HPC@UNM2003-003
September 2003

HPC@UNM
THE CENTER FOR HIGH PERFORMANCE COMPUTING

TECHNICAL REPORT

XML Schema and Web Services for ETL in the Staging Area of a Scientific Data Warehouse

Authors

Mykola Dudar
The University of New Mexico
The Center for High Performance Computing
Department of Computer Science

Owen Eddins
The University of New Mexico
The Center for High Performance Computing

Susan M. Wilson
The University of New Mexico
The Center for High Performance Computing

Susan R. Atlas
The University of New Mexico
The Center for High Performance Computing
Department of Physics and Astronomy and Center for Advanced Studies

Robert Veroff
The University of New Mexico
Department of Computer Science

The University of New Mexico
The Center for High Performance Computing
MSC01 1190 Albuquerque, NM 87131-001
505.277.8249 *(main)* 505.277.8235 *(fax)*
www.hpc.unm.edu

The University of New Mexico

Disclaimer

The Center for High Performance Computing at the University of New Mexico (HPC@UNM) provides a focus for high performance computing and communication at the University of New Mexico (UNM). HPC@UNM is committed to innovative research in computational and computer science with emphasis on both algorithm development and applications. As part of the commitment, HPC@UNM sponsors this technical report series. The technical reports are subject to internal review by HPC@UNM. However, the material, as presented, does not necessarily reflect any position of HPC@UNM. Further, neither UNM nor HPC@UNM makes any warranty or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information contained in this report.

Marc S. Ingber, Director, HPC@UNM

Barney Maccabe, Associate Director, HPC@UNM

XML Schema and Web Services for ETL in the Staging Area of a Scientific Data Warehouse

Mykola Dudar[1,2], Owen Eddins[2], Susan M. Wilson[2], Susan R. Atlas[2,3] and Robert Veroff[1,]*

[1]Department of Computer Science, [2]Center for High Performance Computing, [3]Department of Physics and Astronomy and Center for Advanced Studies, University of New Mexico, MS C01 1190, 1 University of New Mexico, Albuquerque, NM 87131, USA

Abstract

The Extraction, Transformation, and Loading (ETL) of disparate sources of operational data into the integrated staging area of a data warehouse is one of the most complex and time-consuming problems facing a data warehouse designer. This task is further complicated when the warehouse is being designed to support scientific research and analysis. In this paper, we describe a novel approach for organizing the staging area of a scientific data warehouse based on a Web services architecture. The design takes advantage of established open standards for XML and Web services, and utilizes XML Schema in a dual role, to provide both the native data exchange format and the data model for the staging area database. An important feature of the design is a logical and physical shift of the data integration process away from warehouse developers and toward data providers. For scientific applications, this shift is expected to yield significantly improved data quality, since individual domain experts will be best qualified to map their respective data models to the warehouse exchange specification. The proposed solution leverages considerable research and development investments in specialized data exchange formats within the scientific community and has enabled the creation of a large-scale bioinformatics data warehouse for archiving, integrating, and analyzing genomic data.

1. Introduction

The design of a data warehouse for scientific data storage and analysis is an ambitious undertaking due to the extreme heterogeneity of the domains that supply the relevant data and the corresponding data sources and data structures themselves. In the bioinformatics application that motivated this work, data sources may, for example, include image and floating point data files from gene expression microarray experiments, clinical data, laboratory experiment annotations, gene ontologies, and literature citations—all of which must be stored in the data warehouse and organized so as to make the information easily accessible to scientist researchers for correlation and analysis [35]. In general, the designer of a data warehouse for supporting scientific research faces a

* To whom correspondence should be addressed.

very complex integration task. In comparatively young fields such as genomics, the task is further complicated by the lack of standards for organizing the data, or even a consensus on what data to store. To succeed, the data warehouse design must be flexible and readily extensible, so that it can be modified to accommodate data from new domains and with new data structures as the nature of the data sources evolves over time.

The Extraction, Transformation and Loading (ETL) of disparate sources of data into the integrated staging area of the data warehouse is a fundamental component of data warehouse design. The staging area serves as a primary entry point into the data warehouse, where data is cleansed of nonessential, incorrect, inconsistent and redundant entries, then stored in a common format to enable consistent interpretation of similar data from different providers. It is generally acknowledged that the design of efficient data staging solutions is extremely difficult, requiring a considerable investment of time and resources [1]; this challenge is compounded for scientific applications, due to the heterogeneous nature of the data sources and schemas and the lack of standardized representations.

In our research on the design of a large-scale bioinformatics data warehouse to support computational biology and biomedical research at the University of New Mexico [2,3], we elected to focus initially on ETL and the design of the staging area, since design choices in this key part of the data warehouse will significantly affect the structure of the other warehouse components (data repository, data marts) accessed by the scientist end users, as well as the quality and completeness of the stored data. The central challenge was to identify an effective and extensible mechanism for integrating a highly complex and diverse set of data sources within the staging area. The solution described in this paper is based on open standard Web services and XML (Extensible Markup Language) technologies. Although motivated by a particular scientific application, our approach is quite general and can be applied to any domain amenable to XML-based data exchange.

The key insight that led to the development of our design was the recognition that the problem of designing an efficient staging area for a data warehouse is conceptually similar to the problem of developing a standardized data model for information exchange among members of a particular user community. XML [4,5] was developed in 1996 to serve as a foundation for data exchange and to provide metadata markup, and has achieved rapid acceptance in both the scientific and commercial worlds. A number of important domain-specific scientific XML standards have emerged in the past several years, aimed at expediting data exchange and enabling collaborating groups to submit supplementary data into centralized repositories [36]. Examples include GeneXML [6] and MAGE-ML [7] (genomics), Chemical ML [8] (chemistry), and the Ecological Metadata Language [9] (ecology). It also is becoming increasingly common for scientific funding agencies to encourage collaborating groups within a research program share data using XML [9,10]. The fact that many of our potential data providers and subscribers already were using XML as their method of data exchange suggested the novel "XML-centric" solution for addressing the data warehouse ETL and integration problem. In particular, we observed that collaborating scientists could easily submit complementary data into our centralized repository if we required that the data be presented in XML format. This approach is particularly appealing for our bioinformatics application, because each laboratory has researchers who specialize in different analysis techniques. This requires very flexible data models that are both standardized as well as cover all

required content. As our work progressed, we identified several additional advantages of the Web services/XML architecture, for example, the reuse of existing solutions for data exchange models and the facilitation of application development requiring partial or complete integration of data from various sources.

To the best of our knowledge, the primary role played by XML in data warehouse applications to date has been in the implementation of data marts and OLAP (on-line analytical processing) cubes [11-13]. We have found only a few examples of its use in data ETL and integration [14,15]. The prototype data warehouse described here makes use of a full range of XML technologies to implement the scientific data ETL, integration into the staging area, and data cleansing. These include XML, XML Schema [16], and Web Services technologies [17]. We have used Oracle 9iR2 XML DB for the data warehouse staging area and repository because of its advanced capabilities for storing XML "natively" in an object-relational database. Additional features of the large-scale bioinformatics data warehouse being implemented at the University of New Mexico using this staging area design will be described in a planned series of future papers.

The remainder of the paper is organized as follows. In Section 2, we provide a brief description of the various XML and Web Services technologies underlying the design of the data warehouse. Section 3 gives a detailed description of the data warehouse architecture. In Section 4, we present a detailed discussion of the advantages and potential pitfalls of our data warehouse architecture. We conclude in Section 5 with a summary and plans for future work.

2. XML and XML-based technologies

There are numerous research and development efforts currently underway that are based on or closely related to markup languages**.** The notion of organizing data in a "description-valued" manner supports the natural way that humans process data. XML can be regarded as the most widely adopted open standard for packaging information together with a description of its meaning. The inherent flexibility, simplicity and extensibility of markup construction allows it to be used in many ways, ranging from the generation of web pages and simple text files, to complex data management systems, distributed applications, network protocols, etc. In this section, we describe the current status and directions of XML and XML-based technologies. Readers already familiar with these technologies can skip this section and proceed directly to the data warehouse architecture description in Section 3.

The Extensible Markup Language was introduced in 1996 in a publication by the World Wide Web Consortium (W3C) [4,5]. It immediately attracted significant attention due to its simplicity and flexibility. XML enables the presentation of information in a textual format using a markup-based approach, where each piece of data is surrounded by short labels indicating its meaning. The language definition also supports the definition of constraints on the logical structure of an XML document using a Document Type Declaration (DTD) grammar. This effectively makes each XML document a self-documenting piece of information. Both XML and the DTD are open standards. This makes it possible for anyone to create a DTD which defines a particular type of XML document with some specialized logical structure. In addition, the standardization of

XML and DTD makes it possible to create corresponding software tools that can process XML data in a standard way.

One of the pitfalls of the DTD specification is that it defines only a limited set of constraints on the content of an XML document. For instance, it has no facility for defining a complex data type for an element. Therefore, one can define a logical structure for a document, but the content of the document cannot be restricted. XML Schema was introduced in 1999 to address the limitations of DTDs [16]. In addition to defining the structure of an XML document, it also declares allowable content and semantics of data inside the documents. An XML document that meets all of the constraints imposed by its XML Schema is described as *conforming to its XML Schema* or as a *valid XML document*. Any XML document can be transformed into another XML document using the XSLT language [18]; also, by using XSL [19], the same XML document can be transformed from structured text such as XML into other textual representations.

With XML Schema, one can construct a data exchange schema and also build systems of data storage and management for XML content, which we will refer to in this paper as *XML DB* (XML database). Currently all major manufacturers of database management systems (DBMS) have implemented solutions for XML DB [20-23]. These products perform the following functions: store XML documents, validate XML documents against their XML Schema, transform documents using XSLT, and provide an interface for extracting documents.

A major shift in how developers of distributed applications think about data exchange is occurring with the emergence of Web Services technology [17]. A *Web service* is defined using the Web Services Description Language (WSDL). WSDL contains the description of remote procedure calls (RPC) in a stateless, one-way message exchange protocol. SOAP (Simple Object Access Protocol) [24] is the open standard definition for RPC used by applications to define a mechanism for the exchange of structured and typed information. For the developer of a client application of a particular web service, it suffices to know the Internet address of the target web service along with its definition in WSDL. In the current state of the technology, generation of the stub parts of both Web service provider and client applications is greatly automated [25,26]. It requires minimal technical knowledge of SOAP or WSDL, and makes the overall development of the source very simple and straightforward.

These XML-based technologies and techniques have significantly shifted the paradigm for data exchange and storage in the IT world. They are part of an effort to improve interoperability of distributed information systems [27]. The issues involved in designing the staging area of a data warehouse are similar to those encountered in designing any distributed system: integration of data from different sources, extraction and loading mechanism, and extensibility of the construction. Because of this, and because of the characteristics of Web services and XML technologies, we view them as viable solutions for performing ETL and integration. This is particularly true for a warehouse designed to support scientific applications, since, as noted above, the acceptance of XML in the scientific community has been particularly rapid and pervasive. The extraction and transformation of data from relational database management systems (RDBMS) and subsequent transformation into XML documents is a well-studied issue [20,21]. SQL/XML has recently emerged as part of the ANSI/ISO

SQL standard [28]; this is an extension to standard SQL that extracts data from an RDBMS and transforms it into an XML document. All major relational database vendors have committed to supporting this new component of SQL. Transferring the resulting XML documents using a Web services architecture is then relatively straightforward. Recent innovations in mapping XML to object-relational data models in a database [20] make it possible to implement XML as part of a data warehouse, without major changes to the overall architecture. In light of these considerations, we decided to investigate how a Web services architecture and XML Schema could be utilized together in the design of a scientific data warehouse, and our initial results are presented in the current paper.

3. Web Services and XML Schema in the Staging Area of a Scientific Data Warehouse

The overall architecture of a scientific data warehouse based on a Web services architecture and utilizing XML Schema in the staging area is presented in Fig. 1.

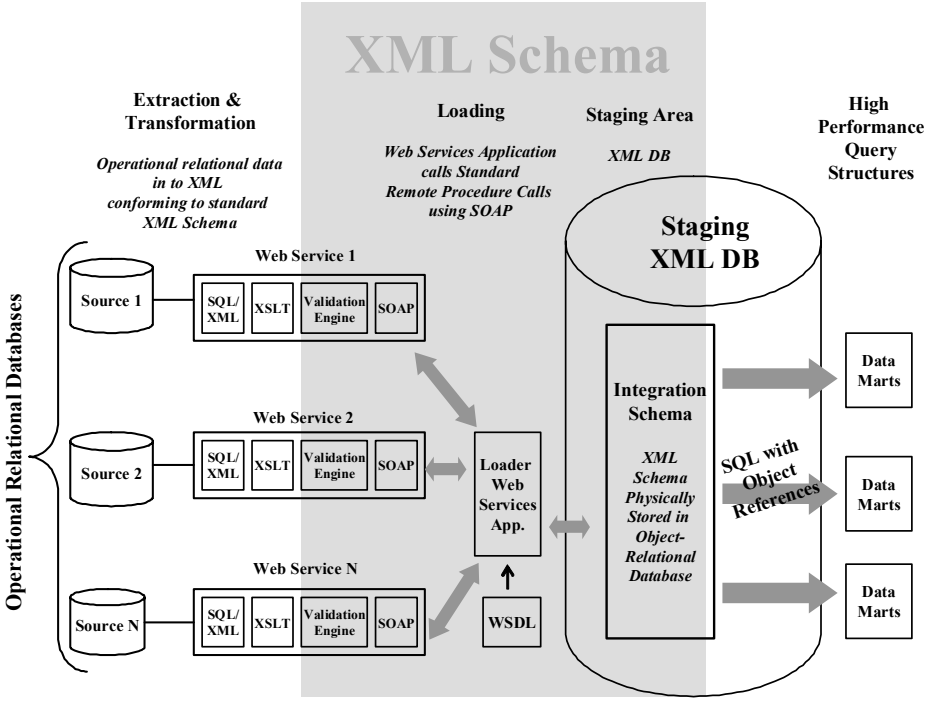

Fig. 1. Overall architecture of a scientific data warehouse utilizing XML Schema and a Web services architecture in the staging area.

There are two major differences between Fig. 1 and a classical data warehouse architecture: we use a Web services architecture to completely organize ETL activities; and we use a prototype XML schema in a dual role. The schema defines the exchange data format of all XML documents that are pulled from online transaction processing (OLTP) systems into the staging database. In addition, the same XML schema defines the data model of the staging XML DB. The *loader* (indicated in Fig. 1) is a web service application that takes the WSDL and issues SOAP messages to the web services. The

extraction and transformation of the data from the operational DBMS into XML documents is implemented by web services. A Web service can be reduced to the following functional blocks: extraction/partial transformation, full transformation, validation, and RPC interface. SQL/XML performs the extraction from the operational DBMS and the partial transformation of the data into an intermediate XML document that roughly models the schema from which it was extracted. The intermediate XML document is then transformed, using XSLT, into an XML document that conforms to the exchange schema. To verify conformance to the exchange schema, the XML document is then checked using an XML Schema validation engine.

The loader is an application that stands next to the staging XML DB. The responsibility of the loader is to request from Web services the extraction and sending of "new" data and—when the XML instances with the "new" data have been received—to store it in the staging database and trigger the necessary activities for integrating the newly arrived data with the data already present in the data warehouse repository. To accomplish this, the loader sends a SOAP message to a Web service requesting updates. After data has been extracted and the corresponding XML document prepared, it is sent back to the loader in another SOAP message. The loader takes the XML document and moves it into the staging area using the XML database's programmatic interface. The XML document is now physically stored in the object-relational database, whose data model is defined by the exchange schema. SQL with object references is then used to move the data from its object-relational structure into the data marts (see Fig. 1), where it can be accessed by users. This last step is the 'bridge' between the Web services architecture and moving the data into more conventional data warehousing structures such as the data marts.

In our initial implementation of the architecture, we have chosen to use technologies and solutions provided by Oracle 9iR2 [20]. However, we stress that although our particular solution has been built using an Oracle platform, the concept and architecture of ETL in the data warehouse using XML Schema and a Web Services architecture is entirely non-proprietary in nature. Similar solutions could be built using other open-source [29] or commercial [21] platforms. We elected to use Oracle technologies for defining the staging XML database primarily because in addition to providing "native" storage for XML content, it also can be queried as a standard SQL database with object-relational (OR) extensions. Alternatively, we could achieve the same functionality using a native XML DB supported by open-standards query languages such as XQuery [30] and XPath [31].

4. Discussion

In this section, we discuss several significant consequences of our staging area architecture, namely integration shift, facilitation of ETL, and the ability to create an entirely XML-based warehouse. We feel that the integration shift is the most important characteristic of this work, because it will lead to enhanced data integrity and better integration of loosely coupled data sets.

The design of a scientific data warehouse differs significantly from a conventional data warehouse with respect to the level of participation expected from the data providers. The data providers of a scientific date warehouse are likely to also be among

the primary users of the data marts. It therefore is in their best interests to participate in an efficient data warehouse solution, even if there is some additional burden placed on them at the stage when the data is initially uploaded into the warehouse.

In a conventional data warehouse, data arrives at the staging area in independent formats; it is then interpreted, transformed, and loaded into the warehouse. Thus, the integration of the data from the operational sources to the staging area is not completed until the transformed data is actually loaded into the staging area.

In our approach, an XML schema specifies the data exchange model to the data providers. This enables us to develop a Web services architecture where the integration of the data is completed immediately after the data has been extracted from a data source's schema and transformed into an XML document that conforms to the exchange schema. The XML document is then moved to the staging area using the Web services architecture. In other words, as a result of the Web services architecture, the integration of the data is logically and physically shifted toward the providers of the data and their source-specific implementations of extraction and transformation.

An integration shift is desirable in our domain because of the complexity and heterogeneity of the bioinformatics data being stored in the data warehouse. Shifting the integration to the provider will improve data quality because the providers are responsible for mapping their data models to the exchange specification, and also for validating their XML documents before providing them to the web services client application. Data providers have an intimate knowledge of which attributes of their data model map to which elements of the exchange specification. In order to facilitate the mapping of the data for the providers, and to make an exchange schema usable and clear for data providers, a well-documented specification must be built, describing the content and constraints of the required elements. An exchange schema is not meant to model the complete internal schema of every operational data source, but rather it provides a schema for mapping the essential data required by the warehouse. This "contract-among-parties" approach to sharing data provides an easier mapping to the exchange schema and better integration and cleansing of the data.

An advantage of the architecture is that it is extensible and can integrate new data that does not map directly to the existing exchange schema's data model. Known relationships between the current exchange schema and the new data are used to update the schema. XML Schema is a flexible language designed to accommodate these types of extensions. A Web service likewise has two features that make it interoperable and extensible. The first is that the data is exchanged *in a standard format* between the data sources' web services and the client application. The second is that this exchange occurs via a standard (RPC) protocol. Therefore, the Web services client can avail itself of data from every data source using a single procedure to import data from multiple sources. The developer writes to the procedure and not to the data.

Another important characteristic of this architecture is the role that XML schema performs in addition to its role as an exchange schema. It has been shown that XML Schema structures map directly to object-relational database structures [37]. Oracle 9iR2 supports the automatic registration of XML Schema documents in the database, which automatically creates an object-relational database representation of the XML Schema. The dual role of the XML Schema as an exchange and object-relational staging schema

allows the complicated data modeling attributes of the XML Schema language to be realized in the data storage model.

The discussion in this paper has focused on XML and Web services technologies as a solution for ETL and the integration of data into a data warehouse. Looking beyond the staging area to the overall warehouse architecture, it is worth noting that an entirely "XML-centric' data warehouse architecture could be achieved by extending the use of these technologies to data marts as well. Research into using XML in various capacities for data marts and Web warehousing is presently underway by others [11,13,32,33]. All of these approaches implement their data storage in the form of XML documents, and any of them could be implemented using the XML/Web services-based architecture described here.

The issue of reusability has been an important motivating concern of this work. As we have already noted, designing the staging area and ETL component of a new data warehouse—even if one takes advantage of the approach described here—requires significant effort. The use of a Web services architecture and standardized XML data exchange schema makes it possible to leverage this architecture in deploying applications other than data warehouses. In the bioinformatics domain, for example, we are investigating the integration of engines for reporting and visualizing experimental data, as well as a large-scale cheminformatics [34] database interfaced to genomics data, for the acceleration of lead identification in pharmaceutical design.

5. Summary and Future Work

In this paper, we have presented a novel design for the staging area of a scientific data warehouse, to support research in computational biology and genomics. This problem has presented significant challenges for the bioinformatics community to date, due to the lack of uniform standards for organizing the data and the inherent complexity and heterogeneity of the data. We have proposed leveraging ongoing efforts in the standardization of scientific data exchange formats in developing our staging area design, using a Web services architecture to organize ETL activities and XML Schema as an integration schema. This combination has allowed us to create an extensible and integrated solution for this part of the data warehouse. An important advantage of our approach is the shift of the data integration from the staging area, in a direction closer to the data providers. We regard this as a very important issue for our domain, because the data providers are expected to have the greatest familiarity with their data and can best appreciate how to integrate it into the standardized exchange specification. We have further noted that although we have described a solution for our particular domain of bioinformatics, and based the implementation on specific choices of software platforms, the approach itself is entirely general and can be applied to any domain facing similar design issues. Moreover, although our approach, as described, is primarily intended for a data warehousing application, it can potentially find application in other contexts requiring partial or complete data integration from diverse data sources.

In future work, we are planning to continue our investigation of the use of XML technologies in data warehousing to address other architectural components of the warehouse. We also are beginning to explore the integration of a cheminformatic

database at UNM with the genomic component of our data warehouse, to facilitate rapid, large-scale data-mining for cancer drug discovery applications.

Acknowledgements

This work was supported in part by the U.S. National Science Foundation under grant No. DBI-0078307 to the National Center for Genome Resources (NCGR) and UNM, and by grants to the University of New Mexico from the W.M. Keck Foundation and the D.H.H.S. National Institutes of Health/National Cancer Institute (CA88361). M.D. thanks I. Smirnova for stimulating discussions during the early stages of this work. The authors gratefully acknowledge Dr. W. Beavis (NCGR) and Profs. S. Ruby, C. Willman, and P. Helman (UNM) for their encouragement, M. Murphy and Dr. S. Matthias for helpful discussions, and the University of New Mexico Center for High Performance Computing for generous computational and database support.

References

[1] P. Vassiliadis, A. Simitsis, and S. Skiadopoulos. Conceptual modeling for ETL processes. In: *Proceedings of the ACM Third International Workshop on Data Warehousing and OLAP (DOLAP)*, pp. 14-21 (DOLAP '02, McLean, VA, 2002).

[2] M. Mosquera-Caro, *et al*. Gene expression profiling for molecular classification and outcome prediction in infant leukemia reveals novel biologic clusters, etiologies and pathways for treatment failure. Preprint, University of New Mexico, to be submitted (2003).

[3] P. Helman, R. Veroff, S. R. Atlas, and C. Willman. A Bayesian network classification methodology for gene expression data. Technical Report TR-CS-2002−18, Department of Computer Science, University of New Mexico, J. Comp. Biol., submitted (2003).

[4] World Wide Web Consortium (W3C), Extensible Markup Language, W3C Working Draft, 14 November, 1996. See: http://www.w3.org/TR/WD-xml-961114.

[5] World Wide Web Consortium, Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation, 6 October, 2000. See: http://www.w3c.org/TR/2000/ /REC-xml-20001006.

[6] H. Mangalam, J. Stewart, J. Zhou, K. Schlauch, M. Waugh, G. Chen, A. D. Farmer, G. Collello, and J. W. Weller. GeneX: An open source gene expression database and integrated tool set. IBM Systems Journal 40(2):552-569 (2001).

[7] P. T. Spellman, M. Miller, J. Stewart, C. Troup, U. Sarkans, S. Chervitz, D. Bernhart, G. Sherlock, C. Ball, M. Lepage, M. Swiatek, W. L. Marks, J. Goncalves, S. Markel, D. Iordan, M. Shojatalab, A. Pizarro, J. White, R. Hubley, E. Deutsch, M. Senger, B. J. Aronow, A. Robinson, D. Basset, C. J. Stoeckert, Jr, and A. Brazma. Design and implementation of microarray gene expression markup language (MAGE-ML). Genome Biology 3 (9):research0046.1-research0046.9 (2002).

[8] CML (Chemical Markup Language), a new approach to managing molecular information using XML and Java based strictly on SGML. See: http://www.xmlcml.org/.

[9] Ecological Metadata Language Specification. See: http://knb.ecoinformatics.org/software/eml/.

[10] Data Specifications for The National Cancer Institute Director's Challenge. See: http://dc.nci.nih.gov/tools/DataManagement.

[11] N. T. Binh, A. M. Tjoa, and O. Mangisengi. MetaCube-X: An XML metadata foundation for interoperability search among Web warehouses. In: *Proceedings of the International Workshop on Design and Management of Data Warehouses (DMDW 2001)* D. Theodoratos, J. Hammer, M. Jeusfeld, and M. Staudt, Eds., Interlaken, Switzerland, June, 2001.

[12] R. M. Bruckner, T. W. Ling, O. Mangisengi, and A. M. Tjoa. A framework for a multidimensional OLAP model using topic maps. In: *Proceedings of the Second International Conference on Web Information Systems Engineering (WISE 2001), Web Semantics Workshop,* 2:109-118 (IEEE Computer Society Press, Kyoto, Japan, 2001).

[13] M. R. Jensen, T. H. Møller, and T. B. Pedersen. Specifying OLAP cubes on XML Data. Technical Report 01-5003, Department of Computer Science, Aalborg University (2001).

[14] B. Ensink, K. Haveman, T. Schavey, and M. Shrestha. XML based adaptation of the composite approach for database integration. In: *Proceedings of the 37th Annual ACM Southeastern Conference (CDROM)*. (ACM Press, New York, NY, 1999).

[15] Z. G. Ives, A. Y. Halevy, and D. S. Weld. An XML query engine for network-bound data. The VLDB Journal 11(4):380-402 (2002).

[16] World Wide Web Consortium (W3C). XML Schema. See: http://www.w3.org/XML/Schema.

[17] World Wide Web Consortium (W3C). Web Services Architecture. See: http://www.w3.org /TR/ws-arch/.

[18] World WideWebConsortium. XSL Transformations (XSLT), Version 1.0, W3C Recommendation 16 November 1999. See: http://www.w3c.org/TR/1999/REC-xslt-19991116.

[19] World Wide Web Consortium (W3C). Extensible Stylesheet Language (XSL), Version 1.0, W3C Recommendation 15 October 2001. See: http://www.w3.org/TR/xsl/.

[20] Oracle 9i Database. See: http://oracle.com/ip/deploy/database/oracle9i/.

[21] SQLXML and XML Mapping technologies. See: http://msdn.microsoft.com/sqlxml.

[22] IBM DB2 XML Extender. See: http://www3.ibm.com/software/data/db2/extenders/xmlext/xmlextbroch.pdf.

[23] NeoCore XML Management System. See: http://www.neocore.com/.

[24] Wold Wide Web Consortium. SOAP Version 1.2: Part 1: Messaging Framework, Recommendation 24 June 2003. See: http://www.w3.org/TR/2003/REC-soap12-part1-20030624/.

[25] Oracle 9i JDeveloper 9.0.3 New Features. See: http://otn.oracle.com/products/jdev/htdocs/jdev03_fo.html.

[26] R. Caron. Getting started with XML Web Services in Visual Basic.NET and Visual C#, Microsoft Developer Network, October, 2002.

[27] About the World Wide Web Consortium. See: http://www.w3.org/Consortium.

[28] A. Eisenberg, J. Melton. SQL/XML and the SQLX Informal Group of Companies. See: http://www.acm.org/sigmod/record/issues/0109/.

[29] Apache Cocoon. See: http://cocoon.apache.org/2.0/.

[30] World Wide Web Consortium (W3C). XQuery 1.0: An XML Query Language, W3C Working Draft, 2 May 2003. See: http://www.w3.org/TR/xquery/.

[31] World Wide Web Consortium (W3C). XML Path Language (XPath), Version 1.0, W3C Recommendation 16 November 1999. See: http://www.w3.org/TR/xpath.

[32] D. Pedersen, K. Riis, and T. B. Pedersen. XML-extended OLAP querying. Technical Report 01-5003, Department of Computer Science, Aalborg University (2001).

[33] O. Mangisengi, J. Huber, C. Hawel, and W. Essmayr. *A Framework for Supporting Interoperability of Data Warehouse Islands Using XML*. (Springer-Verlag, New York, 2001).

[34] I. Muegge and M. Rarey. Small Molecule Docking and Scoring. In: *Reviews in Computational Chemistry, Vol. 17*, K. B. Lipkowitz and D. B. Boyd, Eds., pp. 1-60 (John Wiley and Sons, New York, 2001).

[35] K. Fellenberg, N. C. Hauser, B. Brors, J. D. Hoheisel, and M. Vingron. Microarray data warehouse allowing for inclusion of experiment annotations in statistical analysis. Bioinformatics 18(3):423-433 (2002).

[36] F. Achard, G. Vaysseix, and E. Barillot. XML, bioinformatics and data integration. Bioinformatics 17(2):115-125 (2001).

[37] W. S. Han, K. H. Lee, and B. S. Lee. An XML storage system for object-oriented/object-relational DBMSs. Journal of Object Technology 2(3):113-126 (2003).