

XML based Adaptation of the Composite Approach for Database Integration

Brian Ensink, Kimberly Haveman, Mochan Shrestha, Todd Schavey

Department of Computer Science and Information Systems

Grand Valley State University

Allendale, MI 49401

ensinkb@river.it.gvsu.edu, havemank@river.it.gvsu.edu, shresthm@river.it.gvsu.edu,

schaveyt@river.it.gvsu.edu

Abstract: In this paper, we introduce a new approach of database integration, through the use of the Extensible Markup Language (XML). We also demonstrate the applicability, through an experiment with an XML subset we call BSML (Bookstore Markup Language) that integrates heterogeneous bookstore databases. BSML addresses a valid problem associated with network-accessible databases, and demonstrates an XML-based solution to the problem. This approach is quick, affordable, and convenient that does not require a large investment of capital and resources when integrating databases.

1.0 Introduction

The way information is gathered and stored is changing rapidly. Placing databases on the Internet for easier public access is a current trend in industry. Unfortunately using multiple web-enabled databases is often strenuous on a user. It is similar to having to logon to a database multiple times to perform a query. Allowing for a search of integrated multiple web-databases at once would, in many instances, greatly ease the search for the information. In this paper, we present an adaptation of Extensible Markup Language (XML) as a new approach for integrating databases, which can be conveniently applied on web-accessible databases.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

'99 ACM Southeast Regional Conference

©1999 ACM 1-58113-128-3/99/0004 5.00

In Section 2 we give an overview of different database integration methods. In Section 3, we describe the current issues in user interaction with a web-based database. Section 4 gives an overview of XML. Section 5 leads into a specific example problem and presents how XML can be used as a solution. Finally, in Section 6 we give a conclusion that recaps the issues and offers an insight into our future work.

2.0 Database Integration Approaches

Database integration plays a key role in the management of heterogeneous databases. There are several approaches to integrating individual databases. The system overhaul [9] approach requires the creation of a new system that consolidates the existing systems into one. In many cases this option is too costly and time-consuming (or just not a possibility at all). The federated approach [3] allows users to choose from a variety of individual database schemas, but it (sometimes unrealistically) expects users to learn the differences among the individual databases. Next, the application cover-up approach [9], provides a short-term solution to data integration. Queries for each application are written independently and are custom to that schema. This approach, although usually the fastest, is often inflexible and unrealistic for a multitude of applications. The composite approach [9,1] is based on creating "a virtual data warehouse" which provides the feel of a single repository, while allowing the data to "remain in its natural distributed setting". The mediated approach [6] is a variation of the composite approach, which enables users to remain within an existing form of database interaction with which they are already familiar. The data warehousing [7] approach creates a repository of physical information

collected from separate databases to provide a tool for trend observation and management decision support. Many different approaches to data integration can be customized to a corporation's requirements.

3.0 Client/Server Database Interaction

Much activity in industry involves client/server interaction with a database, which facilitates separation of the database server and any entity or client interacting with it. The client may interact with one or many databases, and the database server may service many different clients. A very common example of this technique is using a web-based client to access a web-enabled database server. This technique utilizes a database's management and data retrieval features as well as the expansive infrastructure of the Internet. Creating web-enabled databases merges these concepts into "cutting-edge technology that will revolutionize how information is served to the public" [8].

In this paper we observe an example of this cutting-edge technology. A bookstore's inventory is stored in one or more databases. Through a web-based client, a user can query these databases to retrieve information on a desired product. We examine a situation when a user wants to check a product's price and availability at more than one bookstore. First, she must locate and go to a number of online bookstores, each with different resources and searching functionality. Second, a search is required to sort through inventory. Finally, after repeating these steps for each bookstore, the user makes a decision on where to purchase the book. This multi-step process is very inefficient and cumbersome. Too much time and effort is required on the user's behalf.

This problem can be generally applied to a number of client/server architectures. Companies and their customers need a central querying system that combines data and various resources from related company databases. Data provided through this system must be consistent and current. A pre-determined integration of databases that are similar in nature would save time and effort on behalf of both the user and the system designers. The future of database integration falls in the hands of developers

looking to simplify database searching in a cost-effective and simple manner.

We would like to answer the call of the future, by suggesting the use of Extensible Markup Language (XML) to help solve the need for efficient and affordable database integration. Based on the composite approach to data integration, the XML-based system would enable integration of heterogeneous databases that are managed by different organizations or companies. The XML approach allows the data to remain at separate locations, each setup and formatted differently. It also allows for integration to take place, providing the look and feel of a single data repository. By adapting XML to the composite approach, we can get information out of multiple databases while retaining the searching precision of an individual database.

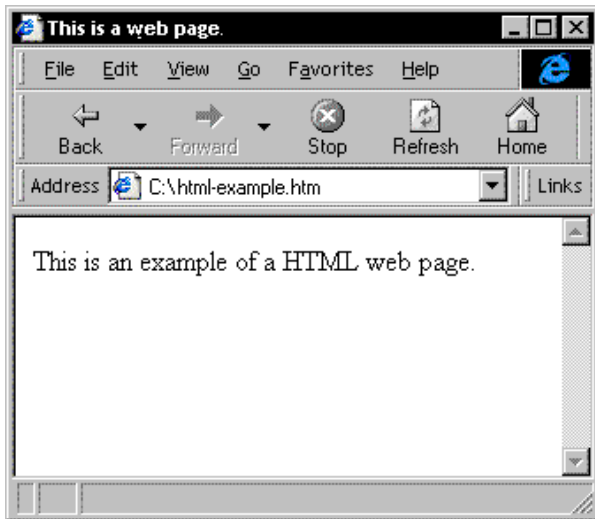
4.0 Extensible Markup Language

According to the World Wide Web Consortium (W3C), XML "is an extremely simple dialect of the Standardized General Markup Language (SGML)" [2]. SGML is an open, complex, and mature technology using tags to capture the *structure* data. Hyper-Text Markup Language (HTML), whose primary goal is the *presentation* of data, is a subset of SGML tags. HTML is used to define the formatting of a web page and virtually all web pages currently on the Internet are created using some version of HTML. Figure 1 shows an example of HTML code and its resulting web page:

Unfortunately, web developers not only started using HTML to format data but also to structure it. This misuse of formatting tags lead to difficulty in parsing, inefficient searching, and high development cost. HTML was never intended to be a data structuring language. Thus in 1996, the SGML Editorial Review Board of the W3C became the XML Working Group and started work on a lightweight version of SGML for the web. As defined by the XML Working Group, XML is "an application profile or restricted form of SGML"[2]. XML has been equipped to work with the data structuring aspects of SGML.

Figure 1

```
<HTML>
<HEAD>
<TITLE>This is a web page</TITLE>
</HEAD>
<BODY>
<P>This is an example of a HTML web page.
</BODY>
</HTML>
```



It is aiming to define what the data itself actually means. The following is an example of XML Code:

XML Code

```
<?xml version="1.0" ?>
<email>
  <to>smithj@domain</to>
  <date>June 1, 1998</date>
  <cc></cc>
  <subject>Going on holiday</subject>
  <body>John, I'm leaving for vacation.
    See you later. </body>
</email>
```

As with SGML, users are able to create their own tags specific to their application. In this example, new tags have been created to represent the structure of an email document. Notice that only the structure has been tagged with XML. The <to> tag signifies who the email is going to, the <date> tag represents the date the email was created, and so on. With this data now structured, any reader, machine or human, would know exactly what the section of the email is by searching for the appropriate tag. This is what

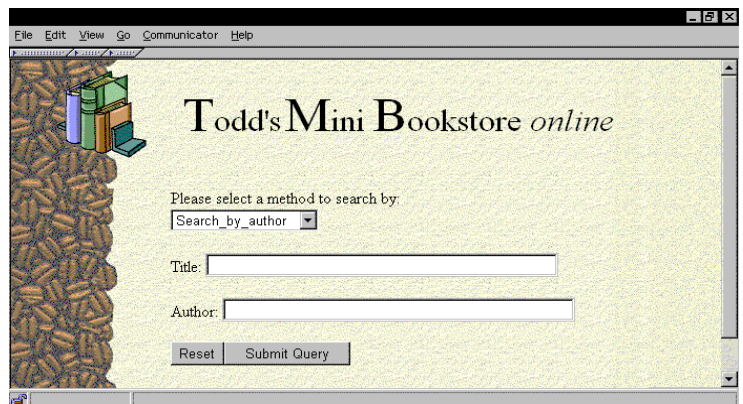
structuring the data means. Another application or language can be used to present the data to the specific requirement of the end users. For instance, the font for the <body> tag *could be* Arial for one application and Verdana in another. Removing the formatting information from the data makes the document less complex. This dramatically increases the performance of parsing and searching.

5.0 Combining Database Integration and XML

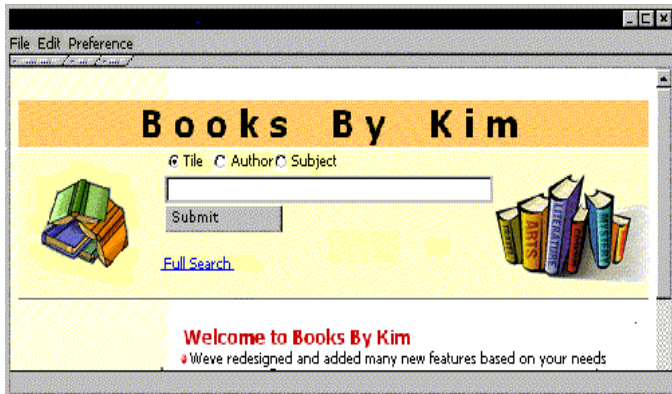
The connection between database integration and XML may not be clear at first, but through an example, it can be more easily examined. Given earlier was the scenario of a user searching multiple on-line bookstores for the best price of a book. Using this scenario, we would like to demonstrate that through the XML adaptation to the composite approach, a person's searching time could be significantly reduced. In this example, we have three, small, independently owned bookstores, each using a relational database to store inventory specific to company. Below are screenshots of each bookstore's search engines.

Note that both the interface and the internal structure are different for these three databases. Data types, the number of tables, the number of attributes differ from database to database. The internal structural details of these databases are given in [10].

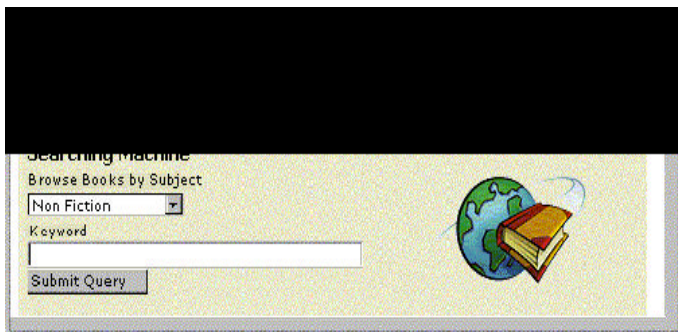
Bookstore 1



Bookstore 2



Bookstore 3



It is safe to assume that each of these small bookstores would prefer to find a quick, simple and cost-effective solution to integrating their databases, while keeping them independent. The XML adaptation to the composite approach of data integration is the solution the bookstores are searching for.

5.1 The XML Solution

The problem emerges because each bookstore has a unique system to store the information how they see fit. The differences may be simple (such as the design of the tables) or more complex (different platforms and/or application software). Our solution, which is similar to the composite approach, involves two steps. In our example, we deal with books. This same approach can apply to any entity: employees, automobiles, or stock quotes.

First we recognize the fact that while the databases are very different, each describes the same entity. These entities share a basic set of attributes. For example, all books have an ISBN, title, publisher and

one or more authors. Our XML solution is to create an object definition (in this case for a book) which contains these common attributes. The power of XML allows us to create these new tags. In essence we are creating a new markup language to describe an entity, in this case a book. Our BookStore Markup Language provides a set of tags to describe the attributes of a book. XML is the tool we used to define our new markup language. Below is an implementation of the BookStore Markup Language:

BookStore Markup Language (BSML)

```
<book>
  <isbn>...</isbn>
  <title>...</title>
  <authors>
    <a1>...</a1>
    <a2>...</a2>
    ...
  </authors>
  ...
</book>
```

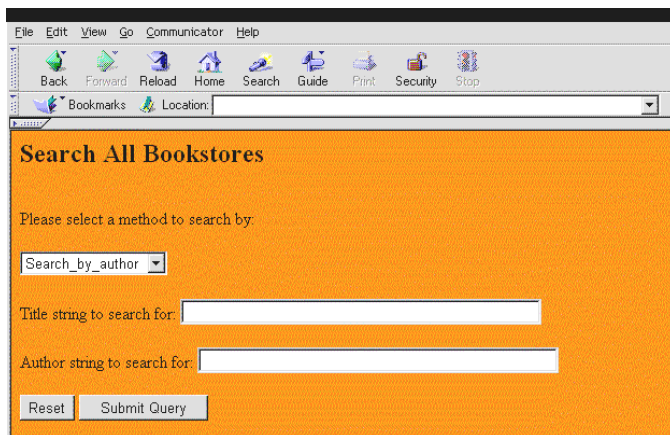
Obviously, being a definition, it does not yet contain any data. It is only a model or template by which the entity is to conform. The `<book>` tag signifies the beginning of a new record. Contained within this tag are other tags to describe the attributes of this book. Consider the title tags, `<title>` value `</title>`. The value contained between the tags represents the title of the book. Also notice how the authors are listed. First we begin a new section with the `<authors>` tag. Then the actual authors are listed within this tag. XML allows an object to contain other objects as we have here. Each of our 3 fictitious bookstores keeps track of this fundamental information on each book. By agreeing on a standard template to use, the bookstores can easily exchange data.

The final step in data integration through XML is the creation of custom programs to generate the XML tagged data from databases. This data conforms to the standard template described in the object definition (above).

In our scenario we created three independent bookstore databases, each with different database format. We then defined the BookStore Markup

Language (BSML). Our final step involved writing a specific program for each database to reproduce the data in BSML. These programs are given in its entirety in [10]. For our example we used C programming language with embedded Oracle 7.1 SQL statements (since our sample databases were implemented in Oracle 7.1). Certainly, this approach is applicable for the mix of different platforms (we were limited to Oracle only due to our resources). By combining the 3 separate BSML documents into a single larger document (by simple concatenation) we have successfully integrated these separate databases into a single unit. Figure 2 shows an example of the web-interface for the searching of this integrated database:

Figure 2



By no means is XML only limited to the web. Imagine the following scenario. A giant corporation with thousands of employees wishes to produce an annual directory listing of all employees. The problem is each department keeps track of employees in a different database system. Our solution would involve creating a standard object definition by which each department can conform. A comprehensive listing of all employees can then be produced, allowing us to create the directory of all employees.

6.0 Conclusion

In this paper, we introduced a new approach to database integration using the emerging technology of Extensible Markup Language. The new method is a

variation of the existing composite approach for integrating databases. The strategy we introduced allows for a *quick, affordable, and convenient* integration process. It gives heterogeneous environments a feasible alternative, especially if the large investment of capital and resources necessary for the other approaches is not acceptable. We illustrated this method by developing a subset of XML called the Bookstore Markup Language (BSML) and using it to integrate independent bookstore sample databases. In our future research we will examine the possibility of expanding our strategy to other database integration approaches, such as mediated approach.

References:

1. Y. Aren, C.Y. Chee, C Hsu, and C. Knoblock. "Retrieving and Integrating Data From Multiple Information Sources". International Journal on Intelligent and Cooperative Information Systems, Vol. 2, No. 2, 1993, pp. 127-158.
2. Tim Bray, Jean Paoli, and C.M. Sperberg-McQueen. "Extensible Markup Language (XML): Part I Syntax," World Wide Web Consortium Working Draft, August 1997. Available at <http://www.w3.org/TR/WD-xml-lang.html>
3. O.A Burkhres, J. Chen, W.Du, and A.K. Elmagarmid. "Interbase: An Execution Environment for Heterogeneous Software Systems". IEEE Computer, August 1993.
4. Dan Connolly, Rohit Khare, Adam Rifkin. "The Evolution of Web Documents: The Ascent of XML," World Wide Web Journal, XML: Principles, Tools, and Techniques, 2: 4, October 1997, pp. 119-127
5. Culshaw, Stuart, Michael Leventhal, and Murray Maloney. "XML and CSS," World Wide Web Journal, XML: Principles, Tools, and Techniques, 2: 4, October 1997, pp. 109-118
6. Brian Ensink, Kim S. Haveman, Todd Schavey, and Mochan Shrestha. XML code generator for web database integration, submitted to 13th National Conference on Undergraduate Research, Rochester, New York, April 1999
7. C. Goh, S. Madnick, and M. Siegal. "Context Interchange: Overcoming the Challenges of

Large-Scale Interoperable Database Systems in a Dynamic Environment". Proceeding of 3rd International Conference on Information and Knowledge Management. November 1994.

8. W. H. Inmon. "Building the Data Warehouse", Wiley Computer Publishing. 2nd Edition. 1996
9. Tania Neild. The Virtual Data Integrator- An Object-Oriented Mediator for Heterogeneous Database Integration, Ph.D. Dissertation, Department of ECE, Northwestern University, Evanston, Illinois.
10. Paoli, Jean, David Schach, Chris Lovett, Andrew Layman, Istvan Cseri. "Building XML Parsers for Microsoft's IE4," World Wide Web Journal, XML: Principles, Tools, and Techniques, 2: 4, October 1997, pp. 187-195.