sub_Attr (WP), and the lines without the symbols "*" and "-" represent the value of Sub_Attr (WP). Next, the name of the text file, path of the text file, name of product, the number of correct extract, incorrect extract, and unmatched are saved in a table called *Structured Information*. Figure 10 illustrates an example of the Structured Information.

Figure 9. Attr (WP), Sub_Attr (WP), and value of Sub_Attr (WP) in a text file after grouping



Name of text	Path of text	Product	Correct extract	Inccorect extract	Unmatched	Total number of possible correct extract
Text 1	F:\my project2\WP in Text after classify\1.txt	Nokia N79	52	0	2	54
Text 2	F:\my project2\WP in Text after classify\2.txt	Nokia 7600	53	0	1	54
Text 3	F:\my project2\WP in Text after classify\3.txt	Nokia 6212 classic	46	3	0	49
Text 4	F:\my project2\WP in Text after classify\4.txt	Nokia 6600 fold	28	0	0	28
Text 5	F:\my project2\WP in Text after classify\5.txt	Nokia 7310 Supernova	27	0	0	27
Text 6	F:\my project2\WP in Text after classify\6.txt	nokia 7600	9	5	10	24
Text 7	F:\my project2\WP in Text after classify\7.txt	Nokia 5800	32	0	0	32

Figure 10. Example of the structured information

3.3. Relevant Information Analyzer (RIA)

Relevant Information Analyzer (RIA) analyzes and simplifies the relevant information extracted and classified from Information Extraction (IE). RIA identifies the attributes and sub-attributes that

belong to the same product that are extracted repetitively and compares them to remove the repetitive attributes and sub-attributes. Three steps need to be considered, namely: (i) Analyzing, (ii) Simplification, and (iii) Presentation.

(i) Step 1 Analyzing: RIA analyzes the extracted and classified information from IE by grouping the attributes and sub-attributes that are extracted repetitively belonging to the same product. The name of the product is saved in the *Structured Information* table. RIA groups the records in the *Structured Information* table based on the name of the product. Those records with the same product name are saved in the same table (*Similar_table*). For example, there are two text files in Figure 10, which are *Text* 2 consisting of 53 correct extract, and *Text* 6 consisting of 9 correct extract for the same product Nokia 7600. *Text* 2 and *Text* 6 are then saved in the same table by RIA.

(ii) Step 2 Simplification: RIA simplifies the analyzed information by comparing the sub-attributes that belong to the same product and removes the attributes and sub-attributes that are duplicates. For example, refer to *Text* 2 and *Text* 6 shown in Figure 10. RIA compares the sub-attributes of *Text* 2 and *Text* 6. *Text* 2 consists of 53 correct extract, while *Text* 6 consists of 9 correct extract which are found to be part of the extracted sub-attributes of *Text* 2. Therefore, RIA removes *Text* 6.

(iii) Step 3 Presentation: RIA presents the simplified extracted information, which is the name of the text file, web page, name of product, the extracted attributes, the extracted sub-attributes that describe the extracted attributes, and the values of the sub-attributes. Figure 11 illustrates an example of the extracted information.

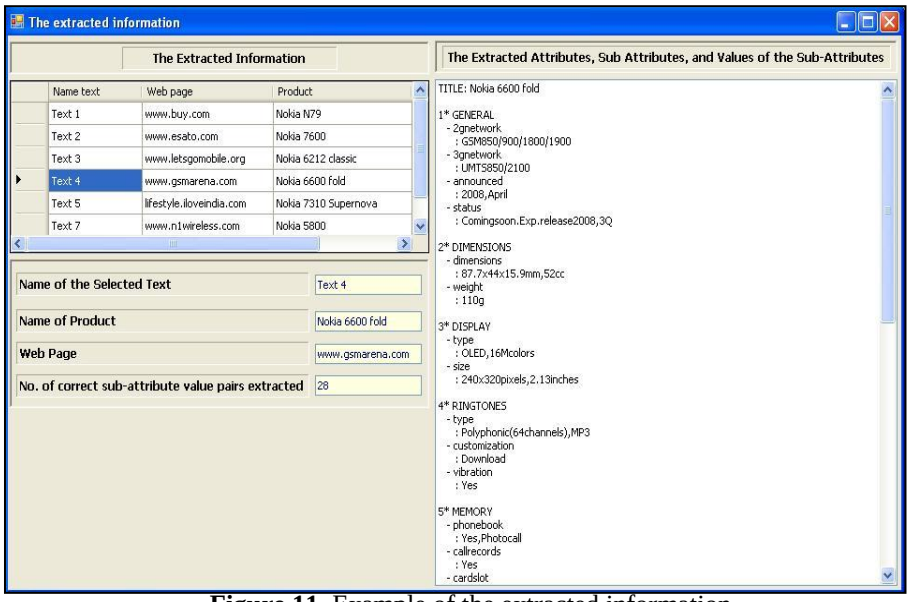


Figure 11. Example of the extracted information

4. Conclusion

In this paper, we proposed a framework to search and extract information from different web data sources. The proposed framework provides facilities to the user during search. A user does not need to visit the homepages of companies to get the information about any product, just write the name of product in the search engine and the framework searches all the available web pages related to the text which the user writes in the search engine, and the user gets the information with little efforts.

5. References

[1] Guntis Arnicans and Girts Karnitis, "Intelligent Integration of Information from Semi-Structured Web Data Sources on the Base of Ontology and Meta-Models", In Proceedings of the 7th International Baltic Conference, pp. 177-186, 2006.

[2] Sung Won Jung, Kyung Hee Sung, Tae Won Park, and Hyuk Chul Kwon, "Intelligent Integration of Information on the Internet for Travelers on Demand", In Proceedings of the ISIE, IEEE International Symposium, Vol. 1, pp. 338-342, 2001.

[3] D. Beneventano and S. Magnani, "A Framework for the Classification and the Reclassification of Electronic Catalogs", In Proceedings of the 2004 ACM Symposium on Applied Computing, pp. 784-788, 2004.

[4] Srinivas Vadrevu, Fatih Gelgi, and Hasan Davulcu, "Information Extraction from Web Pages using Presentation Regularities and Domain Knowledge", Journal of World Wide Web, Springer Netherlands, Vol. 10, Issue. 2, pp. 157-179, 2007.

[5] Horacio Saggion, Adam Funk, Diana Maynard, and Kalina Bontcheva, "Ontology-based Information Extraction for Business Intelligence", In Proceedings of the 6th International Semantic Web Conference and the 2nd Asian Semantic Web Conference, Vol. 4825, pp. 843-856, 2007.

[6] Fatima Ashraf and Reda Alhajj, "ClusTex: Information Extraction from HTML Pages", In Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'O7), pp. 355-360, 2007.

[7] Bernard J. Jansen, Danielle L. Booth, and Amanda Spink, "Determining the User Intent of Web Search Engine Queries", In Proceedings of the WWW International Conference, pp. 1149, 2007.

[8] Paul Viola and Mukund Narasimhand, "Learning to Extract Information from Semi-Structured Text using a Discriminative Context Free Grammar", In Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 330-337, 2005.

[9] Gilles Nachouki, "A Method for Information Extraction from the Web", Journal of Information and Communication Technologies, Vol. 1, pp. 517-521, 2006.

[10] Utku Irmak and Torsten Suel, "Interactive Wrapper Generation with Minimal User Effort", In Proceedings of the 15th International Conference on World Wide Web, pp. 553–563, 2006.

[11] Hongkun Zhao, Weiyi Meng, and Clement Yu, "Automatic Extraction of Dynamic Record Sections from Search Engine Result Pages", In Proceedings of the 32nd International Conference on Very Large Data Bases, pp. 989-1000, 2006.

[12] Hongkun Zhao, Weiyi Meng, and Clement Yu, "Mining Templates from Search Result Records of Search Engines", In Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 884-893, 2007.

[13] Fei, Hong, Zhuang, and Zhao, "Information Extraction System in Large-Scale Web", Journal of Communications and Information Technology, ISCIT, Vol. 2, pp. 809- 812, 2005.

[14] Shady Shehata, Fakhri Karray, and Mohamed Kamel, "Enhancing Search Engine Quality using Concept-based Text Retrieval", In Proceedings of the International Conference on Web Intelligence (WI07), pp. 26-32, 2007.

[15] Shaker M., Ibrahim H., Mustapha A., and Nurliyana L. "A Framework for Extracting Information from Semi-Structured Web Data Sources". In Proceedings of 3rd 2008 International Conference on Convergence and Hybrid Information Technology, Vol. 1, pp. 27-31, 2008.