A PROTOTYPE XML-BASED IMPLEMENTATION OF AN INTEGRATED 'INTELLIGENT' NEONATAL INTENSIVE CARE UNIT

Christina Catley[1] and Dr. Monique Frize[1, 2]

[1] Department of Systems and Computer Engineering, Carleton University, Ottawa, ON, Canada
[2] School of Information Tech & Eng., 161 Louis Pasteur, University of Ottawa, Ottawa, ON, Canada

Abstract- **As a growing number of studies suggest that clinical decision support systems (CDSSs) have a positive impact on reducing medical errors, the significance of integrated CDSSs becomes evident. Our Medical Intelligent Decision Aid Systems (IDEAs) Research Group is exploring this potential by deploying a prototype implementation of integrated clinical decision support in a Neonatal Intensive Care Unit (NICU), the goal being to evaluate the effectiveness of real-time decision support. Our paper presents the eXtended Markup Language (XML)-based technological foundation for this integrated 'intelligent' NICU. This includes a description of the logical data manipulations required to process medical data, produce an XML-enabled CDSS result, and transmit this information to the NICU client in real-time.**

Keywords- **Clinical Decision Support Systems, NICU, integration, XML, XSLT, XML Schema, WML**

I. INTRODUCTION

Clinical decision support systems (CDSSs) have the potential to greatly improve decision making in health care settings. As suggested by Ball: "*Clinical decision support systems are being used to enhance decision making and improve efficiency in diverse health care environments, from acute care to ambulatory practice. As clinical decision support systems become more commonplace, they will feed the data repositories that are key to evidence-based medicine"* [1]. An excellent example of such a repository is the real-time database being implemented at the Children's Hospital of Eastern Ontario (CHEO) Neonatal Intensive Care Unit (NICU), where real-time information from the NICU devices feeds the date repository. All patient information will be remotely accessible to enable the sharing of data with secure external sites, such as other Hospital Information System (HIS) units, and to allow remote, wireless patient monitoring by physicians.

An integrated solution for all the NICU monitoring equipment has been developed and implemented, as of September 2002, by Agilent Technologies. The integrated solution includes a patient documentation centre, online patient monitoring and database access, an enhanced report generator, and patient vital signs monitoring. Working on this foundation, our research team's objective is to extend the current Agilent Technologies information centre to incorporate real-time clinical decision support for physicians- in effect creating an 'intelligent' NICU. With this in mind, the data received from the NICU real-time data repository must support a broad range of applications, since the data must be able to be processed through Artificial Neural Network (ANN), Case-Based Reasoning (CBR) systems or other CDSS software based on the physicians' runtime preferences.

As previously discussed, our research team is enabling CDSS application integration by creating an eXtended Markup Language (XML)-enabled interface between our group's CDSSs; promoting seamless processing of XML data by a wide variety of clinical decision support tools [2]. This paper will provide a detailed description of the manner in which the XML family of standards is employed to achieve CDSS application integration in the NICU, the information flow through the integrated system, and a discussion of our ongoing work.

II. METHODOLOGY

The methodology used to achieve CDSS application integration within the NICU is based on the World Wide Web Consortium's (W3C) XML family of standards: namely XML, eXtended Stylesheet Language Transforms (XSLT), and the XML Schema Language. This section elaborates on these standards, followed by a description of the NICU system's deployment diagram.

A. *The XML Family of Standards*

From a health care perspective XML has an important role to play in promoting the acceptance of computer applications within the medical community. In fact, XML has become an integral component in many clinical systems now under development [3], and while the number of XML-based applications currently being developed is already large, this number is expected to increase dramatically in the near future [4].

XML makes data portable by separating the presentation of the data from the data content; in addition users are able to define original content by creating new XML schemas or Document Type Definitions (DTDs). Schemas, or DTDs, are sent with the data (or referenced within the XML document) to describe how the receiving application will interpret the markup tags, thereby creating self-describing data. The schemas are interpreted and potentially validated by XML parsers and subsequently directed to enterprise applications [2]. In addition, both schemas and DTDs can be used as a basis of comparison to determine if an XML document's structure is well formed (e.g. conforms to the rules of XML, also known as document validation), and to identify errors in the data itself. Error identification is known as 'data validation' or 'schema validation' and is essential in medical environments where the data integrity is crucial.

XSLT is a W3C standard that transforms XML content from one form into another for the purpose of promoting interoperability between vocabularies. Using XSLTs, XML data can be transformed into another XML document, an HTML page, a C++ file, or any other format for which the developer creates a matching XSL transform.

B. *XML Tools: Processors and Parsers*

XSL transforms are processed by XML processors, one of the most common of which is the Xalan XML processor from the Apache open-source project. The ability to transform XML documents into diverse forms is essential in our prototype implementation, where XSL transforms are used to transform XML documents representing patient data into a CDSS-processable format based on the specific type of clinical decision support required.

An XML parser interprets an XML document so that it can be used by applications. A popular parser is the Xerces4J parser created by the Apache open-source project; it forms the basis for both IBM's XML Parser for Java (XML4J) and Sun's Java API for XML Parsing (JAXP). XML parsers are normally categorised as being validating or non-validating and as supporting either the Document Object Model (DOM) or the Simple API for XML (SAX).

The DOM is a recommendation of the W3C; it provides a common interface for manipulating document structures with the aim that DOM-compliant code should be interoperable between different parsers. SAX parsers are event-driven rather than structure oriented. The parser alerts the application that an event has been generated within the document, it is left to the application code to handle these events. The choice of parser is based largely on what the application is meant to do with the data. Applications that work with large bodies of data where speed is a requirement would be best suited to SAX, since only the necessary events are handled. Applications that need to parse and store information for later use are best suited to the DOM.

C. *NICU Integrated CDSS Deployment Model*

A deployment diagram is a Unified Modeling Language (UML) artefact for describing the physical distribution of a system across a set of processing nodes, where a node represents either a processor or a device. The integrated NICU's deployment diagram is shown in Fig. 1. Connectivity is Internet-based for external clients and Intranet-based for hospital information system clients. XML provides a standard means of data representation for creating a portable and interoperable system where HTTP is the communication protocol.

The Simple Object Access Protocol (SOAP) is a possible future means of transmitting XML data between distributed network locations. SOAP represents only one transmission possibility; Internet communities are repeatedly proposing the use of XML in network protocols and distributed applications- SOAP, XML-RPC, and XML Metadata Interchange (XMI) are only a few examples [5]. SOAP provides messaging for web services applications; a web service is an interface that describes a collection of operations that are network-accessible through standardized XML-messaging. The potential of web service delivery in health care offers a strong case for employing standardized SOAP envelopes in XML messaging. Additionally, employing protocols such as SOAP is beneficial because the envelope level can manage issues such as guaranteed delivery, transmission errors, and routing information [6].

III. RESULTS

Regardless of whether XML processing is performed locally or in a distributed environment, several key data manipulations must be achieved that are reusable throughout

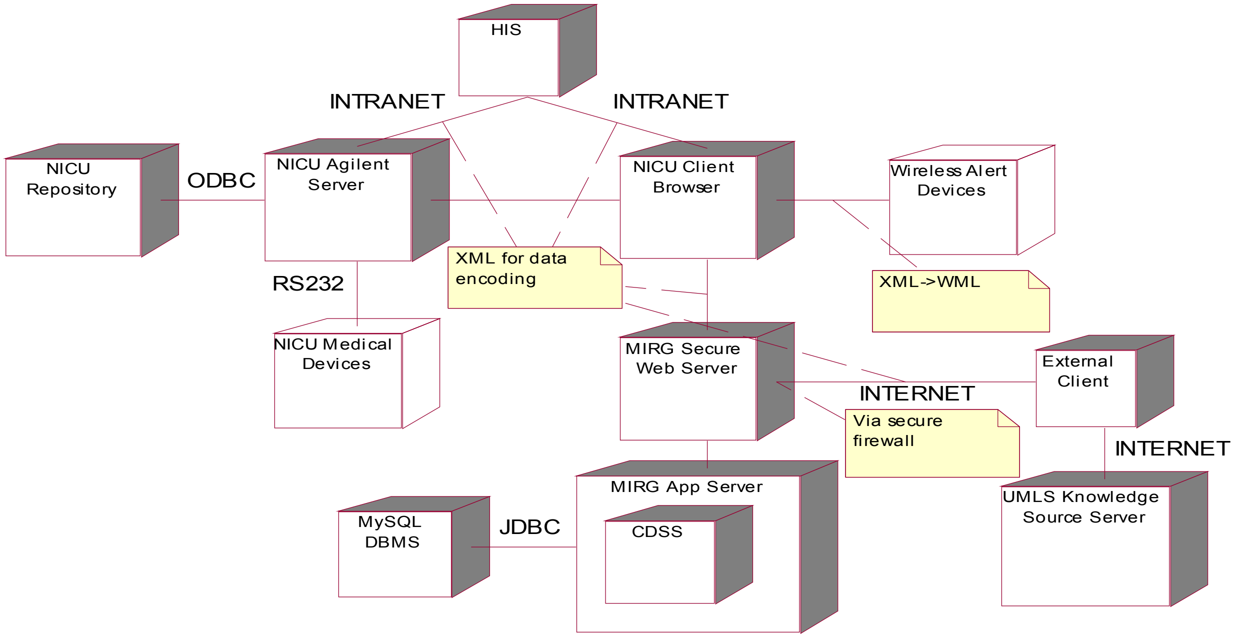

Figure 1 : Integrated NICU UML Deployment Diagram

the implementation. Fig. 2 illustrates how these manipulations are employed in a distributed environment where the Medical IDEAs Research Group's (MIRG) application server handles requests from CHEO's NICU clients; the numbering scheme used in fig. 2 corresponds to the numbered list below.

(1) Initially, the system requires the creation of standardized XML Schemas based on medical ontologies. The schemas are subsequently stored in MIRG's Schema Repository. The Unified Medical Language System's (UMLS) metathesaurus provides a user interface for standardizing medical terminology by mapping medical terms to UMLS-determined Concept Unique Identifier (CUI) values.

(2) At the NICU client-side, repository data received from medical devices must be converted to XML prior to transmission to MIRG's application server for CDSS processing. This transformation is performed by code that allows relational database information to be automatically converted to XML, based on referenced XML schemas. There is a different stored schema for each possible CDSS as well as for each different combination of minimum data sets used by that CDSS to predict outcomes. Rigidly constraining the CDSS input data is possible with XML Schema 1.0, as it allows the specification of complex data types such as string, numeric types, time-related types and XML 1.0 tokenized types (ENTITY, ID, etc..) [7], as well as allowing the user to specify the acceptable range of attribute values.

(3) To realise this system, XML documents must be stored at the MIRG server-side for auditing, processing and system evaluation purposes. Basically, there are two different methods of storing XML documents:

a) Convert data to another model, such as a relational or object-oriented model.

b) Store the XML documents in their native format using an XML enabled DataBase Management System (DBMS) that provides document indexing.

To decide which method is most applicable one needs to consider the context in which the data is being used. Bourret suggests data that has a highly regular structure, such as scientific and medical data, and is used by non-XML applications, will most likely be stored in a relational database and software will be required to transfer the data between XML documents and the database [8]. Bourret refers to this as data-centric data, where XML is mainly used for transport and processing rather than for display purposes. It follows that the decision was made to transform data received at the MIRG server into a relational model to be stored for future auditing purposes. Kim explains that using a relational model allows the XML document to be mapped directly into the relational database server. *"The use of a relational database that maps all the tags and fields of the XML document into relational tables and columns allows for all the information in the XML document to be stored and retrieved in full context"* [9].

At the MIRG server-side the relational database

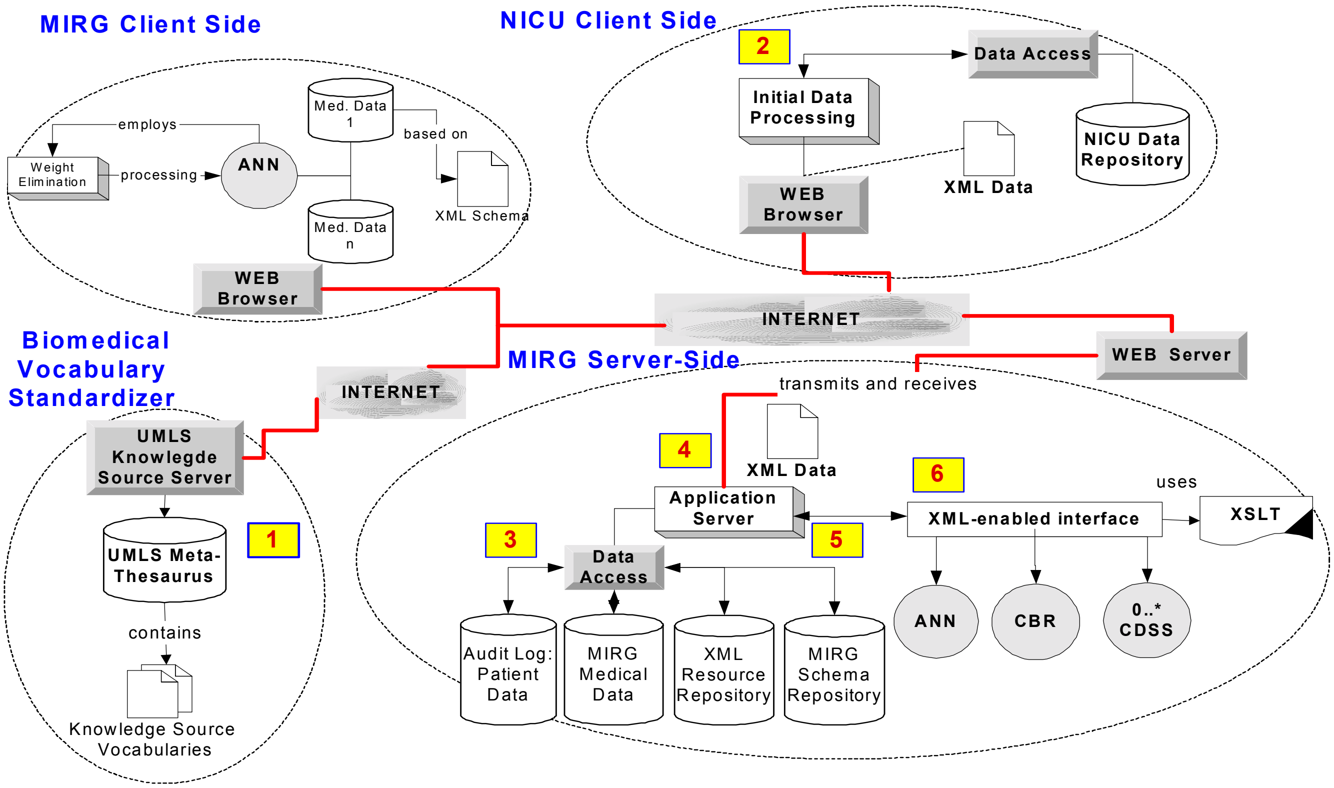

Figure 2 : XML-Based Information Flow in Integrated NICU

management system employed was the open source MySQL DBMS and the connection to this database was achieved via Java Database Connectivity (JDBC), a vendor independent method for accessing databases using Java. In this instance the choice of SQL DBMS is irrelevant as long as the appropriate database driver is referenced. It is worthwhile to note that the NICU client-side (or any other generic CDSS-client) does not use XML only as a transport mechanism; XML is also employed to display CDSS results in the browser. For this reason, implementing a DBMS that stores XML data in native format at the client could be advantageous in terms of reducing the number of data interfaces. As native XML databases evolve, this will be considered as future work.

(4) Upon receiving a client request, the MIRG application server parses the XML-based patient document to extract specific elements and attributes required for CDSS processing, such as the CDSS identifier and the required output. The parser used in this implementation is based on the DOM as the received data must be stored for future use.

(5) After retrieving all user CDSS processing parameters, validation is performed based on either the DTD or schema dependent on the document's specification. XML documents received from both internal and external sources must be validated prior to medical processing to ensure they match the CDSS' data constraints.

(6) Lastly, the data must be processed based on specific XML transforms to achieve a CDSS-ready data format. As discussed, the W3C's XSLT is used to create an interoperable interface between MIRG's CDSSs. Such interfaces enable 'transparent' data-sharing of standardized XML data for processing by multiple CDSS applications. XSLT transforms were created for converting the XML documents into a format that can be processed by either the ANN or the CBR. Additionally, the results emitted from both the ANN and the CBR must be transformed into XML format and displayed on the client's browser. This requires another set of stored transforms, one for each CDSS.

V. Discussion

The prototype implementation described in this paper is only a first step in creating an 'intelligent' and integrated NICU, where XML-based technologies were chosen as a flexible and scaleable foundation on which to build. One of the deciding factors in employing XML is its ability to share information throughout the Hospital Information System (HIS) and with other users over the Internet. The desire to share information within the HIS also motivated the creation of a separate MIRG application server, since ideally MIRG's CDSSs will process data received from throughout the HIS, rather than solely serving the NICU. A further goal is to integrate results received from MIRG's CDSSs directly into the patient's HIS electronic patient record.

An extension of this prototype will involve implementing automatic CDSS invocation based on real-time patient repository data. At this time, the focus is on manual physician-invoked support. Automatic invocation will require defining acceptable thresholds for the repository medical device values; exceeding these thresholds would cause automatic CDSS-invocation. Currently, work is also underway to create physician-defined threshold levels for each specific CDSS output and to integrate these thresholds into the CDSS logic to generate real-time alerts. This includes employing wireless technologies, such as the Wireless Markup Language (WML), to transmit XML-based patient alert data to physicians' wireless devices.

V. Conclusion

Clinical decision support systems are being advocated as a means of reducing medical errors and improving decision making in health care. The prototype system described in this paper will allow our group to analyse the effectiveness of CDSSs in the NICU both qualitatively, through physician feedback, and quantitatively, via repository data analysis. Thus, the prototype serves as a foundation for deploying MIRG's neural networks and case-based reasoning tools in the NICU and evaluating the long-term impacts on physician decision-making. Real-time NICU CDSS data processing has the potential to be very unique; currently most decision support used in hospital environments provide knowledge or handle administrative work; actual deployment and usage of CDSSs in real-time clinical settings is currently limited.

References

[1] Ball M J, Douglas J V, Lillis J, Health Informatics: Managing Information to Deliver Value. Proc. of Medinfo 2001;305-308.

[2] Catley C, Frize M, Design of a Health Care Architecture for Medical Data Interoperability and Application Integration. Proc. Joint BMES/EMBS Conference, Houston 2002; 1952-1953.

[3] Okstein C, XML: A Key Technology for Sharing Clinical Information. MD Computing 1999;16(9):31-33.

[4] Stalidis G, Prentza A, Vlachos I, Maglavera S, Koutsouris D. Medical Support System for Continuation of Care Based on Web Technology. International Journal of Medical Informatics 2001;64(2):385-400.

[5] Bonifati A, Ceri S, Paraboschi S. Pushing Reactive Services to XML Repositories Using Active Rules. Commun. of ACM 2001;633-641.

[6] Gregory A. XML Schema Design for Business-to-Business e-Commerce. XML Europe 2000; http://www.gca.org/papers/xmleurope2000/author/s21-01auth1.html, last access May 30, 2002.

[7] Chase, N. XML and Java from Scratch. Indianapolis, Indiana, QUE Publishers, 2001.

[8] Bourret R. XML and Databases, XMLDBMS; http://www.bourret.com/xml, last access June 2, 2002.

[9] Kim J, Feng D D, Cai T W, Eberl S. Integrated Multimedia Data Agent in E-Health. Conferences in Research and Practice in Information Technology, Vol 11, http://www.cs.usyd.edu.au/~vip2001/proceedings/jinman1.pdf, last access June 11, 2002.