An UML-XML-RDB Model Mapping Solution for Facilitating Information Standardization and Sharing in Construction Industry

I-Chen Wu[1] and Shang-Hsien Hsieh[2]

Department of Civil Engineering, National Taiwan University
Taipei 10617, Taiwan, R.O.C.

kwu@caece.net[1] and shhsieh@ce.ntu.edu.tw[2]

Abstract: To facilitate information standardization and sharing in Construction Industry, this paper presents a simple but effective approach that maps the UML (Unified Modeling Language) object-oriented information model related to a construction project to an XML schema, then to a Relational DataBase (RDB) schema. First of all, the mapping between UML model and XML schema is discussed since UML has been a popular tool to model the static structure and dynamic behaviors of the information and processes in a construction project, while XML has become a de-facto standard for information sharing and exchange. Then, a set of consistent rules for mapping from XML schema to RDB's Entity-Relational (E-R) model are studied and established since RDB has been the most popular choice for information management. The present study focuses on making the set of rules simple and easy-to-implement for most applications in construction industry. Finally, a mapping tool for automatically generating RDB schemas from XML Schemas is developed.

Keywords: UML, XML Schema, RDB, Information Standardization and Sharing

Introduction

During the life cycle of a construction project, voluminous data and information are usually created along the delivery processes of construction products. There is always a need to share and exchange these engineering data and information among related parties involved in the project. However, because data models defined in the information management systems of various parties are usually different, it is always difficult to directly map from one data model to another for the purpose of information sharing and exchange.

To address the aforementioned information-sharing problem among various parties, standardization of the data model for a construction project is usually inevitable. Due to the popularity of object-oriented modeling approach in recent years, an object-oriented information model is often constructed to represent the static structure and dynamic behaviors of the information and processes in a construction project and expressed by UML (Unified Modeling Language) [1], a popular tool for object-oriented modeling. Moreover, Extensible Markup Language (XML) [2] has become a de-facto standard for information sharing and exchange in recent years. Therefore, there is a need to define an XML schema based upon the UML object-oriented information model to further facilitate information sharing,

Furthermore, due to the popular use of Relational Database (RDB) technique for information management, it is also important to address the mapping between an XML schema and an RDB model. However, the mapping is not an easy one because the data model of an XML document is fundamentally different from that of a relational database. Especially the structure of an XML document is hierarchical and the XML elements may be nested and repeated.

Although several RDB providers have provided common data import/export tools to allow for data transformation between the XML documents and the RDB tables, the capabilities of these tools are still quite limited. Recently some commercial software has also provided tools for transforming information in XML documents into RDB. However, most of them can only transform simple XML documents. That is, if the XML documents have a nested data structure and association, most of these tools are still incapable of making a complete transformation.

In addition, several XML-to-Relational transformation algorithms and mapping tools have been proposed in the literature. For example, Bourret et al. [3] introduced an XML-RDB mapping language to specify transformation rules for generating an RDB schema from an existing XML DTD (Document Type Definition). They also proposed a lightweight, DBMS- and platform-independent load/extract utility to facilitate the data transfer between XML documents and relational databases. Lee and Chu [4] proposes a method where the hidden semantic constraints in DTD are systematically found and translated into relational data models. However, most of them deal with XML DTD, instead of XML Schema, which is more flexible and offers more supports for data types.

This paper presents an UML-XML-RDB model mapping solution that maps the UML object-oriented information model to an XML schema, then to an RDB schema. First of all, the mapping between UML model and XML schema is discussed. Then, a set of consistent rules for mapping from XML schema to RDB's Entity-Relational (E-R) model is presented. In this work, the set of rules is made simple and easy-to-implement for most applications in construction industry. Finally, a mapping tool developed for automatically generating RDB schemas from XML Schemas is discussed.

Constructing XML Schema from UML Data Model

This work adopts the concept of the XML Metadata Interchange specification (XMI), which defines a rigorous approach for generating an XML DTD from a metamodel definition, and slightly extends the approach of XMI for mapping object-oriented data model expressed by UML to XML Schema. The transformation rules employed in the mapping process are discussed as follows:

1. Mapping UML Classes to XML Elements

The UML Classes show the structural and behavioral features in the object-oriented Model. These features include attributes, association, aggregation, and composition. On the other hand, XML elements serve as a container for attribute and child elements. Thus, mapping UML classes to XML elements are quite straightforward.

2.Mapping UML Attributes to XML Attributes or Elements

Basically, either a primitive data type or an enumeration of UML attributes may be represented as an XML attribute. However, XML parser removes all extra whitespace characters, such as tabs, linefeeds, etc. That makes XML attributes mainly appropriate for simple datatypes of short string values. On the other hand, one can map attributes of an UML class to separate child elements of the corresponding XML element of the class.

3.Flagging UML Object Relationships by an XML Attribute

The current version (Version 1.0) of XML Schema does not yet have direct and full supports for expressing a complete object-oriented model, especially the distinction between the delegation and aggregation relationships that commonly exist in the model. Therefore, this work employs a special attribute named "relation" with a value of either "delegation" or "aggregation" to flag the relationship between the owning XML element and its child elements. It will be shown later in this paper how this special attribute is used to help mapping the XML object schema to a RDB table schema.

4.Constraints on Naming XML Elements

In general, the UML class name is directly used as the XML tag name in the mapping process. However, there are certain constraints we must comply when naming the XML elements:

- The tag name cannot have spaces in it, but symbols like ".", "-", and "_" are allowed.
- The tag name should not start with the string "XML".

Mapping XML Objects to Relational Database Tables

XML provides many of the things found in databases, such as storage (e.g., XML documents), schemas (e.g., DTDs, and XML schemas), query languages (e.g., XQuery, XPath, XQL), programming interfaces (e.g., SAX, DOM, JDOM), and more. However, XML lacks many important features that are supported by the real databases, such as efficient storage, indexes, security, transactions and data integrity, multi-user access, triggers, queries across multiple documents, etc. [5] Therefore, databases are still the top choice for managing information in a enterprise or government agency. Although several kinds of database technologies are currently available in the market (e.g., object-oriented database, object-relational database, etc.), RDB technology remains the most popular one employed in the construction industry.

On the other hand, there is a trend for using XML documents with a standardized schema as a medium for information sharing and exchange among different parties involved in a construction project. In order to transfer data from XML documents to an RDB for efficient management, it is necessary to map the XML document schema (XML Schema) to the RDB table schema.

Because the XML Schema can be viewed as a tree of objects that is mapped from the object model of its corresponding UML class diagrams (as already discussed in the previous section), this work employs the following object-relational mapping rules to map the XML schema to RDB tables [6]:

1.Mapping Elements to Tables

An XML Element object with attributes, element content, or mixed content (i.e., a complex Element object) is mapped to a separate RDB table as shown in Fig. 1.

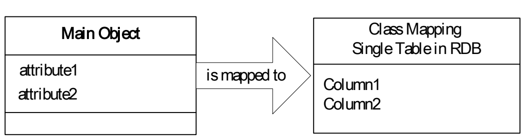

Fig. 1 Mapping an XML Element to an RDB table

2.Mapping Attributes to Columns

Since an XML Element is mapped to an RDB table, the attributes of the XML Element is naturally mapped to the columns of the corresponding RDB table automatically (as shown in Fig. 2).

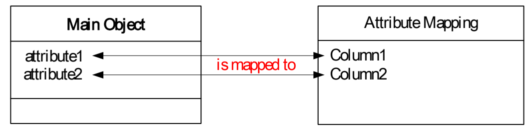

Fig. 2 Mapping attributes of an XML Element to the columns of its corresponding RDB table

3.Mapping XML Elements with Delegation or Aggregation Relationship to RDB tables

For mapping the delegation (or dependency) relationship between XML Element objects, the parent object and its child objects are mapped to separate RDB tables with a primary key and a foreign key, respectively, for associating them (as shown in Fig. 3).

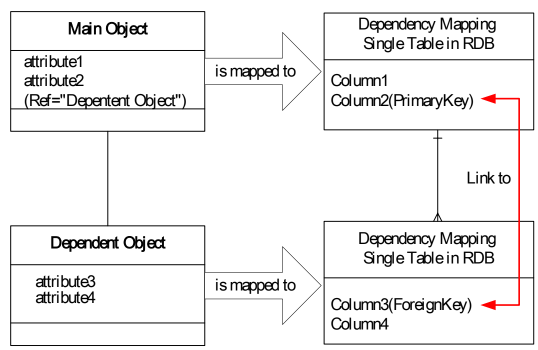

Fig. 3 Mapping XML Element objects with delegation (or dependency) relation-ship to RDB tables

For mapping the aggregation relationship between XML Element objects, the parent object and its child objects are mapped to a single RDB table uniting attributes of all the XML Element objects (as shown in Fig. 4).

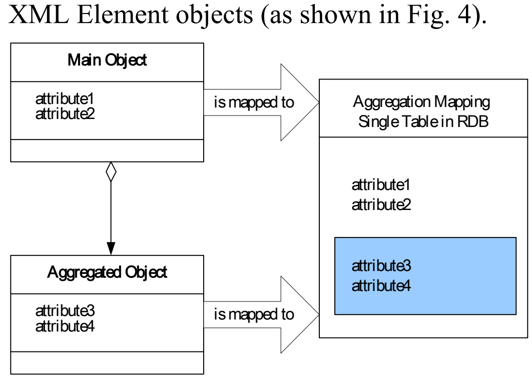

Fig. 4 Mapping XML Element objects with aggregation relationship to a RDB table

XML-RDB Mapping Tool

In this work, an XML-RDB mapping tool, called Schema2RDB, has been developed to verify the mapping rules discussed in the previous section and to automate and ease the task of creating RDB tables based on an existing XML schema. The tool is implemented using Java solutions mainly due to its platform-independent capabilities. JDOM API [7] is employed for an easy and efficient reading, manipulation, and writing of an XML document. JDOM itself is not a parser; instead, it is a wrapper. Therefore, it requires the presence of an underlying parser. In this work, Xerces is used for the parser. Because JDOM is used to handle all types of XML Data in this work, we can say that Schema2RDB is a 100% pure Java application.

Figure 5 shows the interactions among the user, Schema2RDB, and an RDB. First of all, the user assigns an XML Schema file and executes Schema2RDB. Then, Schema2RDB starts parsing the XML Schema file, applies the mapping rules discussed in the previous section, and generates SQL statements automatically via JDBC [8] to create relational database tables (in this work, Microsoft SQL Server). Finally, the RDB responds with the information about the created database tables and Schema2RDB forwards the information to the user.

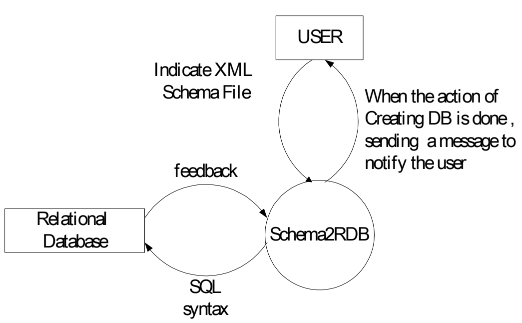

Fig. 5 Interactions among the user, Schema2RDB, and an RDB

The process of mapping an XML schema to RDB table schemas in Schema2RDB is discussed in more detail as follows:

1. As shown in Fig. 6, the structure of the XML schema is first parsed by a XML parser and converted into an XML DOM. The parser provides a way to extract XML schema information and can automatically resolve names of elements and attributes.

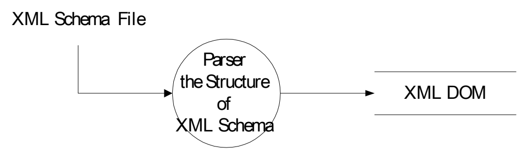

Fig. 6. Parsing the structure of XML schema

2. After the parser has resolved the names of the elements and attributes, Schema2RDB determines the names of the tables and columns as well as relationships for mapping to a relational schema.

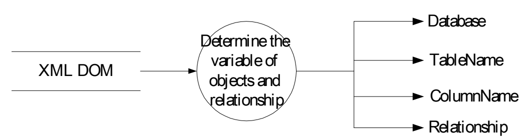

Fig. 7 Determination of the names of the tables and columns as well as relationships for RDB

3. Schema2RDB then composes the SQL statements required to create the RDB tables corresponding to the parsed XML schema (as shown in Fig. 8). The template of a SQL statement is first constructed according to the variable, then the value of the variable is used to complete the composition process.

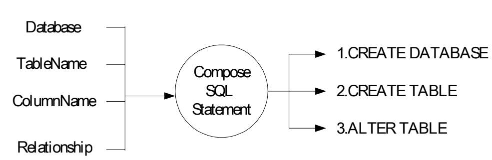

Fig. 8 Composing a SQL statement

4. Through the JDBC driver, Schema2RDB creates a connection to the relational database, executes the SQL statements, and finally closes the database connection. At this point, database tables are ready in

the RDB for accepting data from the corresponding XML documents.

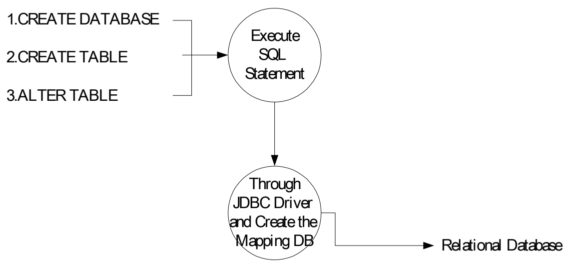

Fig. 9 Process for creating RDB tables

In addition, Schema2RDB have been successfully tested by several verification examples. Although these examples are not considered sophisticated ones, the effectiveness of Schema2RDB in mapping XML Schema to RDB table schema has been observed.

Conclusions

This paper has presented a simple solution for mapping UML object-oriented model to XML Schema, and then to RDB table schema. The framework and process of the mapping approach have been discussed herein. A set of simple, consistent, and easy-to-implement mapping rules has been proposed. In addition, a mapping tool, called Schema2RDB, has been developed to facilitate automatic generation of RDB table schemas from XML Schemas. It is hope that the work presented in this paper can help facilitate information standardization and sharing in construction industry

REFERENCES

1. M. Fowler and K. Scott, UML Distilled, 2nd Edition, Addison Wesley, Boston, 2000.

2. W. J. Pardi, XML in Action, Microsoft Press, Washington, 1999.

3. R. Bourret, C. Bornhövd, and A. Buchmann, "A Generic Load/Extract Utility for Data Transfer Between XML Documents and Relational Databases," Proceedings of the Second International Workshop on Advance Issues of E-Commerce and Web-based Information Systems (WECWIS 2000), San Jose, California, U.S.A., June 8-9, 2000.

4. D. Lee and W. W. Chu, "CPI: Constraints-Preserving Inlining Algorithm for Mapping XML DTD to Relational Schema," Data Knowledge Engineering, Vol. 39, No. 1, 2001, 3-25.

5. R. Bourret, "XML and Databases," from web page at http://www.rpbourret.com/xml/XMLAndDatabases.htm, February, 2002.

6. I. C. Wu, "Information Sharing and Exchange in Building Administration Using XML Technology," M.S. Thesis, Department of Civil Engineering, National Taiwan University, Taipei, Taiwan, 2002.

7. J. Britt, "Interactive Java & JDOM online tutorial," Commercially available from the TopXML website at http://www.topxml.com/, 2002.

8. G. Reese, Database Programming with JDBC and Java, 2/e, O'Reilly, U.S.A., 2001.