An XML Data Management System Using an Object-Relational Database

S. H. Nam[a], T. S. Jung[b], T. K. Kim[b], K. R. Kim[c], H. K. Zahng[c], J. S. Yoo[d] and W. S. Cho[c]

[a] Dept. of Bio Information Technology, Chungbuk National University
12, Gaesin-dong, Heungduk-gu, Chungju-si, Chungbuk,361-763, Korea
Tel: 043-276-3258, Fax: 043-266-8865, E-mail:daks_gada@chungbuk.ac.kr

[b]Dept. of Information Industrial Engineering, Chungbuk National University
12, Gaesin-dong, Heungduk-gu, Chungju-si, Chungbuk,361-763, Korea
Tel: 043-276-3258, Fax: 043-266-8865, E-mail :{mispro, tkkim}@chungbuk.ac.kr

[c]Dept. of Management Information Systems, 2nd BK team, Chungbuk National University
12, Gaesin-dong, Heungduk-gu, Chungju-si, Chungbuk, 361-763, Korea
Tel: 043-276-3258, Fax: 043-266-8865, E-mail:{k2ran, lodestone, wscho}@chungbuk.ac.kr

[d]Dept. of Computer and Communication Engineering, BIT team, Chungbuk National University
12, Gaesin-dong, Heungduk-gu, Chungju-si, Chungbuk, 361-763, Korea
Tel: 043-276-9961, Fax: 043-276-9961, E-mail:yjs@chungbuk.ac.kr

Abstract: *We propose an XML document storage system, called XDMS (XML Document Management System), by using an object-relational DBMS. XDMS generates object database schema from XML Schema and stores the XML documents in an object-relational database. SAX parser is used for understanding the structure of the XML documents, and XDMS transforms the documents into objects in the database. Experiment shows that object-relational databases provide more efficient storage and query model compared with relational databases.*
Keywords: *XML; XML Schema; object-relational DBMSs; SAX parser*

1. INTRODUCTION

Recently, XML has been used as a standard format for storing and exchanging data from various application domains such as e-business or bioinformatics areas[12]. As the amount of XML documents increases, efficient storage and query facility becomes an important issue [1, 2, 3, 4, 5, 6]. There may be three kinds of storage model for the XML documents: *relational databases, object-relational databases, and native XML databases.* Note that most of the conventional research uses relational databases or native XML databases as their storage systems [1, 3].

Relational databases have difficulty in the modeling of complex documents (tree structure) in the flat tables, and require complex queries with expensive join operations [1, 3]. Note that XML schema has been defined in an object data model [10]. **XML native databases** may be a good candidate for storing XML documents,

but it is still premature for large XML documents [2]. XML native databases lack in the well-known database functions such as high-level queries, optimal query processing, indexing, concurrency control and recovery, etc. **Object-relational databases** provide the benefit of the relational maturity and the richness of object-oriented modeling, including complex data types, multi-valued attributes, and object identifiers [4, 5]. Especially, XML schema (or queries) can be easily transformed into object-relational schema (or queries) because of their data model similarity [4].

We propose an *XML Data Management System (XDMS)* for XML documents with XML schema. In XDMS, all the components of the XML documents are stored and retrieved in an object-relational database. SAX parser is used for understanding the structure of the XML documents. Note that SAX parser requires small memory space compared with DOM parser, and thus XDMS is suitable for handling huge XML documents. The key features of XDMS are summarized as follows:

(1) XDMS automatically creates object-relational database schema from an XML schema. We devised a sophisticated algorithm for the transformation from XML schema into object-relational database schema.
(2) Rich data modeling features such as object reference, multi-valued attribute, and complex objects make the storage and retrieval activities easier than the relational databases
(3) SAX parser instead of DOM parser is used for memory efficiency; this is very useful for handling huge XML documents. In the bioinformatics applications, we should handle huge biological data in an XML format, which cannot be represented in the main memory if we use DOM parser.
(4) XDMS returns the query result in an XML format, which can directly be used in the other applications.

2. SYSTEM ARCHITECTURE

Fig. 1 shows the architecture of XDMS. XDMS has four components: Schema Generator, XML Document Insertion Module, Query Translator, and XML Document Generator.

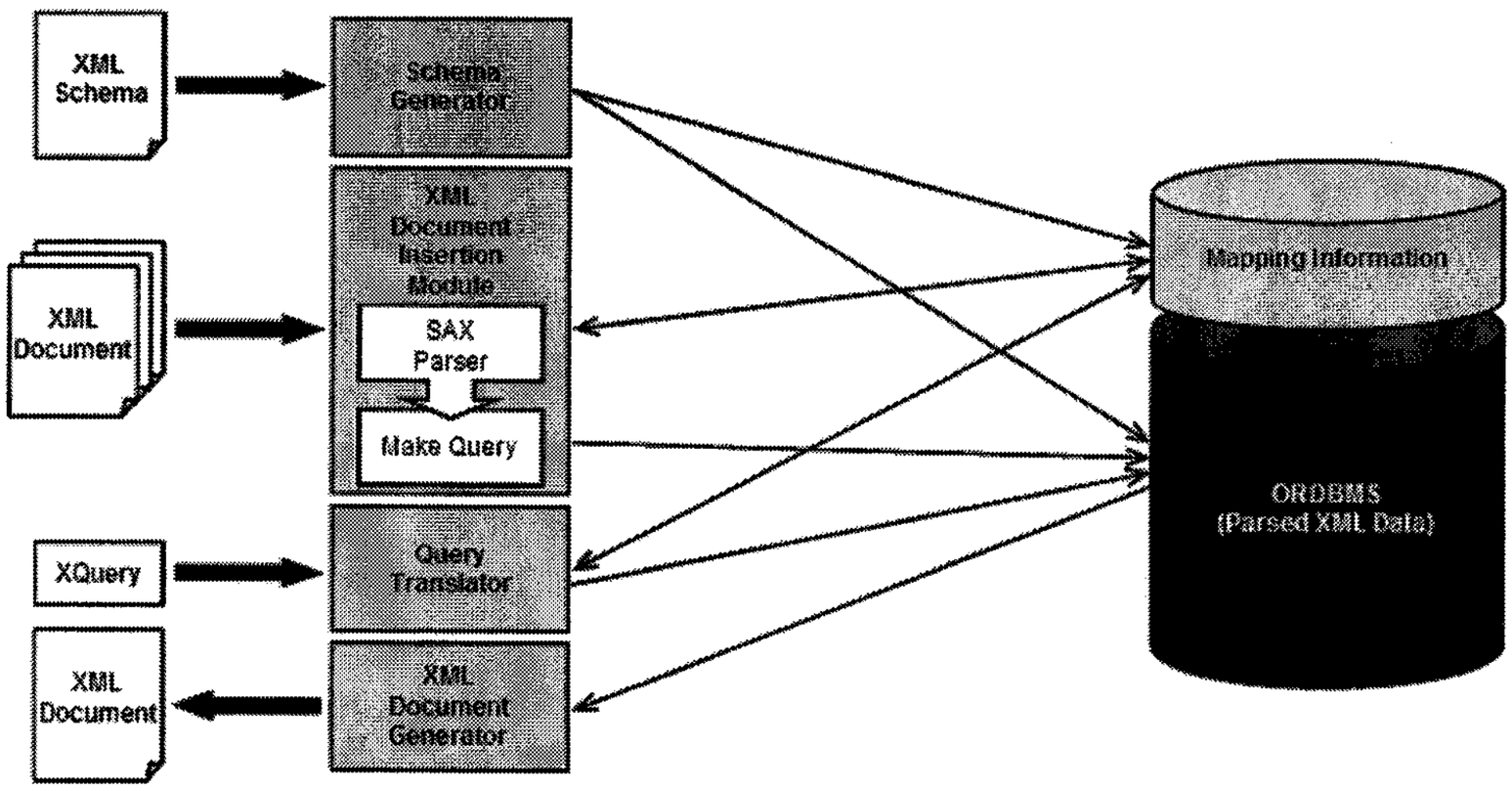

Figure 1. The structure of XDMS

Schema Generator analyzes an XML schema and generates corresponding object database schema. The relationships between XML schema and object database are

stored at Mapping Information, which is a part of the system catalog. The mapping information is useful for storing/retrieval of XML documents into/from the database. **XML Document Insertion Module** inserts XML documents into the database by using SAX Parser. Insert statements are generated for the XML documents. **Query Translator** generates SQL queries from XQuery by using Mapping Information. **XML Document Generator** publishes XML documents from the query results.

We use an object-relational database management system (ORDBMS) for storing and managing XML documents. Since XML itself conforms to the object data model, mapping from XML documents into an object-relational database is straightforward. Furthermore, the ORDBMS supports various data types and multi-valued attributes, complex XML data can be managed easily by the ORDBMSs compared with the Relational DBMSs.

3. EXPERIMENT

SBML (Systems Biology Markup Language) is a kind of XML application for providing a computer-readable format for representing models of biochemical reaction networks [11]. Many bioinformatics sites publish SBML documents for given user queries [13, 14]. In the experiment, we use XDMS as the storage system for storing and elaborate querying for the SBML documents.

Fig. 2 shows the UML notation for the SBML schema and corresponding object-relational database schema. Note that 8 multi-valued attributes have user-defined data types (i.e., classes), and the mapping is very complex in the relational databases. However, in the object-relational database they can be transformed into set-valued attributes and classes, respectively.

SBML schema	Object-Relational database schema
id : Oid {use="optional"}	id : Oid
name : string {use="optional"}	name : String
functionDefinition : FunctionDefinition [0...*]	functionDefinition: FunctionDefinition {SET}
unitDefinition : UnitDefinition [0...*]	unitDefinition : UnitDefinition {SET}
compartment : Compartment [0...*]	compartment : Compartment {SET}
species : Species [0...*]	species : Species {SET}
parameter : Parameter [0...*]	parameter : Parameter {SET}
rule : Rule [0...*]	rule : Rule {SET}
reaction : Reaction [0...*]	reaction : Reaction {SET}
event : Event [0...*]	event : Event {SET}

Figure 2. SBML schema (in UML form) and corresponding object-relational database schema

Fig. 3 (a) and (b) show the comparison of the relational database (RDB) schema and object-relational database (ORDB) schema for the SBML schema in Fig. 2. Relational database has many tables (11 tables) compared with object-relational database (4 classes) because of the gap between XML schema and relational databases. The real experiment shows that storage requirement in XDMS is much less than that of relational databases.

4. CONCLUSION

We proposed an XML document management system (XDMS) for efficient storage and retrieval of XML documents. XDMS uses an object-relational DBMS instead of relational DBMSs or native DBMSs for the storage of the XML documents. XDMS generates object database schema from XML schema and store XML documents into the object database. Experiment shows that object-relational database systems provide efficient storage and retrieval of XML documents compared with conventional systems, which use relational databases or native XML DBMSs.

Acknowledgments

This work was supported by the Chungbuk BIT Research-Oriented University Consortium and the 2nd Brain Korea Project.

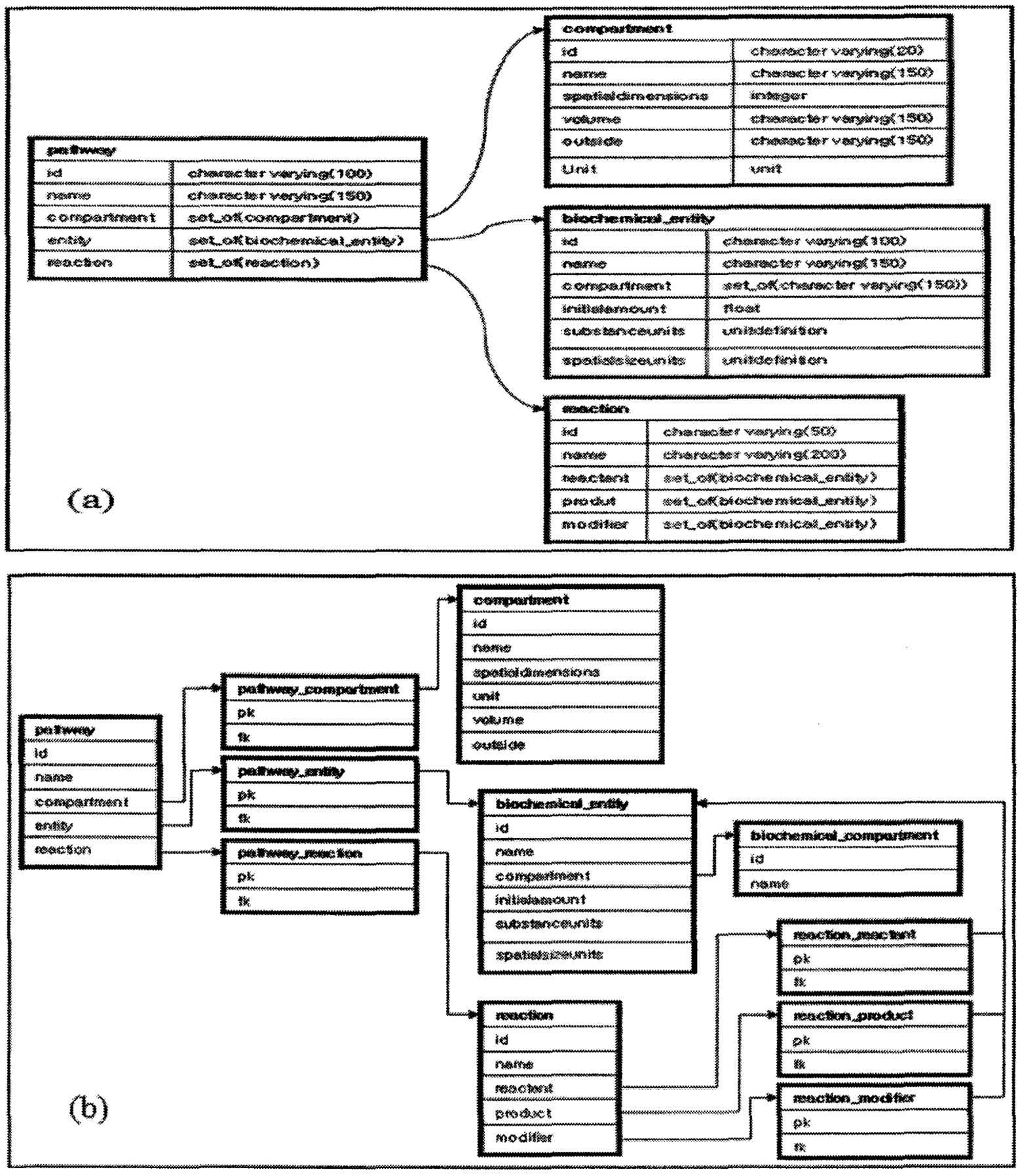

Figure 3. Comparison of database schema: (a) ORDB, (b) RDB

References:

[1] I. Tatarinov and S. D. Viglas. (2002). "Storing and Querying Ordered XML Using a Relational Database System, "In *Proc. Intl. Conf. on Management of Data*", ACM SIGMOD.

[2] Elliotte Rusty Harold, Adjunct Professor. (2005). "*Managing XML data : Native XML databases*", IBM Web Site, http://www-128.ibm.com/developerworks/xml/library/x-mxd4.html

[3] J. Shanmugasundaram, K. Tufte, G. He, C. Zhang, D. DeWitt, and J. Naughton. (1999). "Relational databases for querying xml documents: Limitations and opportunities", In *Proc. Intl. Conf. on 25th VLDB*.

[4] Seung-Hyun Jung, Tae-Sung Jung, Tae-Kyung Kim, Kyoung-Ran Kim, Jae-Soo Yoo and Wan-Sup Cho. (2006). "An Efficient Storage Model for the SBML Documents Using Object Databases", *VLDB Workshop on Data Mining and Bioinformatics*.

[5] T. K. Kim and W. S. Cho. (2005). "A DTD-dependent XML Data Management System : An Object-Relational Approach," In *Proc. International Conference on Internet & Multimedia Systems & Applications (IMSA)*, Hawaii, 248-253.

[6] K. Runapongsa, J. M. Patel. (2002). "Storing and Querying XML Data in Object-Relational DBMSs," *EDBT Workshop*, pp.266-285.

[7] UniSQL Web Site, http://www.cubrid.com/

[8] Tamino XML database Web Site, http://www.softwareag.com/

[9] eXcelon Inc. Web Site, http://www.exceloncorp.com/

[10] W3C Web Site, http://www.w3.org/

[11] SBML Web Site, http://sbml.org/index.psp

[12] XML Web Site, http://www.w3.org/XML/

[13] KEGG Web Site, http://www.genome.jp/kegg/

[14] NCBI Web Site, http://www.ncbi.nlm.nih.gov/