Transformation from Web Pages to Optimal Normal Form Database Schema Using a Conceptual Schema Approach

Oviliani Yenty Yuliana
Faculty of Business and Economics
Petra Christian University
Surabaya, Indonesia
oviliani@petra.ac.id

Suphamit Chittayasothorn
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand
suphamit.ch@kmitl.ac.th

***Abstract*— Web pages and their embedded documents are a good source of information. However, issuing complex queries directly on web pages using direct pattern-matching techniques are challenging tasks that require lengthy procedural programming. Also, programmers working directly on web pages are in charge of the search path and routing, which depend on individual document structures and affect the correctness, completeness, and performance of the results. In contrast, relational databases are well-structured and backed by mathematical principles. Well-designed relational database structures are known to be anomalies-free. The standard relational database language, SQL, is a non-procedural language that defines the required results precisely. Query results are both correct and complete. Performance issues are handled by intelligent query optimizers employed by modern-day Database Management System (DBMS). This paper suggests an approach that transforms documents embedded on web pages in HTML format to corresponding relational database structures and populations. Functional dependencies (FDs) and multi-valued dependencies (MVDs) obtained from documents on the webpages are used to construct conceptual schema diagrams, which are further transformed into the Optimal Normal Form (ONF) relational database structures. In this research project, the Object Role Model (ORM) conceptual schema model is employed. The paper discusses the ORM and the rationales behind its usage. The detection of FDs and MVDs from webpage documents and the technical properties of the ONF relational database structures. Illustrated examples are also provided.**

Keywords*— *webpage document, transformation, conceptual schema, multivalued dependencies, optimal normal form

I. INTRODUCTION

In the era of big data, the business environments are increasingly complex, challenges, and unpredictable conditions. Business competition in globalization is also getting tighter; business competitors come globally. To be able to survive, businesses cannot rely solely on intuition. These problems can be overcome with Business Intelligence (BI) [1] and Data Analytics. BI collects data from various sources and formats, then processes the data into accurate and useful information to assist decision-makers. In other words, we call the businesses in a data-driven decision support system to exist in the global competition.

Businesses need data sources from outside the company in order to compete with other businesses. External data sources can be structured, semi-structured, and unstructured. The structured data is relatively easy to integrate into relational databases and data warehouses. However, the outside data are not free from errors, different data formats, or different field names from the business's data structure. Therefore businesses need a tool called extract-load-transform (ELT) or extract-transform-load (ETL) [2, 3]. The two terms are often interchanged; in this paper, we use the term ETL. The role of ETL is also needed to extract semi-structured data and unstructured data. The successfully transformed data into a relational database or data warehouse can be retrieved easily using queries to reuse, such as price comparison and analysis data.

Researches related to extracting data from flat/text files have been conducted for several decades. Data extraction from semi-structured data, such as CSV files, have been conducted by many researchers, such as [4-6]. Research on ETL to extract and transform other semi-structured data, such as XML as a language for data interchange, into relational databases have been conducted, e.g., [7, 8]. However, the ETL to extract and transform semi-structured, i.e., XML data, into the optimal normal-schema database using the conceptual schema approach were conducted by [9-11]. With the internet of things in Industry 4.0, more and more data/information will be published on the internet. Web pages, webs for short, will be the best potential data source for BI. Many web data extraction researches have been conducted before the Industrial 4.0 era [12-20].

In general, web pages are divided into two types, namely Surface webs and Deep/Dark webs. Surface webs contain extended text information (i.e., unstructured data), and the webmaster updates the information. They call static webs and ware surveyed by [12]. Surface webs commonly used by News websites, e.g., the Bangkok Post in Fig. 1. Google has indexed the Surface webs so that Google can search the information on Surface webs. It is about 10%[1] of the entire webs. In other words, the rest of the webs, around 90%[2], are Deep/Dark webs [13-20]. Google could not index the Deep/Dark webs information, so Google cannot search for the Deep webs. A Deep web example is shown in Fig. 2, and it contains semi-structured data, such as title, authors, format, price, ratings of book. It requires an application to find data from a database based on a user's keyword, a red box at the top of Fig. 2. A web browser formats and embeds the retrieved data into a template dynamically for displaying the information, so the Deep webs call dynamic webs. These mechanisms make the information on Deep webs are full of

[1] https://en.wikipedia.org/wiki/Surface_web

[2] https://www.kratikal.com/blog/surface-web-and-dark-web-exploring-layers-of-web/

978-0-7381-4484-9/21/$31.00 ©2021 IEEE

noise. We cannot use queries directly on Deep Webs, which is a challenge and contribution of this paper. For the next discussion, web pages means Deep webs.

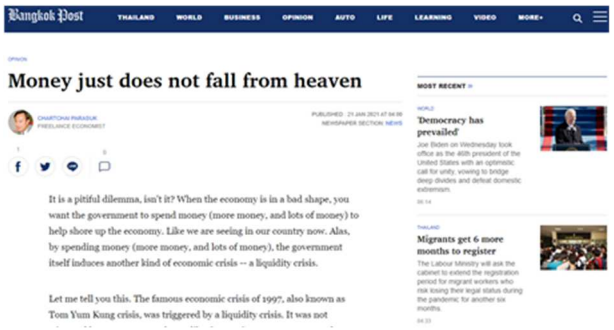


Fig. 1. Surface web.

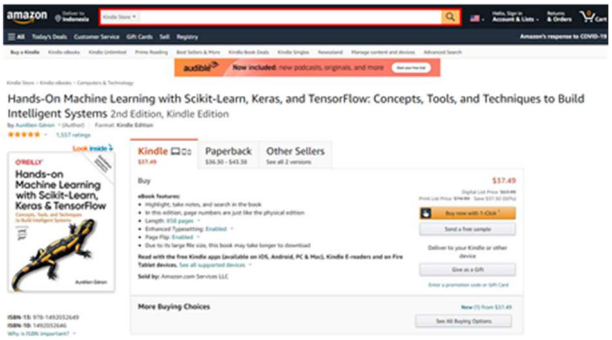


Fig. 2. Deep web.

DCADE [20] algorithm outperformed state-of-the-art methods [16, 17]. It succeeded in extracting attribute-value pairs, optional data, and repetitive pattern data (i.e., functional dependencies). Also, it separates each repetitive pattern into other tables. However, DCADE has not yet detected whether multi-valued dependencies were found in the extracted data. This paper aims to (1) adopt DCADE for extracting data from the Deep webs and (2) transform the data into optimal normal schema database form using a conceptual schema approach.

The remainder of this paper is organized as follows. Section 2 provides a comparison with related work on web data extraction techniques. Section 3 describes document extraction, and Section 4 presents conceptual schema and transformation to relational database structures. Section 5 concludes the paper and proposes future work.

II. Literature Review

Researches on data extraction from the Deep web have been conducted by [13-20]. They are differentiated based on the number of web page inputs. Researches on data extraction using one web page input were conducted by [13-15], in general they used a repeating structure of HTML tags, such as tables (<table>, <tr>, <th>, and <td>) and list (and <li>). For example, consider a conference schedule in Fig. 3 on the left side. A table (<table>) contains four columns, i.e., Speaker, Topics, No Sessions, and Manual Read. The table contains five rows tagged by <tr>, and the first line is the title of the table tagged by <th>, while data rows are tagged by <td>. With the HTML repetitive structure detection, the Deep web's data can easily be extracted. Web designers use tables or list HTML structures to display data, but they also use them for formating data on the web. However, three data on the above table, they call attribute-value pairs, will not be extracted as data by [13-15] because they do not tag as tables or lists.

Other researchers [16-20] use several Deep webs as inputs to confirm information not tagged by tables and lists to solve the previous problem. Those researchers use how the data are embedded into a template and formatted to display. The assumption is that the templates on web pages will be the same, the data are different, and the formatting patterns of data on the web are made consistent. For example, see the two Deep webs in Fig. 3. The first line on both webs is the same as the string "computer Science Conference," it is the template. The second lines on both webs are found with the same string, "Date," which acts as the template (attribute). Also, "20/02/2021" and "21/02/2021" are found on the left and right webs, respectively. The two strings are different, which are interpreted as the value of Date. The Attribute-value pair Time is optional, and it is only in the left schedule in Fig. 3. The fifth line and the fourth line for left and right webs contain the same string for the table headings. Researchers [16, 17] will discard templates and formats when deep webs are parsed with this assumption. In other words, the remaining information is extracted data. From the discussion [20], [16, 17] did not consider optional data, the method [16] was not robust, and performance [17] was not high due to not using regular expressions, so there found false positive and negative data extraction.

Computer Science Conference

Date: **20/02/2021**
Time: **13:00-15:00**
Room: **AV 501**

Speaker	Topics	No Sessions	Manual Read
John	WordProc	3	Object Programming
	D++	1	Adv WordProc
			Database Book
Tom	DB9	3	Database Book

Computer Science Conference

Date: **20/02/2021**
Room: **AV 305**

Speaker	Topics	No Sessions	Manual Read
Peter	WordProc	4	Into to WordProc
	SuperOS	2	Adv WordProc
			Operating Systems
Rob	D++	3	Database Book
			Operating Systems

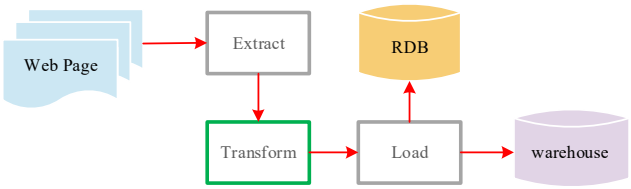
Fig. 3. Two example deep web of conference.

Researches [18-20] used a different method, which encodes HTML paths and contents, then mining the mandatory template first. The template is used to segment webs and solves problems for each segment. Mine optional and repetitive templates/data, and then use regular expressions to align data in a segment. The method is called divide and conquer alignment with dynamic encoding for full-page data extractions (DCADE) [20]. The experimental results show that DECADE outperforms other techniques. Therefore, this paper adopts DCADE, which will be explained in more detail in the next session. However, the null values in the Speaker, Topics, or No Sessions column in the table in Fig. 3 do not mean optional data. They are multi-valued, and the multi-value dependency has still not been resolved by DCADE.

III. Web Document Extraction

The proposed ETL framework in this paper is shown in Fig. 4. Web data extraction using DCADE [20] will be explained briefly in this section, transforming the extracted data into an optimal normal schema database using the conceptual schema approach will be discussed in the next section. Once we have obtained this schema, loading the data into a relational database or data warehouse becomes obvious beyond the paper scope.

For optional attribute-value pair, repetitive pattern, and robust extraction, first DCADE encodes HTML tags, CSS attributes (i.e., class and id), and HTML contents. As we



know, HTML tags (PathId) are generally used for formatting data on a web page. DCADE encodes PathId with HTML content into a content-equivalent class (CECId) for template mining. Meanwhile, to mining data patterns, DCADE encodes PathId, CSS attributes, and an encoded regular-expression of content into typeset equivalence class (TECId).

In the next step, DCADE mines mandatory templates recursively on web pages based on CECId (i.e., similar Path and equal Content). This technique is based on how the web pages are formed and embedded the data into the template. With the discovery of templates, extracting data is very helpful. DCADE uses mandatory templates to segment web pages to make it easier to solve data extraction problems. Each segment will be marked by a mandatory template known as a landmark, which will be used in the next step. For example, see Fig. 3, the landmarks are Computer Science Conference, Date, Room, Speaker, Topics, No Sessions, and Manual Read.

In each segment where there is more than one gap between two segments, DACDE uses CECId and TECId to do pattern mining. See Fig. 3, more than one segment gap is the segment between Date and Room, and the segment between Manual Read to the end of the website. This step aims to (1) find the optional template and (2) frequent pattern alignment to find the repetitive pattern. If the number of CECId from all web pages meets a given threshold, then the template is declared an automatic template. For example, in Fig. 3, Time is an optional template. There is no repetitive template (CECId) in the picture, but we find the repetitive data (TECId) of a table. In other words, we find the data region of a rich format data of a two-dimensional table.

The final step of DCADE, re-arrange columns and split the table. Based on the previous step's data regions, DCADE uses the repetitive data pattern as a table data structure (record boundary). Each row in a table has the same data structure. To extract data that may contain optional data, DCADE uses multi-string alignment to place data according to its TECId in the correct data structure. Finally, DCADE will split each repetitive pattern into a separate table and keep a pointer from the main table into the separated table.

From the explanation above, we can see that DCADE has successfully mined mandatory and optional templates based on CECId. DCADE has also succeeded in aligning data according to its data structure based on TECId. Besides, DCADE managed to find a repetitive pattern template/data, and the data was stored in a separate table. However, DCADE has not considered multi-valued dependencies (MVD), as shown in Fig. 3. For a while, every empty cell is considered optional data. In the next section, we will explain how to transform the extracted data to optimal normal form database schema using a conceptual schema approach.

IV. Conceptual Schema and Transformation to Relational Database Structures

Information systems in the organization are good sources of information. Data from operational information systems are collected and summarized so that queries from the management or users can be accommodated. This information from the internal sources is considered essential, especially for operation-level management. In contrast, for higher-level management, external data and documents from web pages play essential roles. Web documents can be searched using pattern-matching techniques. Web crawlers extract data from the web mainly by matching keywords but lack the ability to answer complex queries that database languages have.

The relational database language SQL (Structured Query Language) is the most widely used database language to date. It is a declarative, non-procedural language that defines the required result. SQL contrasts with other database languages, allowing the programmers to find the best way to obtain the result. SQL is a high productivity language that can handle complex queries. It is very popular for querying and manipulating databases. One of the properties that make it the most widely used database language to date is the ability to formulate ad hoc queries in the stand alone interactive mode without the need to be embedded in other programs written in other programming languages. Documents extracted from web pages described in the previous sections can be further analyzed to detect relationships between data items on the web pages and construct a conceptual schema diagram. The conceptual schema diagram is then transformed into relational database schemas. We employ the Object Role Model (ORM) conceptual schema model [21], which has a well-defined transformation algorithm that guarantees the Optimal Normal Form (ONF) relational database schemas.

After the invention of the logical database models, including the relational database model in the late sixties, researchers were interested in finding a higher-level data model that could capture more semantics. The ORM was invented as a conceptual model. It is easy to understand and use to describe the application's Universe of Discourse (UoD). ORM is a fact-based conceptual model. The ORM model is based on the concepts of entity type and relationship type. Additional concepts are label types and some additional integrity constraints. Unlike other conceptual data models, the concept of an attribute is completely eliminated. In conceptual modeling, it has been long debated on the difference between entity types and attributes. It is one of the confusing issues for inexperienced data modelers. The ORM model, therefore, retains only the concept of entity types.

An entity or entity instance is defined as an object of interest. Each entity instance represents a real-world object instance. Entity instances of the same type belong to an entity type. The term entity refers to an entity instance. Each label instance represents a name (a label) or a value, not an actual object. Label instances of the same type belong to a label type. The term label refers to a label instance. Entity types and label types are collectively called object types. The actual object is an entity instance, but a label instance is a name or a value. The ORM clearly separates these two concepts. In the ORM conceptual schema notation, an entity type is represented by a solid ellipse, and a dotted ellipse represents a label type. For identification purposes, an entity type must have at least a label type associate with it. A relationship between an entity type and a label type is called a reference type. Role boxes

symbolize each reference type, and each role denotes the participation of an object type. The arrow on top of the role box denotes its unique participation. It is called a uniqueness constraint, accordingly. An entity type must have a unique label type as its unique identifier as a 1:1 reference type.

Once label type(s) and reference type(s) of an entity type is specified. The next step is to relate entity types together. Relationships between entity types are called fact types. A fact type in ORM is the most basic relationship type between entity types. Fact types must be elementary, meaning that the fact type must not be decomposable. This is due to the fact that an ORM relationship type (fact type, reference type) is a representation of a deep-structured natural language sentence type. It comprises a subject, a verb, and an optional object.

The ORM model is well equipped with detailed transformation algorithms that guarantee 5NF relational table structures [22]. In fact, the ORM group of researchers called them the ONF transformation algorithm. The ONF is defined as 5NF, which has a minimum number of relational table structures. Early ONF transformation has nine steps, but more recent ones go to the detailed fourteen steps [21]. Those steps are for people who have no knowledge about the normalization process but need a cookbook approach to jump directly to the 5NF without knowing normalization. In fact, the ORM model is intended to be used by those who have a limited database design background.

According to original database design references on 5NF, we can transform ORM diagrams into 5NF relational table structures in only two steps. It is due to the fact that an ORM relationship type (fact type, reference type) is a representation of a deep-structured natural language sentence type. It comprises a subject, a verb, and an optional object. All relationship types in ORM must therefore be elementary, meaning that they must not be decomposable. The implication on this non-decomposable property of ORM object types is very strong. If each of them is considered a relational table structure, it is already in the 5NF since it cannot be split. These small relational table structures can then be joined back by their common primary keys or even candidate keys to form the bigger 5NF relational table structures.

The two-step transformation from an ORM conceptual schema to relational table structures is therefore as follows:

Step 1: Transform each relationship type into a relational table structure. Each attribute is from the unique identifier or label type of the participating object type. The primary key attribute(s) of the relational table structure is from the unique identifier or label type under a uniqueness constraint.

Step 2: Join relational table structures that share the same primary key by using the natural join.

An important note here is the above transformation steps are actually the synthesis or bottom up approach of relational database design. Small, trouble-free 5NF relational table structures are combined back to bigger 5NF relational table structures. The result is a set of 5NF relational table structures with the smallest number of table structures; thus, the ONF.

After the extraction of web pages, two documents are obtained. The first document Software has only functional dependencies (FDs, which is symbolized by →) between data items. FDs Software Name → Vendor, Software Name → Price, and Manual → Software Name can be detected from the Software document. From the Conference document, Multivalued Dependencies (MVDs, which is symbolized by →→) [23] Speaker →→ Manual, and Speaker →→ (Topics, No Sessions) can also be detected. An FD is an m:1 relationship between data items. MVDs are m:n relationships between data items. Based on these dependencies, an ORM diagram that describes the document in Fig. 5 can be constructed as shown in Fig. 6.

Software

Software Name	Vendor	Price (USD)	Manual
WordProc	NanoSoft	400	Into to WordProc
			Adv WordProc
DB9	ABC Corp.	300	Database Book
SuperOS	TTT	400	Operating Systems
D++	Khrisna	350	Object Programming
			D++ Reference

Conference

Speaker	Topics	No Sessions	Manual
John	WordProc	3	Object Programming
	D++	1	Adv WordProc
			Database Book
Tom	DB9	3	Database Book
Peter	WordProc	4	Into to WordProc
	SuperOS	2	Adv WordProc
			Operating Systems
Rob	D++	3	Database Book
			Operating Systems

Fig. 5. Two documents on software products and conference talks which are extracted from webpages.

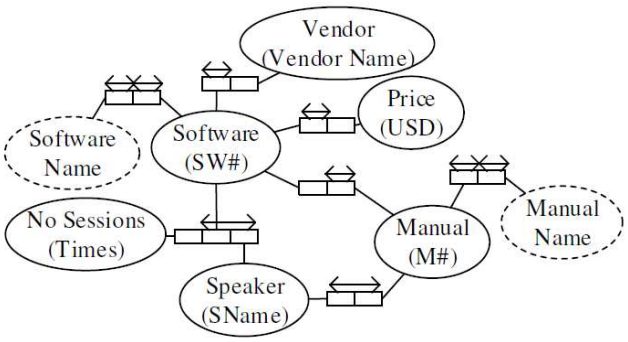


Fig. 6. The ORM conceptual schema which describes entity types, label types, and relationship types of the documents in Fig. 5.

In Fig. 6, the ORM conceptual schema has six entity types: Software, Vendor, Price, No Sessions, Manual, and Speaker. It can be detected that the value instances of Manual from the Conference document are a subset of the value instances of Manual from the Software document. They, therefore, belong to the same entity type Manual. Also, the value instances of Topics from the Conference document are a subset of the value instances of the Software Name of the Software document. They, therefore, belong to the same entity type Software. For internal references, a surrogate unique identifier SW# is introduced for the entity type Software, and a unique surrogate identifier M# is introduced for the entity type Manual.

The corresponding relational database schemas, as shown in Fig. 7, are obtained after applying the two transformation steps to the ORM conceptual schema model in Fig. 6. Each relationship type of the conceptual schema can be perceived as a non-splittable relational schema in 5NF. Relationship instances of each relationship type are obtained from the documents. Thus, non-splittable 5NF tables are obtained. Common primary keys or candidate keys of the 5NF tables are detected, and the tables with common primary or candidate keys are then joined together to obtain the ONF tables. These final tables are ready to be used for querying using a complete relational language or data warehouse operations.

Each relationship type of an ORM diagram represents a deep structured natural language statement type which cannot be further decomposed. This is intentional. Each relationship instance is, therefore, a simple sentence that can be directly

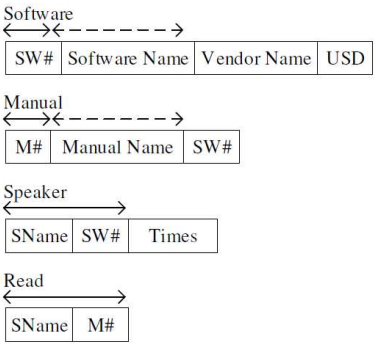


Fig. 7. Corresponding relational database schemas transformed from the ORM diagram in Fig. 6 based on the 2-step transformation.

obtained from the document. Fig.8 shows populated tables which are obtained from the extraction. Each row of the Manual, Read, and Speak tables represents one fact instance. Each row of the Software table comprises three fact instances. All tables are in the Optimal Normal Form (ONF), which means they are in 5NF with the minimum number of database tables. Complex SQL queries can be applied to these tables. Our transformation from web pages to optimal normal form database schema using a conceptual schema approach is useful for interactive requests and applications development.

Software

SW#	Software Name	Vendor Name	USD
S01	WordProc	NanoSoft	400
S02	DB9	ABC Corp.	300
S03	SuperOS	TTT	400
S04	D++	Khrisna	350

Manual

M#	Manual Name	SW#
M01	Into to WordProc	S01
M02	Adv WordProc	S01
M03	Database Book	S02
M04	Operating Systems	S03
M05	Object Programming	S04
M06	D++ Reference	S04

Speaker

SName	SW#	Time
John	S01	3
John	S04	1
Tom	S02	3
Peter	S01	4
Peter	S03	2
Rob	S04	3

Read

SName	M#
John	M05
John	M02
John	M03
Tom	M03
Peter	M01
Peter	M02
Peter	M04
Rob	M03
Rob	M04

Fig. 8. The relational database tables with populated rows from the documents in Fig.5.

V. CONCLUSION

This paper suggests an approach that transforms documents embedded in HTML format to corresponding relational database structures and populations. The main motivation is the fact that relational database languages such as SQL can handle complex queries in a real-world business environment much better than web-based languages with higher productivity. Functional dependencies (FDs) and multi-valued dependencies (MVDs) obtained from documents on the web pages are used to construct conceptual schema diagrams, which are further transformed into the Optimal Normal Form (ONF) relational database structures. In this research project, the Object Role Model (ORM) conceptual schema model is employed. The paper discusses the ORM and the rationales behind its usage. The detected FDs and MVDs from web page documents construct a corresponding ORM conceptual schema model in the ONF relational database structures. Illustrated examples are also provided.

REFERENCES

[1] S. Negash and P. Gray, Business intelligence, Handbook on decision support systems 2, 2008, pp.175–193.

[2] S. K., Bansal, Towards a semantic extract-transform-load (ETL) framework for big data integration, 2014 IEEE International Congress on Big Data, 2014, pp.522–529.

[3] S. K., Bansal and S. Kagemann, “Integrating big data: A semantic extract-transform-load framework,” Computer, vol. 48, pp. 42–50, 2015.

[4] M. Arenas, F. Maturana, C. Riveros and D. Vrgoč, “A framework for annotating CSV-like data,” Proceedings of the VLDB Endowment, pp. 876–887, 2016.

[5] S. H., Mahmud, M. A., Hossin, H. Jahan, S. R. H. Noori and T. Bhuiyan, CSV-ANNOTATE: Generate annotated tables from CSV file, International Conference on Artificial Intelligence and Big Data (ICAIBD), 2018, pp. 71–75.

[6] C. Christodoulakis, E. B. Munson, M. Gabel, A. D. Brown and R. Miller, “Pytheas: pattern-based table discovery in CSV files,” Proceedings of the VLDB Endowment, vol. 13, pp. 2075–2089, 2020.

[7] H. Aman and R. Ibrahim, “Reverse engineering: from XML to Uml for generation of software requirement specification,” 8th International Conference on Information Technology in Asia (CITA), pp. 1–6, July 2013.

[8] B. Vo and H. T. Nguyen, “Mining frequent closed itemsets from multidimensional databases,” International Journal of Computational Vision and Robotics, vol. 5, pp. 217–230, August 2015.

[9] O. Y. Yuliana and S. Chittayasothorn, “XML schema re-engineering using a conceptual schema approach,” International Conference on Information Technology: Coding and Computing (ITCC'05), vol. 2, pp. 255–260, April 2005.

[10] O. Y. Yuliana and S. Chittayasothorn, “A conceptual schema based XML Schema with integrity constraints checking,” International Conference on Convergence and Hybrid Information Technology, pp. 19–24, August 2008.

[11] O. Y. Yuliana and S. Chittayasothorn, “Deriving conceptual schema from XML databases,” First Asian Conference on Intelligent Information and Database Systems, pp. 40–45, April 2009.

[12] S. Sarawagi, “Information Extraction”, Foundations and Trends in Databases, vol. 1, March 2008.

[13] F. Tim, G. Georg, G. Giovanni, G. Xiaonan, O. Giorgio, S. Christian and W. Cheng, “DIADEM: thousands of websites to a single database,” the VLDB, vol. 7, pp. 1845–1856, 2014.

[14] S. Shi, C. Liu, Y. Shen, C. Yuan and Y. Huang, “AutoRM: an effective approach for automatic Web data record mining,” Knowl-Based Syst, vol. 89, pp. 314–331, 2015.

[15] A. Omari, B. Kimelfeld, E. Yahav and S. Shoham, “Lossless separation of web pages into layout code and data,” the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp. 1805–1814, 2016.

[16] V. Crescenzi and G. Mecca, “Automatic information extraction from large websites,” Journal of the ACM (JACM), vol. 51, pp. 731–779, 2005.

[17] H. A. Sleiman and R. Corchuelo, “Tex: an efficient and effective unsupervised web information extractor,” Knowl-Based Syst, vol. 39, pp. 109–123, 2013.

[18] O. Y. Yuliana and C.-H. Chang, “AFIS: aligning detail-pages for full schema induction,” Conference on Technologies and Applications of Artificial Intelligence (TAAI), pp. 220–227, November 2016.

[19] O. Y. Yuliana and C.-H. Chang, “A novel alignment algorithm for effective web data extraction from singleton-item pages,” Applied Intelligence, vol. 48, pp. 4355–4370, June 2018.

[20] O. Y. Yuliana and C.-H. Chang, “DCADE: divide and conquer alignment with dynamic encoding for full page data extraction,” Applied Intelligence, vol. 50, pp. 271–295, July 2019.

[21] T. A. Halpin and T. Morgan, Information Modeling and Relational Databases, 2nd ed., The Morgan Kaufmann Series in Data Management Systems, 2008.

[22] R. Fagin, “Normal forms and relational database operators,” Proceedings of the 1979 ACM SIGMOD international conference on Management of data, pp. 153–160, May 1979.

[23] R. Fagin, “Multivalued dependencies and a new normal form for relational databases,” ACM Transactions on Database Systems (TODS), vol. 2, pp. 262–278, September 1977.