A New Architecture for Dynamic E-Business Database Interoperability

Omar Khadeer Hussain and Ben Soh
Department Of Computer Science and Computer Engineering
La Trobe University
Bundoora, VIC, Australia 3083

Abstract

There have been many approaches of data sharing in e-businesses, attempting at mapping each field at the source to its corresponding field at the target. These approaches eliminate the need for deriving the global schema and hence eliminate a substantial overhead. But, the approaches work out well only when both the source and the target databases have approximately the same content. Moreover, if some concepts of source do not have the counterpart fields in the target then the mapping will lose the concepts. This problem will be further aggregated by the mapping compositions and the large size of the databases. In this paper, we propose a framework to alleviate the interoperability problem.

1. Introduction

The development of XML has generated unrealistic claims and expectations in relation to its use in dynamic e-businesses. These claims have misled people to think XML as an application which can achieve interoperability easily [1], whereas XML improves data integration with the help of other applications and not by itself alone [2] . In reality Data Integration and Interoperability will continue to be a challenging issue but XML with the help of other applications will help to achieve it in an efficient way for e-business databases. A lot of discussions on data integration have been done [3, 4, 5].

For achieving data integration it is essential that the data sources must be mapped efficiently. There are two general approaches for mapping the data sources, source-to-global and source-to-source. In source-to-global mapping, the local fields from each source are mapped to a set of globally defined fields and data exchange is achieved with those fields. Source-to-global or "global as view" mapping is tedious, considering the definition of global fields first and then mapping them with the local fields from each source. One such technique has been discussed in [6] and it proposes algorithms for efficient query re-writing and optimization.

In source-to-source mapping, each field from the source is mapped to its corresponding field at the target, thus avoiding the need for a set of globally defined fields, hence eliminating a substantial overhead. But mapping each field from the source to the target too is tedious when there are a large number of pairs to be mapped. One of such approaches has been followed by [7]. It proposes an architecture which does not require the mappings to be defined between each pair of sources. Rather, schema mappings are being composed to derive new mappings. Hence it provides the option for the source to map to another source. But this approach works out well when both the source and the target databases have approximately the same content and, if some concepts of source do not have the corresponding fields in the target then the mapping will lose the concepts. This problem will be further aggregated by the mapping compositions and the large size of the databases. In this paper, we propose a framework to alleviate the problem.

2. Problem Definition

As mentioned in Section 1, the development of XML has created a lot of expectations towards database interoperability. But some of the problems that XML have to overcome are:

- Eliminating the multiple standards.
- To confirm and to be consistent with the semantics
- To adapt to each data source terminology and this problem is quite common in multi-databases.

There are some techniques by which interoperability between databases can be achieved. These techniques handle XML by following either source-to-global or source-to-source mappings. These approaches are good

Proceedings of the IEEE International Conference on E-Commerce Technology for Dynamic E-Business (CEC-East'04)
0-7695-2206-8/04 $ 20.00 IEEE

only when the databases are small but become quite tedious and cumbersome when the size of the databases grows and with it the number of fields to be mapped.

What we need is an approach that will help in achieving interoperability without using either source-to-global or source-to-source mapping. To this end, we propose an architecture that is application specific and do not use either of these techniques.

3. Proposed Solution

Our aim is to define a simple but an efficient way of achieving interoperability. For achieving interoperability we need to have a set of common fields through which we share the information. Our proposed approach is based on defining declarations over the data sources which identify the fields that can be mapped from each data source in the form of predicates. These declarations are then specified to an application specific vocabulary and data sharing can then be achieved through these fields. Defining the fields to the vocabulary is possible through Semantic Web and its applications, which provide a common framework, allowing data to be shared across applications. As mentioned in [8] "The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation". Semantic Web promises to expose the information content of the web resources which apart from text also includes audio and image.

The proposed architecture is defined and explained with an example and how the semantic web and its technologies are used in making this architecture is discussed in the next subsections.

3.1 Defining the Architecture

In our architecture we assume the availability of standard vocabulary predicate library, which consist of the common fields in the form of predicates related to an application. From each data source in the architecture, application specific semantic declarations are defined to the application specific vocabulary. These declarations consist of the predicates which will be used in data sharing and they expose the semantic concepts present in each database to the application specific vocabulary. Internal mappings are defined between each field of the predicate.

The aim is to achieve interoperability easily by allowing the user to write the queries by referring to these predicates. This takes the burden from the user to create the mappings between each field in the sources to be used for data sharing. The queries that the user writes by looking at the vocabulary gets executed according to the internal mappings defined between them and the result is given back. Hence neither source-to-source or source-to-global mapping is utilized for data sharing.

The architecture is shown in figure 1. It shows two data source across different application and the standard vocabulary defined over each data source and how the user queries it.

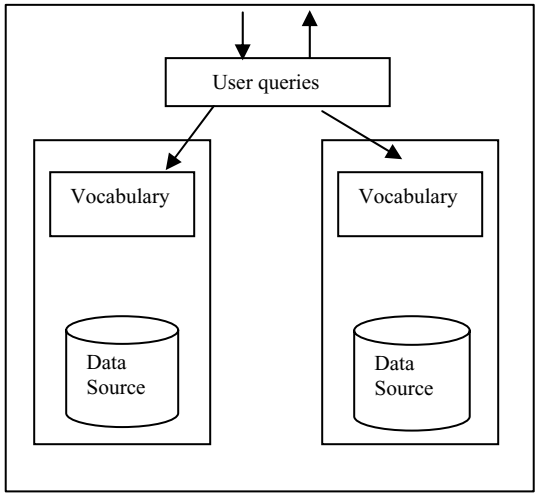

Figure 1

3.2 Role of Semantic Web and its applications

The semantic web is an efficient way of representing information relating to dynamic e-businesses. It is like a mesh, which contains information that can easily be accessed from any where [9]. It provides a framework that is used for sharing data across the applications. Usually the semantic web is built on syntaxes that use URIs to represent data, in triples based structure. Many triples of URI data that can be held in databases, or interchanged on the Web use a set of particular syntaxes. These syntaxes are called "Resource Description Framework". It is based on the Resource Description Framework, which integrates a variety of applications using XML for syntax and URIs for naming [10].

The semantic web comes with its own set of technologies such as Resource Description Framework (RDF), the ontology description

language. The technology that is of interest to us in developing this application is the Resource Description Framework (RDF). RDF maps the information directly and unambiguously to a model which is decentralized, and for which there are many generic parsers already available. An RDF application can know which bits of data are the semantics of the application. RDF is a mechanism for specifying metadata about a source that includes not only data but also the semantic concepts present within it. N-Triples are a line-based, plain text format for encoding an RDF graph. The RDF abstract syntax is a set of triples [11]. An RDF triple contains three components: subject, predicate and object. The abstract syntax is the syntax over which the formal semantics are defined. RDF normally employs predicates over subjects and values for describing such concepts.

3.3 An Example

To demonstrate the architecture, we will consider an example of e-business involving two major stores which sells some same products. It is possible that each store may have one or more than one branch in a state and the branches are spread all across the country. Each store has their own databases, which store the information of its branches and items available in each branch and it is stored according to its own structure of its database. Let the name of the two stores be *store 1* and *store 2,* whose structure is represented in figure 2 & 3 respectively. Suppose the supplier of the product wants to check the availability of the products in both the stores.

```
<! ELEMENT store (branch*)>
<! ELEMENT branch (state, city, item*)>
<! ELEMENT item (id, name)>
<! ATTLIST branchid ID #REQUIRED>
```

Figure 2

```
<! ELEMENT store (items, branches)
<! ELEMENT items (item*)>
<! ELEMENT item (id, name)>
<! ELEMENT branches (branch*)>
<! ELEMENT branch (state, city)>
<ATTLIST item branchid IDREF
 #REQUIRED
<ATTLIST branchid ID #REQUIRED
```

Figure 3

A query will be formulated of this kind "For each state list the state information and the distinct items available in the stores of that state". As both the stores sell the same product so the item id will be the same for those products. This can be done by writing the query on each database using XQuery. However, there are two disadvantages in that. They are:

- Each field in the data source must be identified and queries have to be written for each data source in question.
- The data obtained from each source must be joined for getting the final data.

If we employ the techniques mentioned in [7, 12] we can achieve better solutions as compared to XQuery but those techniques follow either source-to-source or source-to-global mapping. But with our technique we can avoid using either of those approaches. Steps for how to achieve that are discussed in the next sub sections.

3.4 Predicates

Before proceeding further with the above example, we briefly discuss what predicates are in our proposed architecture. The standard vocabulary predicate libraries consist of the common fields through which data sharing is achieved. An application specific standard predicate library has a number of predicates that are needed for modeling of the application. A standard vocabulary predicate library lists:

- Predicate names and usages
- The type and role of each argument
- Information about identifiers.

In order to achieve maximum flexibility the predicates defined should be as simple as possible. Each predicate defined provides a relationship between two arguments and it should not be divisible further into smaller predicates. Considering the predicates needed for achieving the result of the above query are:

- Item-name
- Item-branch
- Branch-state
- Branch-city

Each predicate consists of two arguments, where the arguments define the field or the information, which it is representing. This is according to the RDF specification where a URI is associated with each subject, predicate and object. E.g. Item-name takes two arguments where the first argument should be an identifier of item and the second argument should be an identifier of name.

3.5 Creating the Mappings

Once the predicates are known, mappings from the predicates to the data fields, which are stored in the database, should be defined. These mappings can be defined by using the mapping languages or transformation rules [13] or XPath. In [13] a framework is shown for refining the relational design of XML storage based on XML key propagation. It develops algorithms for checking whether a functional dependency is propagated from XML keys, and for finding a minimum cover for all functional dependencies propagated from XML keys. The mappings between the binary predicates and the data source are defined in the following form:

P($A,$B)<-path1 $G, $G/path2 $A, $G/path3 $B

where $A, $B are the arguments of the predicates and $G is the join or the glue variable which is used for joining. Once the mapping is done, the user writes the queries over the predicates and they get executed according to the mappings defined and the result is given back.

4. Conclusion

Resource Description Framework is a technology of the semantic web which semantically marks the data source to the application specific vocabulary. By the presence of the standard predicate libraries and by defining the semantic declarations and the internal mappings between the predicates and data fields we are allowing the user to formulate the query by looking at the common vocabulary thus eliminating source-to-source or source-to-global mappings. User queries are spread over these vocabularies and they are executed according to its internal mappings.

But creating RDF semantics for large databases can be quite tedious. To over come this we can develop tools or write in a transformation language like XSLT which takes the XML data and transforms it into the required format. Future work includes finding an efficient way of formulating a query and taking into the account the query optimization.

5. References

[1] Stuart E. Madnick, "The misguided Silver Bullet: What XML will and will NOT do to help Information Integration" *Modulant Solutions White Paper July 2001.*

[2] Jon. Bosak, "Media-independent Publishing: Four myths about XML", *IEEE Computer,* Vol. 31, No. 10, October 1998, pp. 120-122.

[3] Maurizio. Lenzerini. "Data integration: A theoretical perspective" *ACM* PODS Wisconsin, June 3-6 2002.

[4] Philip. A. Bernstein. "Applying model management to Classical Metadata problems." *CIDR*, Asilomar, January 5-8 2003.

[5] Erhard. Rahm and Philip. A. Bernstein. "A survey of approaches to automatic schema matching". *The VLDB Journal 10,* 2001 ,pp. 334-350.

[6] Bernd Amann, Catriel Beeri, Irini Fundulaki and Michel Scholl "Querying XML sources using an ontology-based mediator", *CoopIS* 2002, pp 429-448.

[7] Alon Halevy, Oren Etzioni, AnHai Doan, Zachary Ivesy, Jayant Madhavan, Luke McDowell, and Igor Tatarinov " Crossing the Structure Chasm" *CIDR 2003.*

[8] Tim Berners-Lee, James Hendler and Ora Lassila "The Semantic Web" http://www.scientificamerican.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21&catID=2.

[9] Sean B. Palmer, "The Semantic Web: An Introduction" http://infomesh.net/2001/swintro/.

[10] Semantic Web: http://www.w3.org/2001/sw/.

[11] Resource Description Framework: Concepts and Abstract Syntax http://www.w3.org/TR/rdf-concepts/.

[12] Lucian Popa, Yannis Velegrakis, Renee J. Miller, Mauricio A. Hernandez, Ronald Fagin "Translating Web Data" *VLDB,* Hong Kong*,* 2002, pp.598-609.

[13] Susan Davidson, Wenfei Fany, Carmem Hara, Jing Qin, "Propagating XML constraints to Relations" *IEEE International Conference on Data Engineering*, Bangalore, March 5-8 2003.