International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 1 Issue 3 ǁ July. 2013 ǁ PP.16-26

Web Services Based Integration Tool for Heterogeneous Databases

Amin Noaman, Fathy Essia, Mostafa Salah
Faculty of Computing & Information Technology, King Abdulaziz University

Abstract: In this paper we introduce an integration system that consists of two subsystems (tools): integration sub-system (tool) and query (sub-system) tool. The integration tool has been built for integrating data from different data stores (databases) that were created with different database engines. The query sub-system (tool) has been built to help a user to query in a structured natural language or structured query language. The integration system has been built based on the web services technology to be adaptable, reusable, maintainable, and distributed. The integration subsystem collects data from heterogeneous data sources, unifies them based on ontology and stores the unified data in a data warehousing, which its schema is generated automatically by the tool. The integration tool is a database engine independent, domain independent and based on ontology scheme. The query tool has been built to accept the requests from a user and manipulate data in the data warehouse and return the results to the user. The query tool generates queries automatically based on the user requirements and data warehouse schema. The user can write his query as structured natural language or structured query language. The system has been implemented and tested.

I. Introduction

The Web contains abundant repositories of information that make selecting just the needed information for an application a great challenge since computers applications understand only the Web pages structure and layout and have no access to their intended meaning. To enable users get information from the Web by querying a database there are two traditional approaches: to enhance query languages to be a Web aware; the other is virtually extraction Web pages with wrappers. The new alternative approach proposed by Embley [1] at Brigham Young University, data extraction group is the Semantic Web.

The Semantic Web aims to enhance the existing Web with a layer of machine-interpretable metadata. The American Heritage Dictionary defines semantics as "the meaning or the interpretation of a word, sentence, or other language form" Embley [1].

The emergence of the Semantic Web will simplify and improve knowledge reuse on the Web and will change the way people can access knowledge, agents will be a knowledge primary consumer. By combining knowledge about their user and his needs with information collected from the Semantic Web, agents can perform tasks via Web services [2] automatically. So agents can understand and reason about information and use it to meet user's needs. They can provide assistance using ontologies, axioms, and languages such as DARPA Agent Markup Language which are cornerstones of the Semantic Web.

Data interoperability occurs when an application can use data from one or more disparate data sources. With the amount of data being produced, stored, and exchanged in the world today, there are numerous situations for which achieving data interoperability is essential. For example: Multiple organizations with their own data storage schemas, such as regional educational services, might merge into one, larger organization and consolidate their data. Also, a head office may require its various organizations to submit annual performance data in a particular format; this format may change from year to year. Two separate organizations having data about a certain topic may wish to exchange or merge this data; however, they do not want to share private data about their employees and finances. Finally, a supplier may wish to exchange data with a manufacturer.

The common issue in these examples is that the data to be exchanged and/or integrated comes from separate sources that were developed independently. This means that the data might reside in completely different formats - for example, some data might be stored in a relational database, the other as XML files, even textual sources can provide data.

In addition, because each data schema is designed independently, these schemas will be different - even if they are expressed in the same data model (e.g. the relational data model) and describe the same domain.

In data integration, a mediated schema is used to provide a uniform query interface for multiple data sources. The mediated schema approach is often used in enterprise data integration, for example when various branches of the same organization merge. In this approach, the data stays in the individual source databases. Queries are expressed in terms of the mediated schema, while wrappers containing schema mappings between the source schemas and the mediated schema translate the queries and the results back and forth.

Other approaches, often used in Web applications, are: peer-to-peer data integration where pairwise mappings are made directly between a number of individual data sources, and the data exchange where mapping is created between a source and a target schema with the goal of moving all of the data from the source database to the target database.

Schema mappings represent a key to achieve data interoperability. It is a precise specification of the relationships between the elements of a source schema and the elements of a target schema. This specification makes it possible to transform data from the source schema to fit into the target schema. Executable schema mappings are schema mappings that can take an instance of a source schema and reform it to meet the syntax and integrity constraints of a target schema. The source and target schemas need not be in the same format; for instance, the source database might be a relational database while the target database could be stored in XML. Executable schema mappings can be expressed in any executable language that can be used to extract data from or input data into the databases, such as SQL, XQuery.

The schema matching involves finding correspondences between pairs of individual elements of the source and target schemas. Taking as input a source schema S and a target schema T, this step outputs a multimapping which consists of pairs of correspondences between elements of S and elements of T. (Where elements, in a relational database, are the attributes of relations). The methods used for this step use clues from the labels of the schema attributes [3], the structures of the schemas [4], and occasionally lexical comparisons to words present in external taxonomies of words [5]. The most effective schema matchers LSD [6] use a hybrid of these techniques. Even the best schema matchers do not achieve 100% accuracy – for example, Doan et al. [6] reported 71% - 92% accuracy for their hybrid matcher, LSD, and noted that two specific characteristics of schemas preventing the accuracy from being higher were: ambiguity in the meaning of labels, and being unable to anticipate every type of format for the data. These deficiencies in accuracy are propagated to the next step in schema mapping creation, mapping generation.

II. Related Work

Brend Amann et al. [7], proposed ontology mediator architecture for the querying and integration of XML data sources. Cruz et al. [8] proposed mediator to providing data interoperability among different databases. Also, Philipi et al. [9] introduced architecture for ontology-driven data integration based on XML technology.

Others presented some solutions to enhance the metadata representation as in Hunter et al. [10] by combining RDF and XML schemas, and Ngmnij et al. [11] by using metadata dictionary as for solving some semantic heterogeneity.

In solving some problems in the query processing, Baoshi et al. [12] presented a query translation approach, Corby et al. [13] addressed the problem of a dedicated ontology-based query language, Saleh [14] presented a semantic framework that addresses the query mapping approach.

In E. Mena et al. [15] OBSERVER is an approach for query processing in global information system. Yingge et al. [16] presented aSDMS system which utilizes software agent and Semantic Web technologies; they addressed the problem of improving the efficiency of information management across weak data. A data warehousing approach with ontology based query facility presented by Munir et al. [17].

Finaly, Al-Ghamdi, et al, [18], developed a software system based on ontology to semantically integrate heterogeneous data sources such as XML and RDF to solve some conflicts that occur in these sources. They used agent framework based on ontology to retrieve data from distributed heterogeneous data sources. They implemented this framework using some modules and libraries of Java, Aglet, Jena and AltovaXML.

III. The Integration System

High-level architecture

Figure 1 illustrates the high-level architecture of the integration system (tools) . The figure shows that the integration system has two tools (susb-systems): query Tool and integration tool. The integration tool reads the schema of each data store and the ontology-2 information and builds data warehouse schema (structure) for those data stores. The query tool receives a structural natural language query and analyzes it based on the ontology-1 information, and builds a SQL query to retrieve the required data from the datawarehouse.

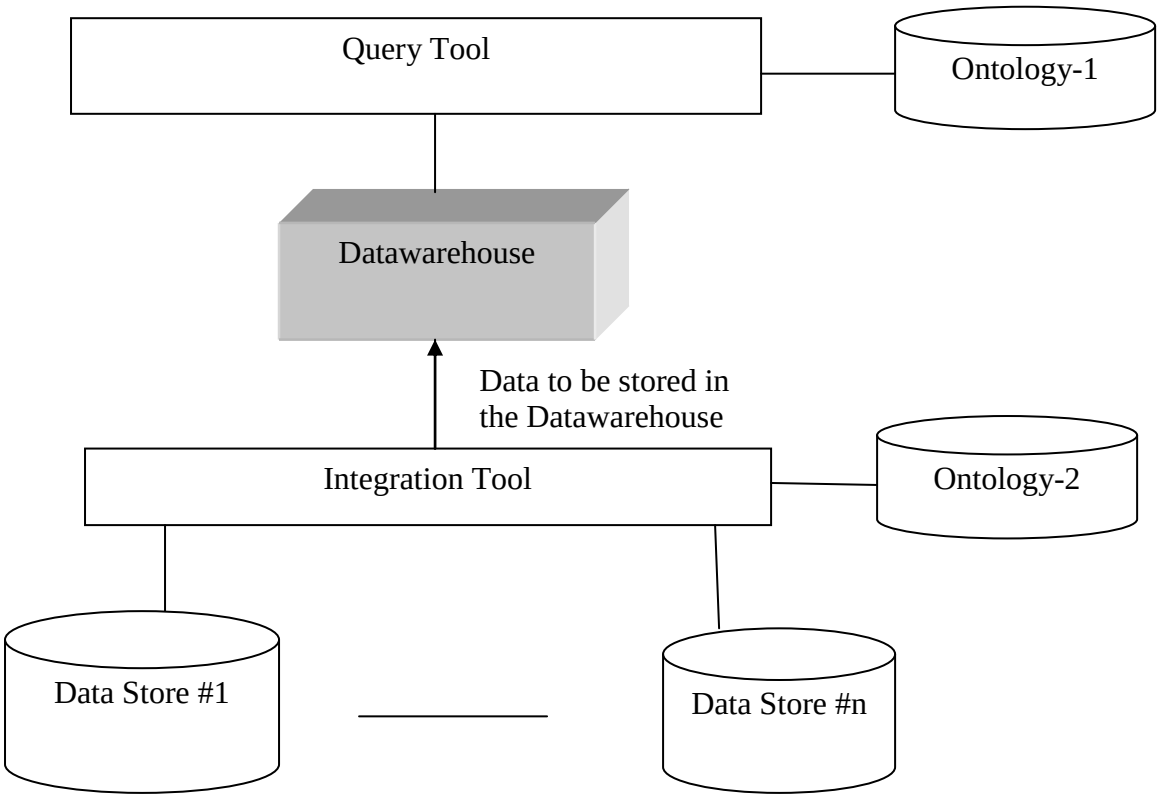


Fig. 1: High level architecture of the system

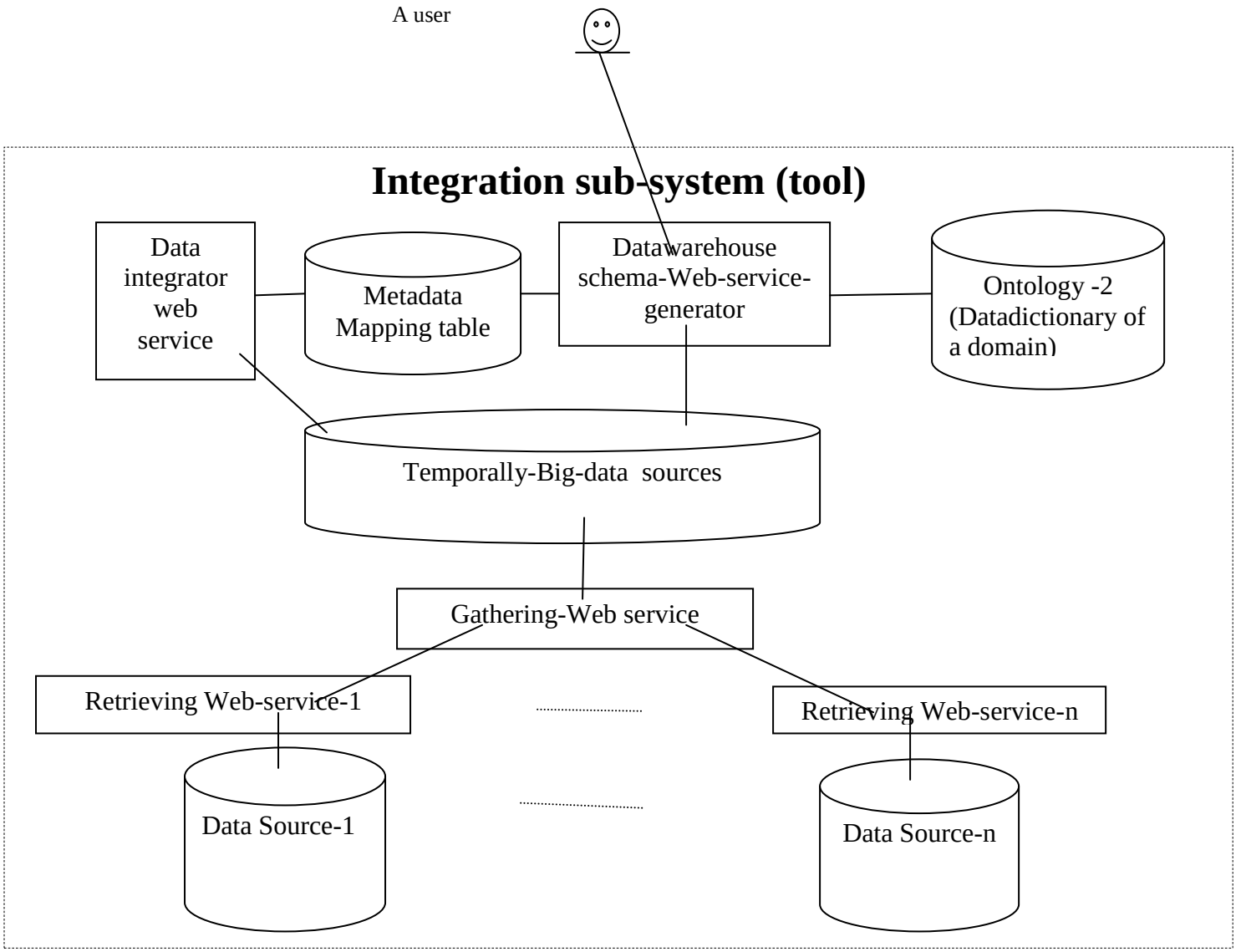


Figure 2: The integrator sub-system

The Architecture of the Integration System

The system consists of two sub-systems: integration sub-system, and query sub-system as shown in figure 2. The integration-subsystem has a set of web services: retrieving web services, Gathering web service, datawarehouse schema-web-service-generator, and data integrator web service. In addition the integration sub-system contains ontology-2 that includes domain data dictionary.

In the above architecture, the retrieving web-service-1 until retrieving web-service-n retrieve the updated or new data from the Data Source-1 unit Data Source-n and return the data to Gathering web service. Each retrieving web service checks off line its corresponding data source to retrieve the updated and new data.
The gathering web service receives the retrieved data from all resources and store them in Temporally-Big-data-sources. This Big-data-resources has all tables of all data sources but all them are stored in the same format of a database engine. This means that gathering web service convert the retrieved tables from different formats to the format of Temporally-Big-data-sources. The ontology-2 holds the data dictionary of the application domain. The data dictionary are stored and updated by the business analyst.

Datawarehouse schema-Web-service-generator reads data dictionary from ontology-2 and structure of all tables in Temporally-Big-data-sources and produce the structure of the data warehouse and the mapping metadata of the current application that are stored in the metadata mapping table. Data integrator web service integrates the data that exist in Temporally-Big-data-sources based on metadata mapping table and stores the integrated data in the datawarehouse.

The query sub-system contains a set of web services in addition to ontology-1. The web services are user interface web service, Query generator-web service, and Dataware-house web service. The user Interface Web service creates a user interface to the user where the user enters his query in formatted (has a syntax) natural language.

Query generator-web service receives the formatted query from the interface web services and based on the triples that are stored in the ontology-1 creates SQL-statement.

Data warehouse web service receives the created SQL-statement and retrieves the required data. The retrieved data is returned to the user interface web service to be displayed to the user.

The data warehouse is shared between the two sub-system and it is built automatically based on specific database engine such as SQL_server or DB2 or others.

Figure 3 shows a sequence diagram for the query subsystem. The diagram illustrates the dynamic behavior of the subsystem. In the sequence diagram, the user writes the query in natural language that accepted by the method write-snl-query() that has been implemented in user interface web service.

The Generate-SQL-s (snl) message is sent by the user interface web service and received by Query generator web service that it generates SQL (structure query language statement) based on ontology-1.

The generated SQL-s is sent with the Execute(SQL-s) message that is accepted by data warehouse web service. The data warehouse web service executes the statement by retrieving data from the data warehouse. The retrieved data is sent as actual argument of the message display-results(r-data) that is received by the user interface web service to be displayed. The sequence diagram of the integration subsystem is shown in figure 4a.

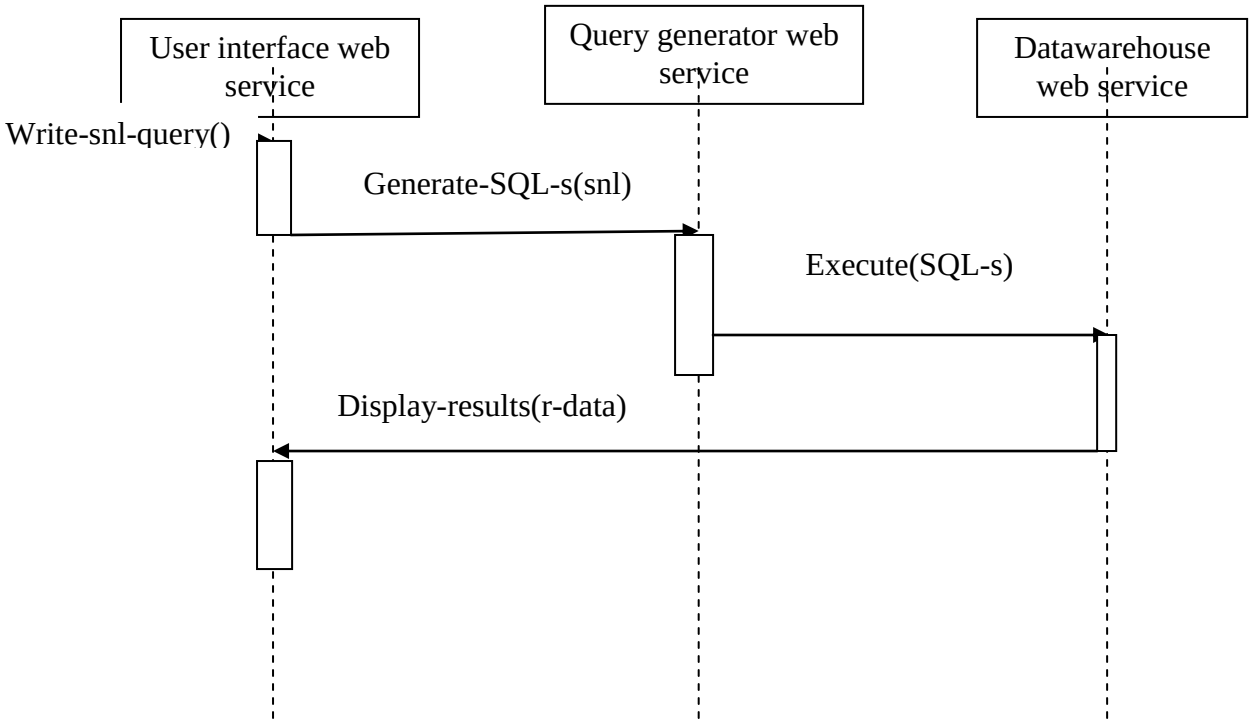


Fig. 3: Sequence diagram of query tool

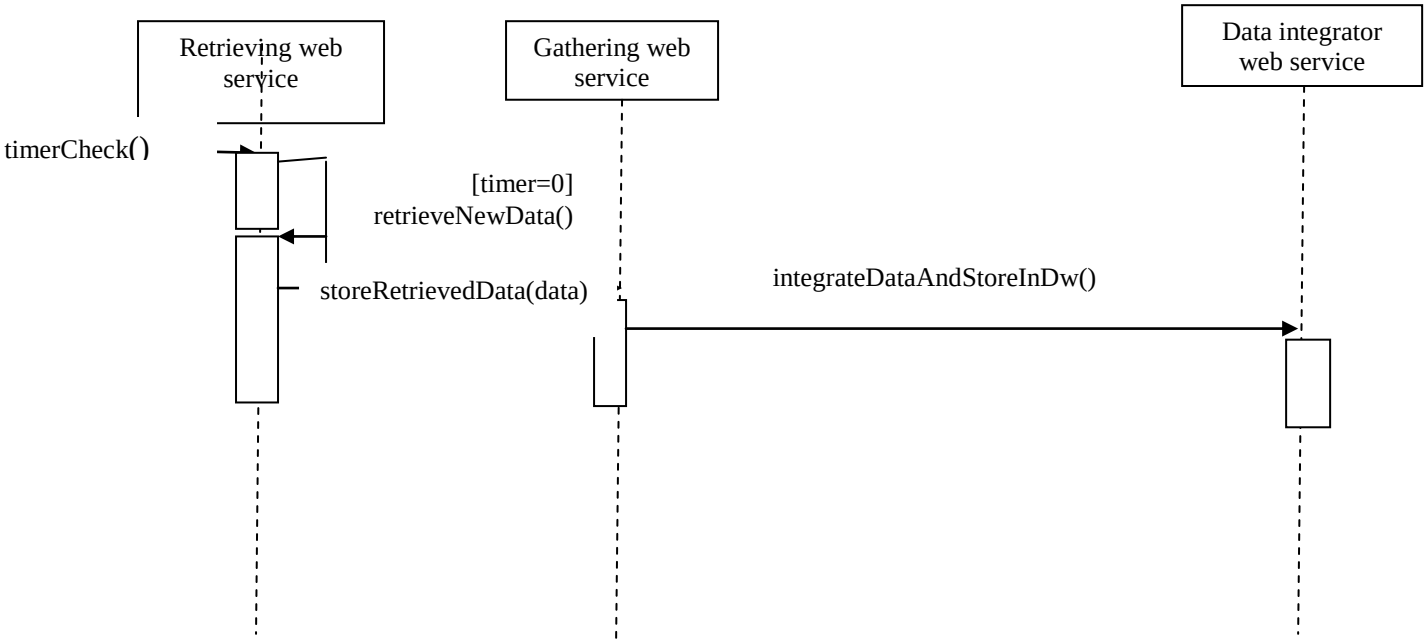


Fig. 4a: Sequence diagram of the integration tool

In sequence diagram(figure 4a), the retrieving web service checks a timer. If the timer value equals zero, then the " retrieveNewData()" method is called to retrieve all new data in data sources. The "storeRetrieveData(data)" method of gathering web service receives the retrieved data to store them in the temporally-big-data sources. The data integrator web service receives "integrateDataAndStoreInDw()" to integrate the gathered data based on metadata mapping table and store them in datawarehouse.

Figure 4b illustrates the sequence diagram of building the schema of datawarehouse. In the diagram the "Datawarehouse schema-Web-service- generator" receives the message "buildDwSchema()" message to build a new datawarehouse. The "Datawarehouse schema-Web-service- generator" sends the "collectSchemasOfDataStores()" message to "gathering web service" to retrieve all schemas of all data stores. Each "retrieving web service" receives the "retrieveDataStoreSchema()" message from the "gathering web service" to return the schema of its linked data store. Schemas of all data stores are returned to "Datawarehouse schema-Web-service- generator" to read the data from ontology2 and finally creates a new datawrehouse.

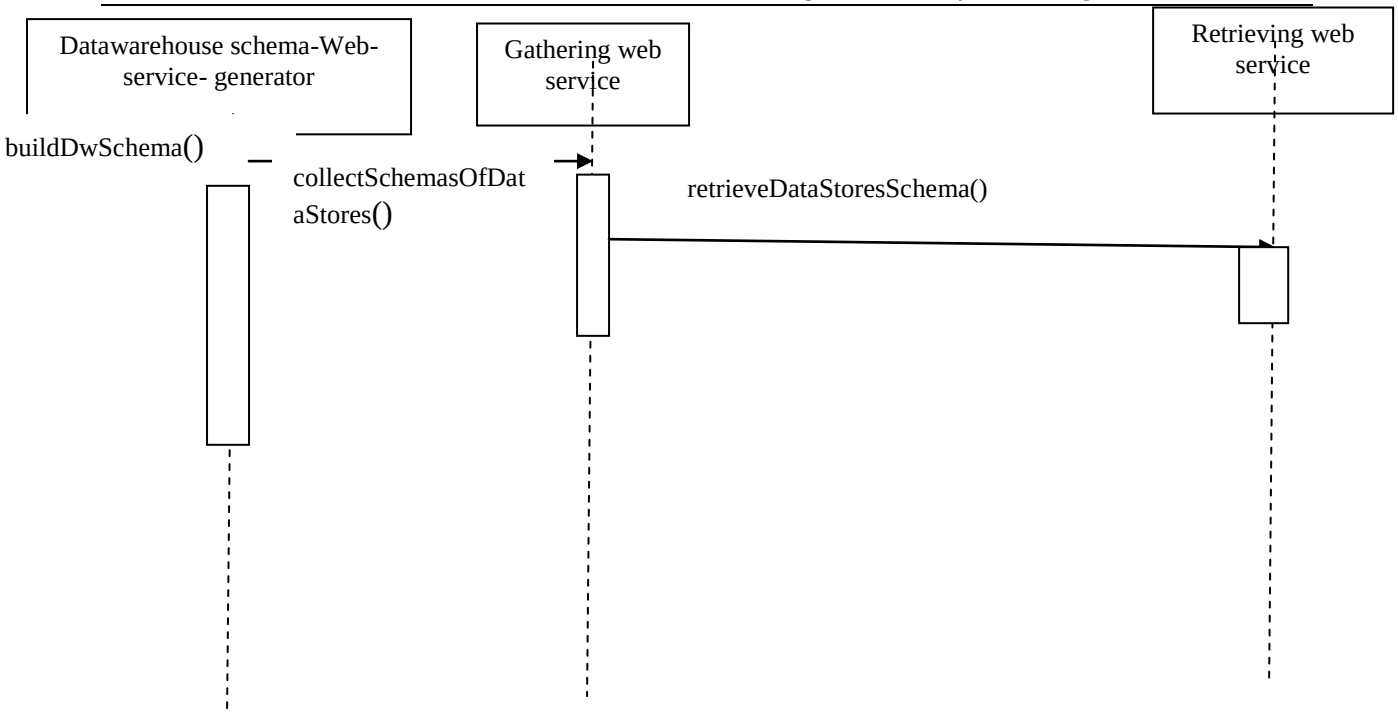


Fig. 4b: Sequence diagram of building datawarehouse schema

IV. Implementation

A prototype has been implemented using Java, and ASP.Net environment. Sun Microsystems provide Java Development Kits (JSDK) for many platforms, a standard edition and enterprise edition. The used data sources in this prototype are XML, and RDF as they are dominant in data interchange.

The XML data source has the following schema structure Figure 5:

XML
Isbn
Title
Auther
Publisher
Date
Version

a. XML schema structure

Integration Prototype

isbn	title	auther	publisher	date	version	Source
0-471-33341-7	Probability and Statistics with Reliability, Queuing and Computer Science Applications	Kishor S. Trivedi	Wiley-Interscience Publication	2002	second edition	XML
0-672-31063-5	Database Developer's Guide With Visual Basic 6	Roger Jennings	Sams Puplishing	1999	0099432	XML
0-07-882231-9	JAVA: The Complete Reference	Patrick Naughton	Osborne Puplishing Company			XML
3-540-63411-8	Intelligent Software Agents	Walter Brenner	Springer Verlag	1998	9814462	XML
0-201-32582-9	Programming and Deplaying Java Mobile Agent With Aglet	Danny B. Lange	Addison-Wesley Puplishing Company	1998	9820525	XML
0-03-075156-x	General Chemistry With Qualitative Analysis	Kenneth Whitten	Saunders College Puplishing	1992	91050628	XML
0-201-11497-6	The C++ Answer Book	Tony L. Hansen	Addison-Wesley Puplishing Company	1990		XML
0-201-			Addison-Wesley			

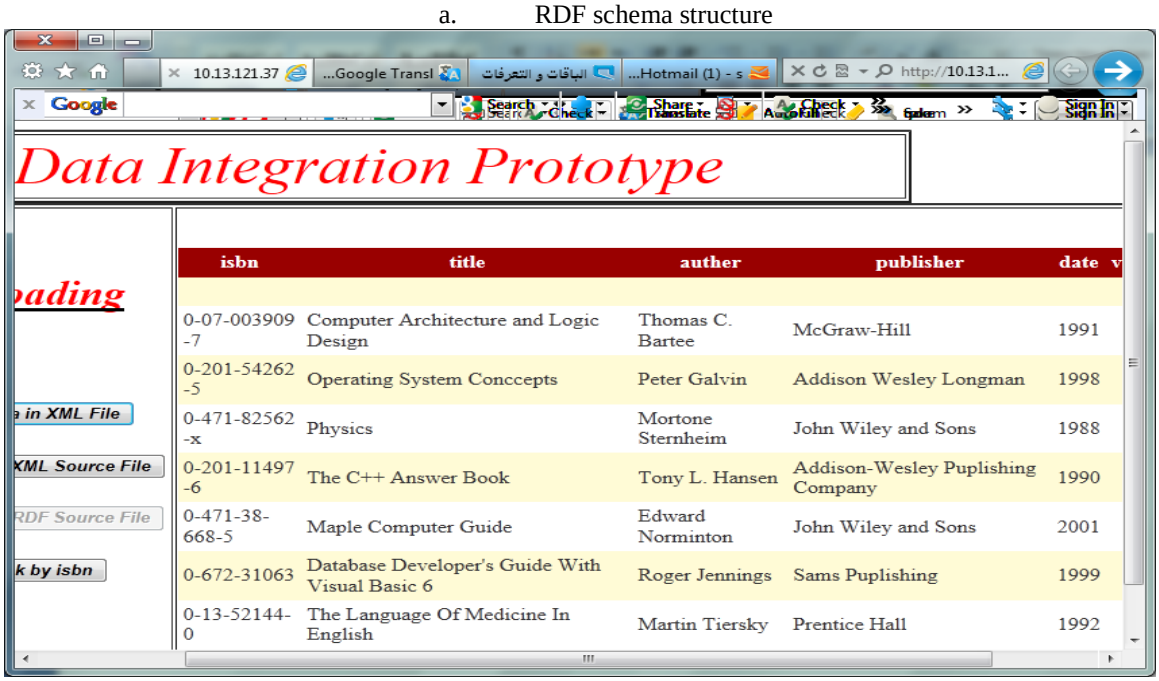
b- XML data sample

Fig 5. XML data sources

The RDF data source has the following schema structure Figure 6:

RDF
Creator
Title
Identifier
Publisher
Date

a. RDF schema structure



Data Integration Prototype

ading

in XML File

XML Source File

RDF Source File

k by isbn

isbn	title	auther	publisher	date
0-07-003909-7	Computer Architecture and Logic Design	Thomas C. Bartee	McGraw-Hill	1991
0-201-54262-5	Operating System Conccepts	Peter Galvin	Addison Wesley Longman	1998
0-471-82562-x	Physics	Mortone Sternheim	John Wiley and Sons	1988
0-201-11497-6	The C++ Answer Book	Tony L. Hansen	Addison-Wesley Puplishing Company	1990
0-471-38-668-5	Maple Computer Guide	Edward Norminton	John Wiley and Sons	2001
0-672-31063	Database Developer's Guide With Visual Basic 6	Roger Jennings	Sams Puplishing	1999
0-13-52144-0	The Language Of Medicine In English	Martin Tiersky	Prentice Hall	1992

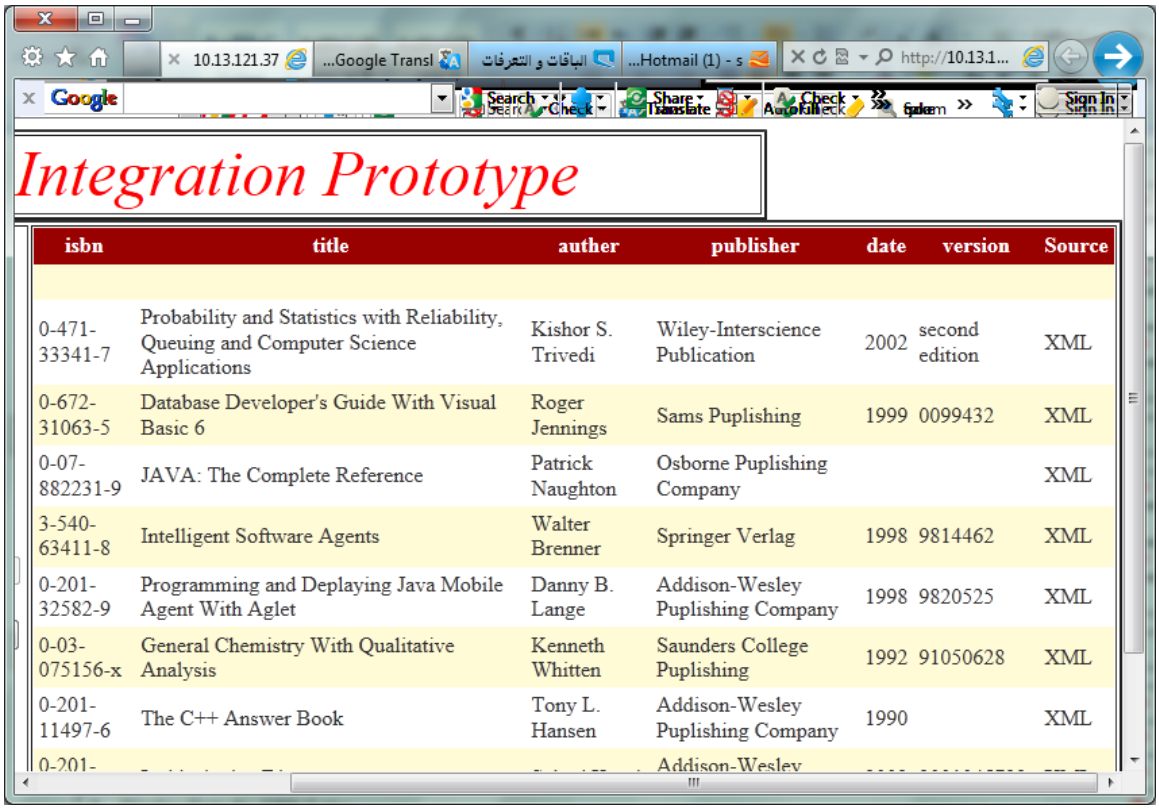
b- RDF data sample

Fig 6. RDF data sources

After that, a unified data source is generated from all the input different sources

Unified Data Source
isbn
title
auther
publisher
date
version
Source

a. The unified Schema structure



isbn	title	auther	publisher	date	version	Source
0-471-33341-7	Probability and Statistics with Reliability, Queuing and Computer Science Applications	Kishor S. Trivedi	Wiley-Interscience Publication	2002	second edition	XML
0-672-31063-5	Database Developer's Guide With Visual Basic 6	Roger Jennings	Sams Puplishing	1999	0099432	XML
0-07-882231-9	JAVA: The Complete Reference	Patrick Naughton	Osborne Puplishing Company			XML
3-540-63411-8	Intelligent Software Agents	Walter Brenner	Springer Verlag	1998	9814462	XML
0-201-32582-9	Programming and Deplaying Java Mobile Agent With Aglet	Danny B. Lange	Addison-Wesley Puplishing Company	1998	9820525	XML
0-03-075156-x	General Chemistry With Qualitative Analysis	Kenneth Whitten	Saunders College Puplishing	1992	91050628	XML
0-201-11497-6	The C++ Answer Book	Tony L. Hansen	Addison-Wesley Puplishing Company	1990		XML

b- Unified data sample

Fig 7. Unified data source

Then, we can request information from the unified data source based in certain information item. For example, we can query book information based on it ISBN as in Figure 8.

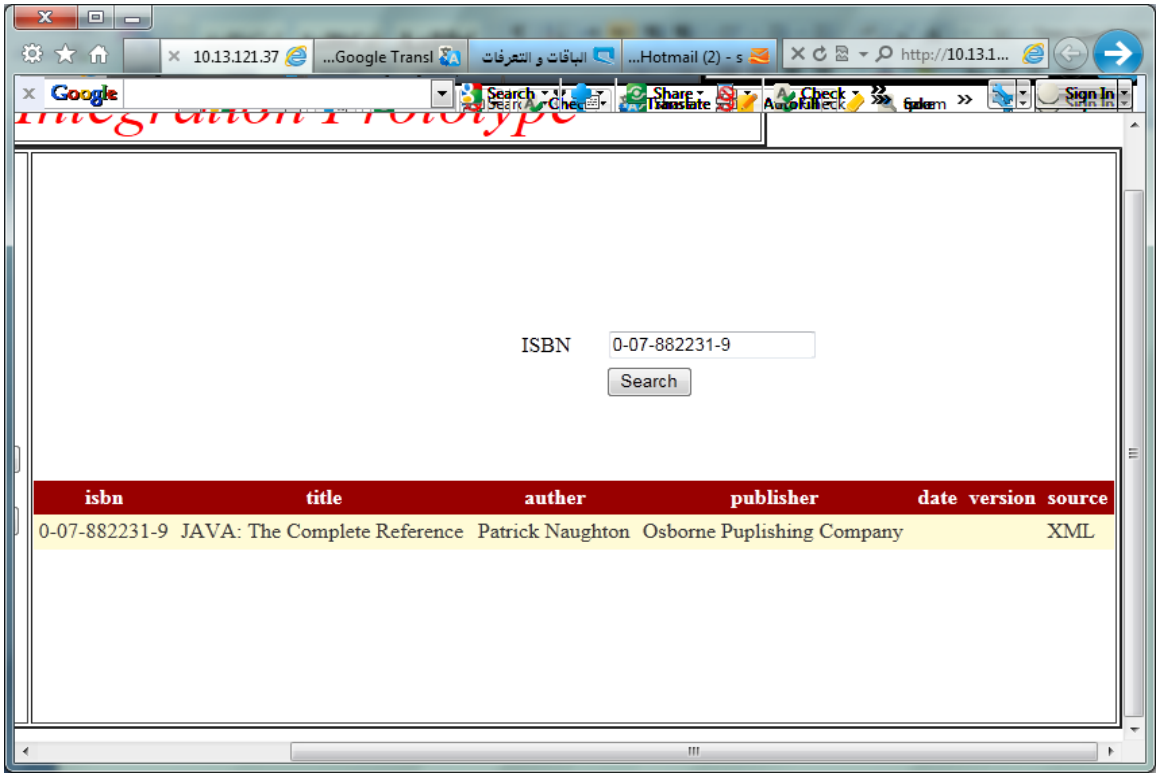


Fig. 8 : Querying the unified data source based on book ISBN

V. Conclusion

In this research we have built an integration system that collects data from different data sources that were generated by different database engines. Also, the system helps users to request data using structured natural language or structured query language. The system consists of two subsystems (tools): integration sub-system (tool) and query (sub-system) tool; the integration system has been built based on the web services technology.

The integration tool has been built as a multi web services for integrating data from different data stores (databases) that were created with different database engines; there is a retrieving web-service for each engine. The integration sub-system (tool) creates the schema of the data warehouse automatically based on domain ontology that is associated with the tool. This means that the time of building data warehouse is reduced and the performance of the application totally is increased.

The query sub-system (tool) has been built to help a user to query in a structured natural language or structured query language. The query sub-system uses local ontology that is associated with the system to understand the structured natural language query and concerted into SQL to retrieve the results from the data warehouse.

The system has been implemented, tested. The system has many advantages: a database engine independent, domain independent, and smart system because it is based on ontology scheme.

Our system also has good attributes: adaptable, reusable, and distribution. The system is adaptable because it can be used with any distributed system (platform or distributed system independent). It is reusable because its web services can be used in building similar tools without recompilation. The system is distributed means that its web services can be deployed on different machines in different locations. The system satisfy two non-functional requirements: scalability and performance. Scalability means that the system can serve any number of users without scarifying the performance. This is because the web services can be deployed on another machines to reduce the load on the existing machines.

References

[1] David W. Embley. “Toward Semantic Understanding—an Approach Based on Information Extraction Ontologies“. In proceedings of the Fifteenth Australasian Database Conference (ADC’04), USA 2004.

[2] Andreas Heß, Nicholas Kushmerick. “Learning to Attach Semantic Metadata to Web Services”. International Semantic Web Conference 2003.

[3] W.W. Cohen, P. Ravikumar, and S.E. Fienberg. A comparison of string distance metrics for name matching tasks. Proceedings of the IJCAI-2003 Workshop on Information Integration on the Web (IIWeb-03), 2003.

[4] S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: a versatile graph matching algorithm and its application to schema matching. Proceedings 18th International Conference on Data Engineering, pages 117–128, 2002.

[5] T. Pedersen, S. Patwardhan, and J. Michelizzi. Wordnet::Similarity - Measuring the relatedness of concepts. Proceedings of the National Conference on Artificial Intelligence, 19:1024–1025, 2004.

[6] A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Ontology matching: A machine learning approach, pages 385–516. Springer Verlag, Berlin, Heiderlberg, New York, 2003.

[7] Brend Amann, Catriel Beeri, Irini Fundulaki, Michel Scholl. " Querying XML Sources Using an Ontology-Based Mediator". In On the Move to Meaningful Internet Systems, Confederated International Conference DOA, CoopIS and ODBASE, pages 429-448, Springer-Verlag, 2002.

[8] Isabel Cruz, Huiyong Xiao, Feihong Hsu. "An Ontology-Based Framework for XML Semantic Integration". In 8th International Database Engineering and Applications Symposium (IDEAS 2004).

[9] Stephan Philipi, Jacob Kohler. "Using XML Technology for the Ontology-Based Semantic Integration of Life Science Databases". IEEE Transactions on Information Technology in Biomedicine, vol. 8 no. 2. June 2004.

[10] Jane Hunter, Carl Lagoze. "Combining RDF and XML Schemas to Enhance Interoperability Between Metadata Application Profiles". Copyright is held by the authors/owner(s). ACM. May 2001.

[11] Ngmnij Arch-int, Peraphon Sophatsathit, Yuefeng Li. "Ontology-Based Metadata Dicctionary for Integration Heterogeneous Information Sources on the WWW". Australian Computer Society Inc.. 2003.

[12] Baoshi Yan, Robert MacGregor. "Translating Naive User Queries on the Semantic Web". Proceedings in Semantic Integration Workshop, ISWC 2003.

[13] Olivier Corby, Rose Dieng-Kuntz, Fabien Gandon. "Approximate Query Processing Based on Ontologies". IEEE Intelligent Systems, IEEE 2006.

[14] Mostafa Saleh." Semantic Query in Heterogeneous Web Data Sources". International Journal of Computers and their Applications,USA, March 2008.

[15] E. Mena, V. Kashyap, A. Sheth, A. Illarramendi. " OBSERVER: An Approach for Query Processing in Global Information Systems Based on Interoperation Across Pre-existing Ontologies". In International Journal on Distributed and Parallel Databases (DAPD), ISSN 0926-8782, v.8 n.2, April 2000.

[16] Yingge A. Wang, Elhadi Shakshuki. "An Agent-based Semantic Web Department Content Management System". ITHET 6th Annual International Conference. 2005 IEEE.

[17 K. Munir, M. Odeh, R. McClatchey, S. Khan, I. Habib. "Semantic Information Retrieval from Distributed Heterogeneous Data Sources". CCS Research Centre, University of West of England, 2007.

[18] N.Al-Ghamdi, M. Saleh, and F. Eassa, "Ontology-Based Query in Heterogeneous & Distributed Data Sources", International Journal of Electrical & Computer Sciences IJECS-IJENS Vol: 10 No: 06, 2010