Forest Inventory and Analysis Information Delivery Architecture

B. Tyler Wilson
North Central Research Station
USDA Forest Service
St. Paul, Minnesota, USA
barrywilson@fs.fed.us

Wim S. Ibes
Pillar Applications Group, Inc.
New Richmond, Wisconsin, USA
wibes@pillarapplications.com

Abstract

The Forest Inventory and Analysis program of the Forest Service, U.S. Department of Agriculture, is developing a new generation of tools based on XML, designed as a service-oriented architecture, and written in Java using established open standards and toolkits. This effort is in response to the opportunity provided by Web services technologies to create an enterprise software development environment of loosely connected applications and services. When based on communication standards rather than specific implementation technologies, these applications and services are capable of a great degree of interoperability and reuse. The U.S. E-Government initiative is aligned with this paradigm and calls for an information architecture that is service-oriented, highly interoperable, and standards-based. This document presents the organizational context and technical details of the N-tier, service-oriented architecture being developed to support this effort.

1. Introduction

The Forest Inventory and Analysis (FIA) program of the United States Department of Agriculture (USDA) Forest Service has been a source of comprehensive information for over 70 years for the assessment of the past, present, and future conditions of the United States' forests. The program is currently comprised of a set of related surveys covering aspects of forest monitoring, forest ownership, timber products output, and tree utilization. The data collected from these surveys have broad applicability for addressing multi-resource, multi-purpose issues at state, regional, and national scales.

FIA data users include researchers in academia, state foresters, consultants, forest industry, environmental advocacy groups, policymakers, and concerned citizens. These users possess a variety of means and abilities for accessing and analyzing data.

To satisfy users' diverse needs, FIA and its cooperators have developed several data and information analysis and delivery products [1]. The Forest Inventory and Analysis database (FIADB) [2] and the Forest and Rangeland Renewable Resources Planning Act database (RPADB) [3] allow for sophisticated analyses by knowledgeable users utilizing their own processing tools. Freely available desktop applications such as RPA Data Wiz [4] and FIAMODEL [5] permit users to explore the RPADB by creating maps and complex tables. Web applications such as Mapmaker [6], Ramiform [7], COLE/SOLE/MOLE [8], and the University of Georgia, USA, mapping tools [9] enable users with little analytical training to create tables, charts, and maps from the FIADB, RPADB, and Timber Products Output (TPO) database using only a Web browser.

Unfortunately, there has been much duplication of effort and little reuse of the various application code outside of the respective development groups. While many users guides are available, documentation targeted at developers is generally not. Also, these applications were developed using disparate technologies and programming languages, some of which are not supported by the Forest Service's proposed information technology (IT) infrastructure. Furthermore, most were designed as stand alone applications, each with its own data, business logic, and presentation tiers. This combination of barriers has made it difficult for programmers outside of the original development groups to add new functionality to the existing applications.

2. Key architecture drivers

The FIA program is not unique in the application development environment described in the introduction. Many governmental organizations, from federal to local, have historically followed similar ad-hoc development models focused on building software

to solve the particular business problem at hand without much concern for the software's function in the context of the whole organization.

Shifting the focus from individual projects to the enterprise for application development and delivery is one of the primary goals of some key federal initiatives on information technology that affect the Forest Service. The Clinger-Cohen Act of 1996 [10], Government Paperwork Elimination Act of 1998 [11], and the E-Government Act of 2002 [12] were enacted to make federal government business processes more efficient through better integration of both intraagency and interagency software applications. It is envisioned that these goals will be accomplished, at least in part, by adhering to interoperability standards, coordinating investments in technology, designing reusable components in decoupled systems, and utilizing forward-thinking technology.

Shortly after the announcement of the E-Government Initiative, the USDA and the Forest Service began developing plans for their enterprise architectures. Both are aligned with the Federal Enterprise Architecture reference models [13] developed by the Office of Management and Budget. Some of the most pertinent recommendations to come out of the Forest Service Enterprise Architecture Project, as they relate to the development effort described in this paper, are to create a service-oriented architecture, implement an N-tier Web framework, and build data-aware applications while using commercial off-the-shelf (COTS) or Open Source software where possible.

3. FIA Information Delivery Architecture

In response to the drivers described in the previous section, the FIA program began a project to develop a new generation of data and information analysis and delivery products. This project represents a significant movement away from the creation of stand alone applications and toward an architecture where interacting software components are designed to be interoperable in a loosely coupled manner, providing improved scalability and flexibility.

This section describes the critical software components and tiers in the FIA Information Delivery Architecture (FIDA) project and some of the details of their implementation. While technically an N-tier architecture, the various services in the FIDA project have been grouped here for clarity into the more familiar data, business, and presentation tiers.

3.1. Application development environment

The decision was made early in the project to use Java as the principal application programming language. There were several reasons behind the decision, including compliance with the agency's IT infrastructure, integration with the Oracle® Database Management System (DBMS), relative vendor neutrality, availability of numerous COTS and Open Source tools for developing and deploying the envisioned architecture, natural linkages with existing development efforts, excellent support through a large user base, and potentially low software costs.

Eclipse [14] was chosen as the Integrated Development Environment (IDE) for many of the same reasons. Its extensive use of plug-ins enables efficient execution of many project tasks including coding, debugging, compiling, versioning, systems administration, database access, and working with Extensible Markup Language (XML). Furthermore, because it is not targeted for use with any particular vendor's products, it provides a development environment that naturally encourages programmers to write Java code in an open, vendor-neutral way.

3.2. Web services environment

The FIDA project implemented a service-oriented architecture by means of Web services, as illustrated in Figure 1. The services were built upon the industry standard stack of XML-based Web service protocols: Simple Object Access Protocol (SOAP), Web Services Description Language (WSDL), and Universal Description, Discovery, and Integration (UDDI).

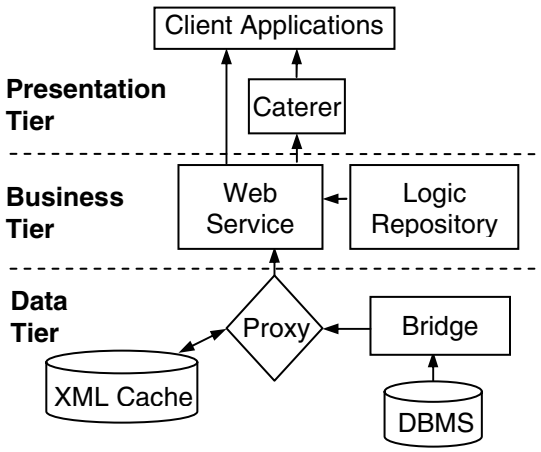

Figure 1. FIDA data flow

As stated earlier, one of the most attractive features of using Java as the programming environment was the existence of many, high quality applications and toolkits for almost any programming task. One of the most critically important applications was a software component for handling the SOAP and WSDL protocols. The Apache Axis project [15] provides a Java implementation of a SOAP server and

client. For the FIDA project, the Axis SOAP server runs as a Java servlet inside of the Jakarta Tomcat servlet container, which itself runs behind the Apache Web server.

3.3. Data tier

Within FIDA, survey data are stored as an XML document, the structure of which is described in detail by a series of XML Schema Definition (XSD) documents. The data tier accepts XQuery and XPath expressions for returning information stored in this XML document in much the same way as a DBMS accepts SQL requests against its internal tables.

Because FIA uses a DBMS for data storage, the FIDA data tier incorporates a bridge in conjunction with a proxy to move data from the DBMS into the XML document. This design ensures that the data in the XML document can be synchronized with the DBMS data source. Because the XML document is generated from the original DBMS source, it is actually a cache. To facilitate regeneration, the cache is logically segmented and can be recreated periodically as needed, either due to memory limitations or based on timestamps.

In addition to utilizing the hierarchical nature of FIA data to optimize retrievals, this design eliminates latencies that occur in the creation of connections to the database and in the execution of SQL queries. FIDA uses the Apache XMLBeans framework [16] to store and access the XML cache in memory. The cache contains no additional information that cannot readily be obtained by going directly to the DBMS.

The data tier passes all XQuery and XPath expressions through a proxy, which ensures that enough of the XML document has been loaded into the cache to fulfill the request. The proxy then passes the request to the XMLBeans XQuery engine for processing.

At the lowest level, access to the database is managed by XQuark [17], which acts as a bridge between XQuery and Structured Query Language (SQL). When a segment of the cache needs to be loaded, the proxy sends an XQuery request to XQuark. XQuark translates the request into SQL, passes the SQL to the DBMS through the servlet engine's database connection, and translates the result to an XML document fragment. This fragment is subsequently loaded into the cache segment by the proxy.

Some business logic must be introduced into the data tier to translate proxy requests into SQL. Additional logic is required to transform the result coming from XQuark into the appropriate XML cache segments. However, the logic applied in the data tier will be easily understood by users moderately familiar with the FIA dataset and can be compared to joining tables or doing simple sums in a standard SQL query. Simple XML configuration files control all business logic resident in the data tier.

3.4. Business tier

The primary purpose of the business tier is to receive client requests, convert them to one or more XQuery statements, and submit the statements to the XML document cache in the data tier. The resultant data are then wrapped in a SOAP envelope and returned to the client. The business tier also gives applications access to the XSD documents describing the FIDA XML cache. These documents are imported into the WSDL that describes the Web service.

All communication with the Web service is conducted through a series of *exchanges* known collectively as a *dialogue* within FIDA. This dialogue continues until the client sends a complete exchange to the server, i.e., an exchange with sufficient information for it to be fulfilled.

In order to bootstrap a dialogue between the client and Web service, the client sends an empty exchange. The server responds with a prototype exchange that, in conjunction with the WSDL file describing the Web service, contains all the information necessary for a client to continue the dialogue.

If the client sends a non-empty but incomplete exchange, the server responds with a similar prototype exchange, appropriately filtered by the information that has been supplied by the client. It should be noted that the exchange is devoid of presentation logic and requires no human interaction with the system. The entire process can be handled by relatively simple algorithms.

In addition to its primary role as client request broker, the business tier acts as a repository for specific business logic components. Examples range from supplying textual descriptions for specific response elements to calculating population estimates based on sample plot data.

The functionality stored in this repository is written in Java so that the methods can easily be incorporated into an Oracle® database as stored functions for use in standard SQL queries. This repository is intended to be the sole provider of business logic for all systems in use by FIDA and its clients.

3.5. Presentation tier

The primary mechanism by which the raw XML data generated by the Web service is converted to a human-readable format is through Extensible Stylesheet Language (XSL) protocols. The XSL family of languages includes XSL Transformations (XSLT) and XSL Formatting Objects (XSL-FO).

There are two types of FIDA client applications: those capable of internally transforming XML through XSL and those that are not. If the client supports XSL, it can transform the raw XML response into a human-readable format. Many standard stylesheets will be available on the FIDA server; however, clients can also create custom stylesheets as necessary.

Clients that cannot transform XML documents using XSL are directed to a Web service known within FIDA as the *caterer*. The purpose of the caterer is to dispatch requests from the client to the primary FIDA Web service, reformat the response using XSL stylesheets, and return the result to the client.

Interactive clients are built as wrapper applications that use XSL stylesheets to guide end-users through the Web service dialogue. The applications are based on workflows and presented to the end-user in the common wizard form.

The focus of the FIDA project for client development has been on browser-based interfaces, according to the recommendations of some of the architectural drivers mentioned previously. To facilitate and standardize these Web applications, the FIDA project developed a Web client framework that requires only that the browser have an XML parser plug-in. To make use of the FIDA Web mapping components, the browser must also have a Scalable Vector Graphics (SVG) [18] viewer plug-in.

Upon initialization, the Web client framework sends an empty exchange to the Web service and receives the prototype exchange in return. This exchange is placed in an ECMAScript XML Document Object Model (DOM) and stored in a permanent HTML frame. This document becomes the event dispatcher for all user interactions. When the user makes a selection, the change is relayed to the DOM. The DOM in turn triggers an event so that other elements of the page can respond to the change. For example, if a page has both a list of states and a map of states, the user may select from either the list or the map. Through the DOM and the event dispatcher, both elements will remain synchronized. When an exchange is sent to the Web service, the DOM is transformed into XML, inserted into a SOAP envelope, and then submitted either to the caterer or directly to the server.

4. Architecture products

The FIDA project has created several software products that serve the needs of FIA data users. One of the primary outcomes has been the development of Web services that give users access to forest sample plot data from the FIADB. These data can also be processed to compute population estimates and sampling errors of forest attributes from the sample data for a defined set of populations, based on FIA recommended post-stratification procedures. For example, through these services users can request estimates and sampling errors of the area of forest land by county and ownership group for the State of Minnesota, USA.

The FIDA project has designed and developed client applications that consume these Web services. Many users of FIA data utilize Microsoft® Office products in their analyses, especially the Excel spreadsheet program. To facilitate access to FIA Web services for these users, a Visual Basic® for Applications (VBA) client was developed within Excel. Through a series of forms, the user is able to specify the data to be retrieved and the ways in which the data are to be processed, if at all. The results are then placed into a spreadsheet, where they can be further manipulated.

Because of the ubiquity and ease of use of Web browsers, much effort was expended in developing a browser interface to the FIA Web services. Much like the VBA client, the Web browser interface permits the user to specify which data are to be retrieved and how they should be compiled. Through the use of stylesheets, these data can then be presented in several alternate formats. From the example given earlier, the county-level summaries for the State of Minnesota could be presented as tables, charts, or choropleth maps.

The choropleth maps generated through the Web interface are useful products on their own. However, they are often even more useful when combined with other maps, because these geospatial data put the choropleth maps into a different context. The mapping component of the Web interface allows the user to combine geospatial data from a number of Open GIS Consortium Web Map Services [19] with the maps generated from the FIA plot data.

5. Future work

In the near term, future phases of the FIDA project will incorporate additional FIA databases into the architecture. Additional services and clients will be developed to enable users to generate tables, charts, and maps from the TPO and Woodland Owners databases, as well as forest health data from the FIADB. New logic will be added to compute estimates and sampling errors from these data for user-defined populations of interest.

In the longer term, as more Web services are deployed by federal agencies, a national UDDI registry will need to be constructed. Once this registry is completed, new FIDA client applications will be developed that consume services from external sources and integrate them with FIA data to perform new analyses. For example, a Web application could be built to combine data from Web services provided

by the U.S. Bureau of the Census with FIA plot data to explore the socioeconomic dimensions of forestry.

Looking ahead even further, it is hoped that the FIDA project will feed into the Federal Chief Information Officers Council's vision of semantic interoperability within the government sector [20]. By defining ontologies for FIA Web services using XML-based standards such as the Resource Description Framework Schema (RDFS) and the Web Ontology Language for Services (OWL-S), Semantic Web services could be created [21]. In conjunction with WSDL and UDDI, these protocols would enable software agents that consume these services to better "understand" what is being exchanged and to "infer" additional information from these services, further enabling automation and interoperability.

References

[1] "Forest Inventory and Analysis National Program—Tools and Data", U.S. Dept. of Agriculture, Forest Service, http://fia.fs.fed.us/tools-data/.

[2] P.D. Miles, G.J. Brand, C.L. Alerich, L.F. Bednar, S.W. Woudenberg, J.F. Glover, and E.N. Ezell, *The Forest Inventory and Analysis database description and users manual, version 1.0, General Technical Report NC-218*, U.S. Dept. of Agriculture, Forest Service, North Central Research Station, St. Paul, MN, 2001.

[3] W.B. Smith, P.D. Miles, J.S. Vissage, and S.A. Pugh, *Forest resources of the United States, 2002, General Technical Report NC-241*, U.S. Dept. of Agriculture, Forest Service, North Central Research Station, St. Paul, MN, 2004.

[4] S.A. Pugh, *RPA Data Wiz users guide, version 1.0, General Technical Report NC-242*, U.S. Dept. of Agriculture, Forest Service, North Central Research Station, St. Paul, MN, 2004.

[5] S.A. Pugh, D.D. Reed, K.S. Pregitzer, and P.D. Miles, *FIAMODEL: users guide, version 3.0, General Technical Report NC-223*, U.S. Dept. of Agriculture, Forest Service, North Central Research Station, St. Paul, MN, 2002.

[6] P.D. Miles, *Forest Inventory Mapmaker users guide, General Technical Report NC-221*, U.S. Dept. of Agriculture, Forest Service, North Central Research Station, St. Paul, MN, 2001.

[7] "Ramiform Home Page", U.S. Dept. of Agriculture, Forest Service, http://ncrs2.fs.fed.us/zope/ramiform/.

[8] "NCASI Statistics and Model Development Group", National Council for Air and Stream Improvement, http://ncasi.uml.edu/.

[9] "Georgia Forest Maps", University of Georgia, http://www.growthandyield.com/main/maps.htm.

[10] United States Congress Senate, *S. 1124, National Defense Authorization Act for Fiscal Year 1996*, 104th Congress, 2nd sess., 3 January 1996, http://www.access.gpo.gov/nara/publaw/104publ.html; DOCID: f:publ106.104.

[11] United States Congress House, *H.R. 4328, Making omnibus consolidated and emergency appropriations for the fiscal year ending September 30, 1999, and for other purposes*, 105th Congress, 21 October 1998, http://www.access.gpo.gov/nara/publaw/105publ.html; DOCID: f:publ277.105.

[12] United States Congress House, *H.R. 2458, E-Government Act of 2002*, 107th Congress, 2nd sess., 17 December 2002, http://www.access.gpo.gov/nara/publaw/107publ.html; DOCID: f:publ347.107.

[13] U.S. Office of Management and Budget, "Federal Enterprise Architecture", The White House, http://www.whitehouse.gov/omb/egov/a-1-fea.html.

[14] "The Eclipse Project", The Eclipse Foundation, http://www.eclipse.org/eclipse/.

[15] "The Apache Axis Project", The Apache Software Foundation, http://ws.apache.org/axis/.

[16] "The Apache XMLBeans Project", The Apache Software Foundation, http://xmlbeans.apache.org/.

[17] "The XQuark project: open source information integration components based on XML and XQuery", XQuark Group, http://xquark.objectweb.org/.

[18] "Scalable Vector Graphics (SVG) 1.1 Specification", W3C®, http://www.w3.org/TR/SVG/.

[19] "Web Map Service Implementation Specification", Open Geospatial Consortium, Inc., http://www.opengeospatial.org/docs/01-068r2.pdf.

[20] "Knowledge Management Working Group", U.S. CIO Council, http://www.km.gov/.

[21] "Semantic Web Services Interest Group", W3C®, http://www.w3.org/2002/ws/swsig/.

(All Web references accessed on May 24, 2005)