Using Specification-Driven Concepts for Distributed Data Management and Dissemination

M. Brian Blake[1]
The MITRE Corporation
7515 Colshire Drive MS N420
McLean, VA 22102
(703) 883-7894
bblake@mitre.org

ABSTRACT

At the MITRE Corporation-Center for Advanced Aviation System Development (CAASD), software engineers work closely with both analyst and domain experts to develop software simulations in the air traffic management domain. In this environment, software simulations are applications that take large amounts of real-world operational information, and through calculations, derivations, and display extends the original information to produce some new insight into the domain. This new insight or knowledge typically comes in the form of a pertinent set of data. Based on this set of information other research groups can further extend this knowledge. The challenge in this environment is a distributed data management system that will allow a distributed set of researchers to share their extended knowledge. This paper presents the motivation and design of such an architecture to support this collaborative knowledge/data sharing environment. This run-time configurable architecture is implemented using web-based technologies such as the Extensible Markup Language (XML), Java Servlets, Extensible Stylesheets (XSL), and a relational database management system (RDBMS).

Categories and Subject Descriptors

D.2.11 [**Software Engineering**]: Software Architectures

General Terms

Design, Experimentation, Human Factors, Languages,

Keywords

Data Management, XML, XSL, Java servlets

1. INTRODUCTION

At the MITRE Corporation-Center for Advanced Aviation System Development (CAASD), researchers develop simulations for both design-time and real-time analysis. This research constitutes a wealth of knowledge in the area of air traffic management and control. This division of MITRE is split into a large number of individual groups that investigate various problems comprising the air traffic domain. Although the groups analyze different problems, the data to support the investigations are typically the same. Also, software engineers in these individual groups design and develop simulations that require the data in different formats (i.e. specialized delimited text files, database format, XML, etc.) Moreover, each group looks at different subsets of data that may cross multiple data sources. Researchers are currently provided with data from outside sources that is gathered and distributed by a data librarian. This data is usually distributed in the same media and format in which it is obtained. The CAASD Repository System (CRS) team at MITRE has identified the need for obtaining the desired raw data from outside sources and building a composite data repository that serves the need of this diverse environment. In addition, when a group is able to perform calculations on base data that extends its utility, the CRS is equipped to allow that group to share that knowledge to other groups who can further extend the utility of that data.

This paper presents the architecture that will allow this knowledge sharing not only for the air traffic domain but also for other domains throughout MITRE such as army strategic motions and telemetry. This approach is unlike similar more static systems [3][6]. The goal of the CRS team is to develop a dynamic architecture that will facilitate this idea of data management. In order to ensure the dynamic nature of this architecture, the CRS team takes a distributed web-based approach. The main reason for this approach is for use of the web browser which is the one universal conduit among a group of software engineers who use an extremely diverse set of languages, environments and tools.

2. THE CRS ARCHITECTURE AND IMPLEMENTATION

The CRS architecture [4] [5] was devised to support a diverse set of analyst and software engineers. These engineers internal to MITRE-CAASD use numerous technologies, programming languages, and interfaces. Web access is the one technology common to all groups. Therefore, the direction in designing the architecture was to use Internet technology as much as possible in gathering data request information, delivering the

CIKM '02, November 4-9, 2002, McLean, Virginia, USA.

[1] Author also has an affiliation with the Department of Computer Science at Georgetown University, 234 Reiss Science Building, Washington, DC 20057, blakeb@cs.georgetown.edu , (202) 687-3084

data to the customers, and integrating the sharing of their tools. The CRS architecture is composed of four autonomous modules that fit seamlessly into the Internet paradigm. These modules are the Client Interface Module, the Interface Specification Module, the Presentation and Query Module, and the Database Management Module. The Client Interface Module is supported by Internet browsers where the customers (internal software engineers) use the browsers to connect to the system.

The Specification Module and the Presentation and Query Module include software services that provide a graphical user interface. The Interface Specification module allows the customers to customize their user interface to meet their specific needs. This is important considering the diverse data needs. The Presentation and Query module allows the customer to choose a standard or specialized interface in order to request data. This can be a standard interface developed by the CRS team, that group, or one that was developed by another group. This module packages the information that will later be used in the Data Management Module. The Data Management Module contains functionality to maintain and extract data from some data repository. This layer consists of software services for extracting data from the relational database management system (RDBMS).

The CRS architecture is developed with the latest in Internet technologies. Figure 1 presents the implementation that supports the architectural details. The CRS implementation mainly uses Java-based technologies. The browsers in the Client Interface Modules connect to the Java Servlet-based components in both the Specification Module and the Presentation and Query Module. Both the User Interface component and Interface Specification component are implemented with Java Servlets. These Servlets are integrated with Java classes that fulfill the underlying query services. The Servlet in the Interface Specification module accepts information from the browsers in the Client Interface module in the form of an HttpServletRequest. This information can be parsed and used to generate the specifications for the HTML-based query form. This module stores this user preference information as an XML file in a shared file system location. In building this XML file, the module gathers database specific information using the Data Extraction Module.

This information is combined with the user preference information. Moreover, savvy software engineers can upload software executables that perform post processing of the data results. The name of this executable is included in the XML file. Subsequently, this XML can be processed with a generic XSL file to dynamically generate the HTML-based query form. This XML file is the centerpiece of the architecture as it is the basis for the execution of the system. This file contains detailed information that allows the Presentation and Query Module to be generic. The benefit here is to allow outside sources to use the same XML format, and without the Interface Specification module, have the ability to use the Presentation and Query module for data retrieval. The Java Servlet in the Presentation and Query module also receives an HttpServletRequest from the browsers. This module receives two independent messages. The first HttpServletRequest designates the particular standard or user-personalized query form to display. Once the user is presented with the query form and submits it, the second HttpServletRequest includes information that will be used to create a generic query on the database. The components in this module make use of remote registry-based services from the Data Access Components to fulfill their database needs. The Data Access Components have internal functionality to incorporate software executables that are plugged in by the software engineer

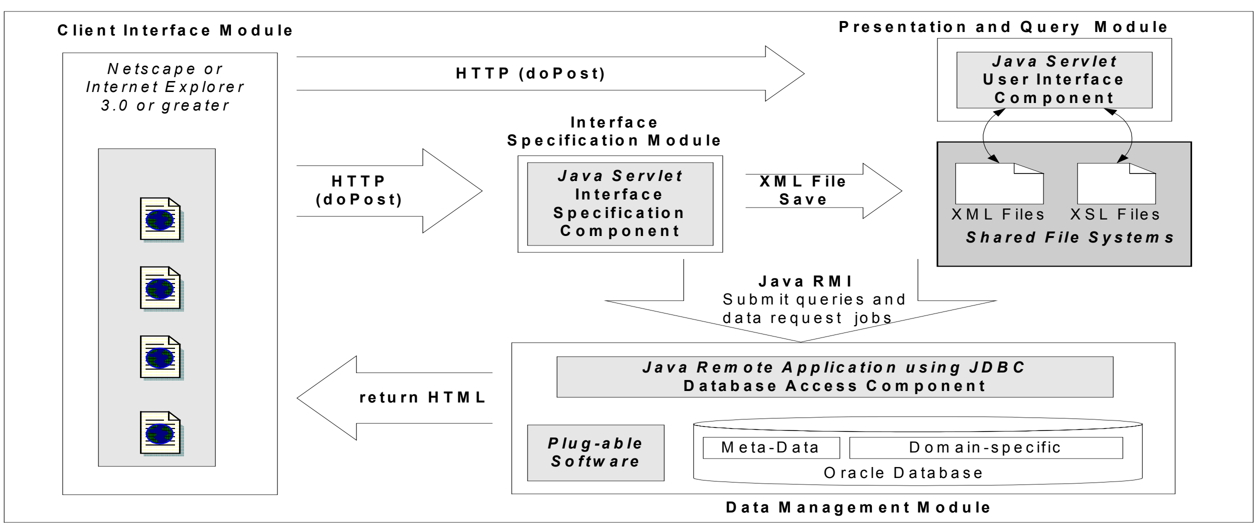

Figure 1. CRS Implementation**.**

3. DISCUSSION AND RELATED WORK

The CRS architecture is a distributed approach to data dissemination that promotes knowledge sharing. A huge benefit of this approach is that the interfaces can be specified dynamically. Therefore, the system can reside on any database and new interfaces can be added seamlessly. Currently the system is being upgraded to allow the XML-based interfaces to be stored in the database. Also, editing features are being constructed so XML files can be generated from some web-based graphical user interfaces. The CRS architecture is also being ported to another database outside of the air traffic control in the area of medical data management.

Though there is a great deal of existing work in software architecture and reconfiguration of software architectures, as discovered by this author, there is not much research towards web-accessible database architectures [1] [2]. The Zelig project introduces a schema that can be coupled with HTML to control various CGI-based database executables [7]. The main difference is that the Zelig project appears to tightly-couple the interface with the database by hard-coding entire query strings in the HTML documents. The CRS architecture extends this approach by incorporating intelligent query building knowledge in the middle-level components. The CRS interface has aspects of the query, while the middle-level components encapsulate the generic automated functionality that builds query strings dynamically. In addition, the CRS architecture has the flexibility to allow additional business-specific components to be plugged in to further format and process the database results. Moreover, the CRS implementation makes use of the later, more flexible technologies of XML and Java Servlets.

Another project that is similar to the CRS architecture is the WebInTool [9]. In the WebInTool, Hu specifies a web to database interface building tool. Hu takes a similar approach as the CRS architecture in promoting a separation of interface and source code. Hu uses several CGI-based modules but similar to the ZELIG project, there is no automated query building knowledge in these modules. In the absence of this feature, the WebInTool does not have the same ability as the CRS implementation to dynamically create a large range of queries.

Cooper [10] devises a middle-tier architecture for data management called eXtensible Data Management (XDM). This approach uses XML requests to a component-based middle-tier to which acts a virtual database of multiple databases. The main issues covered in this work are toward connecting multiple databases. Though the component-based middle-tier approach is similar to the CRS, the CRS explores issues in data dissemination, which does not seem to be the focus of XDM.

There are also industry tools that have rapid database-to-web development tools. Such tools as Oracle's WebDB [10] and Crystal Reports [11] allow developers to build graphical user interfaces that construct formatted database reports. Queries are built dynamically at design time, so forms must be rebuilt when the schema changes. The CRS is more flexible in the fact that some schema changes do not affect the system, since queries are built at run-time. In addition, each of these corporations has more complex proprietary methods to specify formats. CRS relies on the open specifications of XML and HTML.

The CRS architecture is a distributed approach to data dissemination that promotes knowledge sharing. A huge benefit of this approach is that the interfaces can be specified dynamically. Therefore, the system can reside on any database and new interfaces can be added seamlessly. Currently the system is being upgraded to allow the XML-based interfaces to be stored in the database. Also, editing features are being constructed so XML files can be generated from some web-based graphical user interfaces. The CRS architecture is also being ported to another database outside of the air traffic control in the area of medical data management.

4. REFERENCES

[1] Aizono, T., Kawano, K., Wataya ,H., and Mori, K., "Autonomous Decentralized Software Structure for Integration of Information and Control Systems, *Proceedings of the 21st International Computer Software and Applications Conference*, Washington DC, August 1997

[2] Allen, R.J., Douence, R. and Garlan, D., "Specifying and Analyzing Dynamic Software Architectures," *Proceedings of the 1998 Conference on Fundamental Approaches to Software Engineering (FASE98)*, March 1998

[3] Ambite, Jose et. al., "Simplifying Data Access: The Energy Data Collection Project", IEEE Computer, pp 47-54, Feb 2001

[4] Blake, M.B., Hamilton, G., and Hoyt, J. "Using Component-Based Development and Web Technologies to Support a Distributed Data Management System", Annals of Software Engineering , Vol. 13, No. 1, pp.13-34, April 2002, Kluwer Academic Publishers

[5] Blake, M.B. and Liquori, P. "An Autonomous Architecture for Distributed Data Management and Dissemination," *IEICE/IEEE Transaction on Information and Systems, Joint Special Issue on Autonomous Decentralized Architectures*, Vol E-48D, October 2001

[6] Klavans, J.L. and Muresan,S., "DEFINDER: Rule-Based Methods for the Extraction of Medical Terminology and their Associated Definitions from On-Line Text" Proc. 2000 American Medical Informatics Assoc. (AMIA) Bethesda, Md. 2000

[7] Shrivastava, S. and Wheater, S., "Architectural Support for Dynamic Reconfiguration of Large Scale Distributed Applications" *The 4th International Conference on Configurable Distributed Systems (CDS'98),* Annapolis, Maryland, USA, May 4-6 1998

[8] Varela, C.A. and C.C. Hayes, " Zelig: Schema-Based Generation of Soft WWW Database Applications", In Proceedings of the 1st International Conference of the World Wide Web (WWW94) Elsevier Science, Geneva, Switzerland 1994.

[9] WebInTool(2002), http://www.ri.bbsrc.ac.uk/webintool.html

[10] Cooper, B.F., Sample, N., Franklin, M.J., Olshansky, J., Shadmon, M., and Cohen, L., "Extensible Data Management in the Middle-Tier," In Proceedings of the 12th International Workshop on Research Issues in Data Engineering (RIDE'02), IEEE Computer Society Press, San Jose, California 2002

[11] Oracle Corporation (2002), WebDB Application 3.0 http://oradoc.photo.net/ora816/webdb.816/a77075/basics.htm

[12] Crystal Reports (2002), http://www.crystaldecisions.com/products/crystalreports/