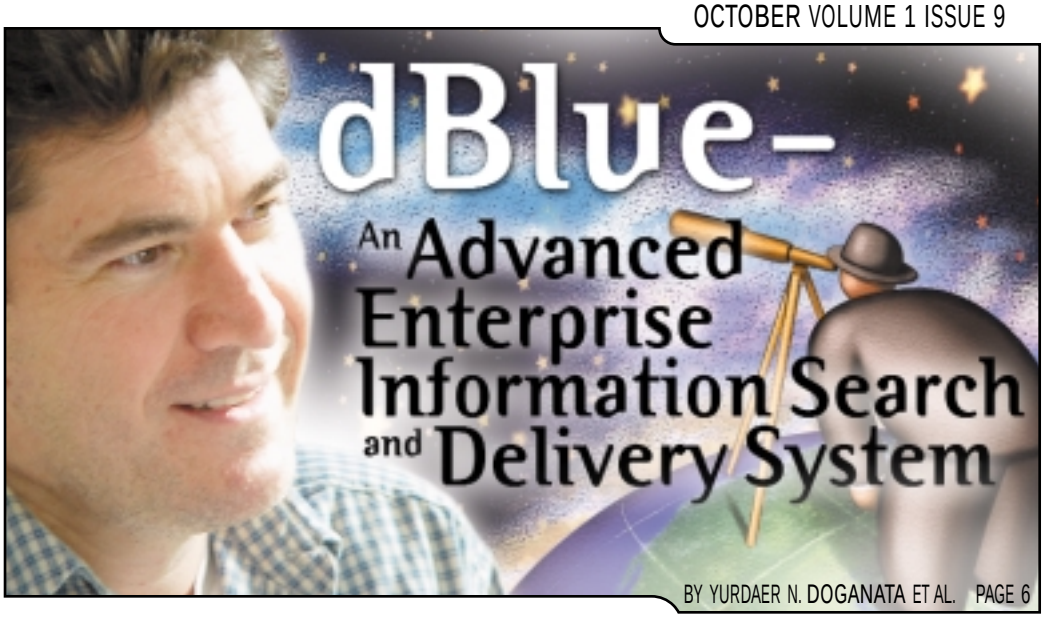
WebSphere® DEVELOPER'S JOURNAL

The World's Leading Independent WebSphere Developer Resource

WebSphereDevelopersJournal.com

OCTOBER VOLUME 1 ISSUE 9

SYS-CON MEDIA



BY YURDAER N. DOGANATA ET AL. PAGE 6

PROLIFICS

WWW.PROLIFICS.COM

WILY TECHNOLOGY

WWW.WILYTECH.COM

EDS, Sun, and Smallpox: Living in the New World

BY JACK MARTIN

Electronic Data Systems (EDS) stunned Wall Street September 18, announcing that third-quarter earnings will be 80 percent lower than previously forecast. Then Wall Street stunned EDS on September 19 by carving off over 53% of their market cap in one day.

The Texas-based company said that businesses are spending less on information technology services and that it expects the market slowdown to continue into next year, as the company has signed fewer new contracts and sees no letup in the near term.

"The slow global economy hurt; it hit us with a force that wasn't expected," EDS chief executive Dick Brown said during a conference call with analysts. "We were more optimistic than we should have been relative to our ability to fight our way through a tough economy."

So, what is EDS to do? It appears that the company will purchase Procter & Gamble's business services division – a deal that also includes a contract to manage information services for P&G, which is estimated to be worth $8 billion over 10 years and would transfer 5,700 P&G employees to EDS. Dallas-based Affiliated Computer Services announced that it was pulling out of negotiations with P&G, just as IBM Global Services, the number one information technology services company, did this year. EDS had previously announced in July that it had halted negotiations with P&G to buy the unit.

Ask yourself, why would Affiliated Computer Services, EDS, and IBM Global Services walk away from an $8 billion deal? The answer is simple. Not too long ago the P&G deal was a $10 billion deal. P&G is trying to squeeze every nickel out of the deal.

Any information technology service company desperate enough to buy business in this down market has been invited to a stupid party. Close margins and no room for error are making this a very risky deal that could lose billions over the long term for whoever gets the contract. This is not the kind of deal that builds profits; betting the ranch on selling technology to a soap company is not where the dollars are in the new world we live in.

Also, Sun is about to release a line of Linux-based desktop systems, dubbed purple boxes, that will catapult the company into a leadership position in an as-yet-undefined market: Linux on the desktop. Linux, the open-source operating system software, is typically used in server environments.

The company is going after corporate users, who won't require the full set of applications that would normally be bundled together on a consumer's desktop. Sun is targeting call-center operations, government agencies, and some education markets; those markets purchase a few million units per year.

All the prospective customers need to do is find a few million low-skilled workers who can quickly and cheaply learn to use Linux. Sure.

Sun's stock hit a 52-week low on September 19, closing at $2.70. All 22 analysts covering Sun have downgraded their estimates of the company's performance. At its new low, Sun's price/earnings ratio is still 75. Traditional value investing views a well-run company priced at 12 to 15 times earnings as a good deal. So look for a very generous target price for Sun in the next 12 months of $0.48 to $0.60 per share – assuming that the current strategy works and they do not start losing money.

In the 1990s, technology companies and dotcoms would regularly announce that they were in the process of changing the world: Microsoft with Windows, Amazon with its e-commerce Web site, Sun with Java, and countless others. The world never really seemed to change much to me – until September 11, 2001.

So now, down in Atlanta, federal health officials have put together guidelines for vaccinating the entire U.S. population against smallpox within five days in case of a bioterrorist attack. The virus could kill up to 30 percent of those infected. The virus they are talking about is real. It's not some stupid computer code designed to make your computer crash – this virus can crash your life, with no way to reboot.

People no longer care about any particular technology; they care what it can do for them.

ABOUT THE AUTHOR... Jack Martin, editor-in-chief of ***WebSphere Developer's Journal***, is cofounder and CEO of Simplex Knowledge Company, an Internet software boutique specializing in WebSphere development. Simplex developed the first remote video transmission system designed specifically for childcare centers, which received worldwide media attention, and the world's first diagnostic-quality ultrasound broadcast system. E-MAIL... jack@sys-con.com

WebSphere® DEVELOPER'S JOURNAL

ADVISORY BOARD
Richard Arone, James Martin, Greg Williams

EDITOR-IN-CHIEF JACK MARTIN
EDITORIAL DIRECTOR JEREMY GEELAN
EXECUTIVE EDITOR GAIL SCHULTZ
CONTRIBUTING EDITOR PATTI MARTIN
SPECIAL PROJECTS EDITOR RICHARD ARONE
MANAGING EDITOR JEAN CASSIDY
EDITOR NANCY VALENTINE
ASSOCIATE EDITOR JAMIE MATUSOW
ASSISTANT EDITOR JENNIFER STILLEY

WRITERS IN THIS ISSUE
Ron Ben-Natan, Robert Costello, Yurdaer N. Doganata, Youssef Drissi, Tong-Haing Fin, Joan Haggarty, Mary Hall, Marcel Heijmans, Lev Kozakov, Moon J. Kim, Christina Lau, Qi Li, Jack Martin, Frank Moreno, Michael Niemi, Lilybeth Tan

SUBSCRIPTIONS
For subscriptions and requests for bulk orders, please send your letters to Subscription Department.
SUBSCRIBE@SYS-CON.COM
Cover Price: $15/Issue Domestic: $149/YR (12 Issues)
Canada/Mexico: $169/YR Overseas: $179/YR
(U.S. Banks or Money Orders)
Back issues: $15 U.S. $20 All others

PRESIDENT AND CEO FUAT A. KIRCAALI
COO/CFO MARK HARABEDIAN
VP, BUSINESS DEVELOPMENT GRISHA DAVIDA
SENIOR VP, SALES & MARKETING CARMEN GONZALEZ
VP, INFORMATION SYSTEMS ROBERT DIAMOND
VP, SALES & MARKETING MILES SILVERMAN
VP, EVENTS GRISHA DAVIDA
PRODUCTION CONSULTANT JIM MORGAN
ASSISTANT CONTROLLER JUDITH CALNAN
ACCOUNTS PAYABLE JOAN LAROSE
ACCOUNTS RECEIVABLE KERRI VON ACHEN
ACCOUNTING CLERK BETTY WHITE
ADVERTISING ACCOUNT MANAGERS MEGAN RING-MUSSA, ROBYN FORMA
ASSOCIATE SALES MANAGERS CARRIE GEBERT, ALISA CATALANO, KRISTIN KUHNLE, LEAH HITTMAN
CONFERENCE MANAGER MICHAEL LYNCH
REGIONAL SALES MANAGERS MICHAEL PESICK, RICHARD ANDERSON
ART DIRECTOR ALEX BOTERO
ASSOCIATE ART DIRECTORS LOUIS CUFFARI, RICHARD SILVERBERG, TAMI BEATTY
WEB DESIGNERS STEPHEN KILMURRAY, CHRISTOPHER CROCE
CONTENT EDITOR LIN GOETZ
CUSTOMER SERVICE REPRESENTATIVE MARGIE DOWNS
CUSTOMER RELATIONS MANAGER RACHEL MCGOURAN

EDITORIAL OFFICES
SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645
Telephone: 201 802-3000 Fax: 201 782-9637
SUBSCRIBE@SYS-CON.COM
WebSphere® Developer's Journal (ISSN# 1535-6914) is published monthly (12 times a year).
Postmaster send address changes to:
WebSphere Developer's Journal, SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645

dBlue– An Advanced Enterprise Information Search and Delivery System

BY YURDAER N. DOGANATA, YOUSSEF DRISSI, TONG-HAING FIN, GREG BROWN, MOON J. KIM, AND LEV KOZAKOV

Fulfilling IBM's one-Web vision

One of the biggest complaints we hear about many company Web sites, from customers and employees alike, is that it's too hard to find what you need. At IBM, with 2.5 million Internet pages and more technical content than any single entity, including the Pentagon, that's no surprise.

A new IBM advanced information search and delivery system for the IBM support site (www.ibm.com/support) is expected to solve this problem. Code-named Digital Blue (dBlue), this project is a digital interface to IBM customers. The result of two years of work and five patentable inventions, dBlue is now available to IBM customers.

The team that created dBlue is calling it "the next generation of enterprise information search-and-delivery systems." This is a WebSphere-based technology with breakthroughs in storing, searching, and retrieving information. Customers will be able to search for IBM technical support information using natural language and will receive results that are categorized, prioritized, and personalized. dBlue provides the foundation for a set of user-oriented support services applicable to all IBM support sites worldwide.

Rich Vazzana, vice president of ibm.com Support and Enablement, took on this project to improve the effectiveness and performance of IBM's Web-enabled post-sales support services. It became the underlying architecture of the "one-Web" vision across multiple IBM Web sites, improving adherence to IBM's company-wide standards and setting the stage for more advanced service offerings. The program will provide customers with IBM support experience, a single IBM support/service portal, toolset, and infrastructure. Hence, cross-IBM "common" support functions will be realized.

"The business goal is to improve goal achievement on the IBM Internet," said Frank Cummiskey, director of IBM eSupport & Services. "The primary reason that customers visit IBM's support sites is to resolve a technical problem. Today, only about 60% actually achieve their goal. Improving our customers' ability to find what they are looking for, as well as to find value in the information they find, will increase self-service on the Web, saving millions of dollars and increasing customer satisfaction."

ABOUT THE AUTHOR

Dr. Yurdaer Doganata is the manager of the Integrated Networked Services group and the dBlue core research team at the IBM T.J. Watson Research Center. His current focus is on alternative techniques and methods for effective information search and retrieval in unstructured and semistructured environments. Yurdaer holds several patents and research awards and has published numerous papers. He is the technical committee cochair of ISCC '03.

E-MAIL
yurdaer@us.ibm.com

System Architecture

Although dBlue architecture does not depend on the WebSphere software platform, it's the platform of choice of the dBlue architects for its scalability, flexibility, reliability, and high performance required for dynamic Web applications hit by millions of customers every month. In addition to the application server mechanisms, the WebSphere software platform provides reliable communication middleware – the WebSphere MQ family. It also supports DB2 Universal Databases, provides a foundation for Web services, and integrates business components for text analysis and machine translation. The WebSphere Everyplace Suite provides an integrated software platform for extending the reach of business applications, enterprise data, and Internet content into the realm of pervasive computing. All this makes the WebSphere software platform the perfect fundament for the dBlue system. Figure 1 is an overview of the dBlue architecture.

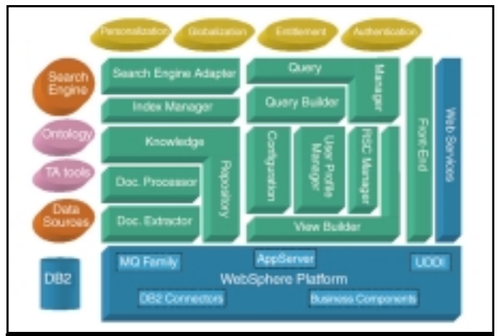


FIG. 1: DBLUE ARCHITECTURE OVERVIEW

The dBlue architecture connects three important elements from the information search world – information sources, search engines, and end users – on the basis of the WebSphere software platform. This is done through a set of components called "The Knowledge Builder." Information sources are data sources such as document repositories, DB2 and Lotus Notes databases, Web sites, and so on. Search engines are programs that can index content and enable searching of the indexed data. End users access dBlue through a front-end interface; the current default interface is a Web interface. The content is extracted from information sources using the Document Extractor and mapped to a unified XML Schema; then it's processed by the Document Processor and stored in the Knowledge Repository.

When a user accesses the system and submits a search query, the Query Manager, along with all the submitted parameters, processes this query. The Query Builder then collects the query and parameters submitted by the user, along with information coming from the user's profile and the system configuration, to build a standard Query object. The Query object is submitted to the search engine through the Search Engine Adapter. The search results flow back to the user through the Search Engine Adapter, the Search Query Manager, and the View Builder. The View Builder uses the Remote Site Customization component and data to construct a personalized view of the search hit list. When the user requests a view of a specific document, this request is processed by the View Builder, which accesses the Knowledge Repository to get the document content and builds a coherent document view.

Enabled by the WebSphere software platform, dBlue introduces various innovative solutions in the areas of information search and delivery. In dBlue:

- Content is indexed using the concept of virtual URLs.
- Search results and documents are rendered by employing dynamic layout features.
- Keyword and navigational search are combined for effective searching.
- Search results and indexing are improved by using text analysis technologies.
- Architecture is enabled for globalization and dual language search.

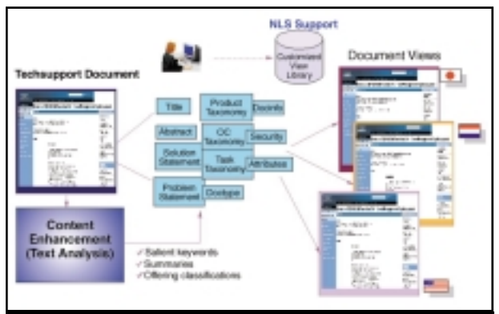


FIG. 2: CREATING MULTIPLE VIEWS FROM THE SAME CONTENT

Virtual URLs and Dynamic Layout

dBlue is a search system, but it doesn't depend on a particular search engine. The technical content to be indexed can be pushed to any search engine using the concept of virtual URLs. Until now, search systems have had to crawl content off a particular address where it's stored. Hence, the documents are replicated redundantly for the purpose of indexing the same information in a different context. With virtual URLs, documents to be indexed are built on the fly from building blocks, eliminating the need for replication and crawling. In other words, the virtual URLs aren't associated with any physically stored documents. This motivates another breakthrough in content storage. In the back end, the documents are broken down into components, such as title, problem, solution, reference, and category, allowing for true knowledge mining and the building of multiple views of the same content. Extracting the documents from their original sources and creating components based on unified XML Schema for technical documents accomplishes this, giving users a great deal of flexibility and allowing them to receive a wider range of information.

In a typical search system, the documents are stored and retrieved with a layout defined by the content providers. In this case the layout is static and cannot be changed to meet customers' needs. dBlue solves this problem by introducing the concept of dynamic layout for creating multiple views from the same content (see Figure 2).

The component-based storage system invented by the dBlue team decomposes documents into data elements without breaking the ties to their original documents. When customers request information in a specific layout, components are analyzed to ensure that they have all the necessary elements for a specific document, which is then built dynamically. This gives the flexibility to separate user experience from the content-generation process and also enables rapid localization and internationalization of the pages.

OC Taxonomy

One of the first challenges was to institute a consistent structure for content creation, since the huge amount of support content that already existed was not suitable for search. In order to structure the content and organize the content-creation process, the unified XML Schema for technical documents was created. This schema incorporates content components, such as title, abstract, problem statement, and solution statement, along with multiple attributes, keywords, references, and attachments.

The second step in organizing the content was creation of the content repository schema that allows storage of both unstructured and structured data. This schema contains more than 30 DB2 tables that provide storage for the document content, along with all associated information, and supports a variety of queries. Then, of course, both existing and new content had to be migrated to this structure. The content migration pipe is powered by the WebSphere MQ family of communication middleware. The documents extracted from their original repositories were converted to XML format based on the unified XML Schema and transferred to the new storage. All document attachments were

ABOUT THE AUTHOR

Youssef Drissi is an advisory software engineer at the IBM T.J. Watson Research Center in Hawthorne, New York. He holds several patents and is the author of several publications. Youssef is a member of the dBlue project architecture and research teams. His current work involves research, architecture, and development of next-generation unstructured information and knowledge management systems.

E-MAIL
youssefd@us.ibm.com

ABOUT THE AUTHOR

Dr. Tong-Haing Fin is a senior software engineer at the IBM T.J. Watson Research Center. He holds a number of patents and is the author of several publications. He is a member of the dBlue project architecture and research teams. His current work involves research and system integration of text analysis and knowledge management systems.

E-MAIL
thfin@us.ibm.com

ABOUT THE AUTHOR

Greg Brown is the dBlue solutions architect and team lead for OneWeb Infrastructure and Integration for the IBM.com e-Support & Service Delivery team. Greg holds several patents related to dBlue and multiple IBM.com awards.

E-MAIL

browng@us.ibm.com

ABOUT THE AUTHOR

Dr. Moon J. Kim is an IBM senior technical staff member with responsibility for the development of the e-Support advanced Web system. Moon also developed many large system solutions, such as S/390 and MPP and was involved in the development of the network systems that later called the broadband high-speed access system, including HFC and FSN. Moon is an IBM Master Inventor who holds 10 patents, has published 10 invention technical papers, and has filed 12.

E-MAIL

moonkim@us.ibm.com

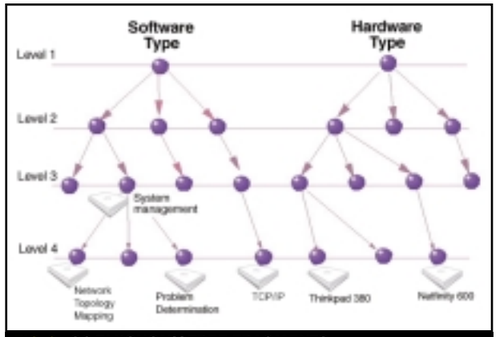


FIG. 3: OC TAXONOMY TREE FRAGMENTS

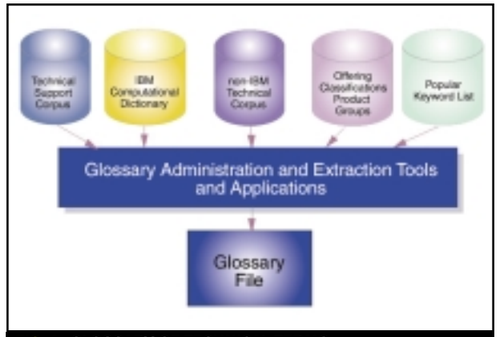


FIG. 4: GLOSSARY OF TECHNICAL TERMS

encoded using "Base64" encoding and incorporated in XML objects. To eliminate unnecessary XML parsing, the transportation was done in a binary format.

Another challenge was determining how to store and dynamically retrieve this information in a scalable and flexible way. The team adopted a categorization scheme based on IBM product offerings, called offering classification (OC). The common library classification can be used, but for IBM technical support all contents are associated with IBM products. With the OC taxonomy attached to the content, the content can easily be shown where it belongs. Figure 3 shows a fragment of the OC taxonomy tree with sample documents that may be found under certain leaves.

Having OC taxonomy information attached to the documents made it possible to combine a keyword with the navigational search. This way, users can narrow down search results with single click.

Combining Keyword with Navigation Search

The way the system is architected allows combining keyword search with navigational search. Based on a topic or a document type, users can narrow down search findings with a single click. This increases the chances of finding the requested information when the user query isn't specific enough to narrow down the search results on the first attempt. The categorized results are returned with the option of filtering the results based on IBM's product offerings and the document types.

Although combining keyword and navigational search refines the search results, it doesn't improve relevancy or precision/recall rates. The following sections discuss some text-analysis techniques used to improve precision/recall.

Content Enhancement for Search Improvement

The quality of full-text search depends mainly on query terms and on how documents are indexed by the search engine. The search results contain the documents that are indexed against the query terms and scored based on certain statistical criteria. In many real-life situations, the relevant documents can't be found or may not appear at the top of the search results because they are scored low or they don't contain the terms exactly as in the query. This is common when users choose variations of the query terms, including inflections, misspellings, abbreviations, and so on. To improve the user experience, dBlue uses text analysis tools developed by IBM Research to enhance the contents of documents. This process is started by extracting terms from a large collection of documents in the IBM technical support domain to create a domain-specific glossary. The terms in the glossary can consist of canonical form, variant form (inflection, abbreviation, misspelling, etc.), synonym, term definition, statistical data, and other information. This initial glossary is enhanced by eliminating irrelevant terms and reranking terms using special dictionaries and algorithms. The process of generating and enhancing the glossary is semi-automatic, using glossary tools and the librarian. Figure 4 shows the multiple components that compose the glossary of technical terms built for the dBlue system.

Based on the glossary, the important keywords in each document are extracted and ranked, and their related glossary terms (variants, synonyms, etc.) are used to enrich the content of the document. The content enrichment is used to create keyword metatags for biased indexing, expand the query terms to include related terms, and enable search for related documents. To improve the user's search experience, keywords are displayed in the search results and navigating through keywords is possible.

Globalization

As part of the effort to allow different languages to be supported from a single Web application consistent with the vision of "one Web" for all regions, dBlue has a globalization process that consists of two main processes: internationalization and localization.

INTERNALIZATION

Internationalization (sometimes abbreviated as i18n) is the process of designing an application so that it can be adapted to various languages and regions without engineering changes. After the internalization of dBlue software components, they can run worldwide with the addition of localized data. Hence, support for new languages doesn't require recompilation. Textual elements, such as status messages and GUI component labels, are stored outside of the source code and retrieved dynamically. Culturally dependent data, such as dates and currencies, appears in formats that conform to the end user's region and language.

The Unicode format, which handles most characters known to mankind, was instrumental in allowing the use of a unique globalized repository where multilingual searchable text and documents are encoded in one unique

format. Unicode was also adopted as a standard format for encoding internal textual data in dBlue.

LOCALIZATION

Localization (sometimes abbreviated as i10n) is the process of adapting software for a specific region or language by adding locale-specific components and translating text. Usually the most time-consuming part of the localization process is the translation of text. Other types of data, such as sounds and images, may require localization if they are culturally sensitive. Localizers also verify that the formatting of dates, numbers, and currencies conforms to local requirements.

Two innovative approaches in the globalization process are worth mentioning. The first allows documents to be searched, regardless of their language, against a query formulated in user-specific language. This is accomplished in dBlue without extra overhead or the need for a translation at runtime through a specific extension of the inverted index, a core component of most search engines. The second allows the achievement of similar results through dynamic mapping of the user's search query at run time, and use of multithreading to submit multilingual queries to the search engine. Figure 5 illustrates some aspects of this innovation.

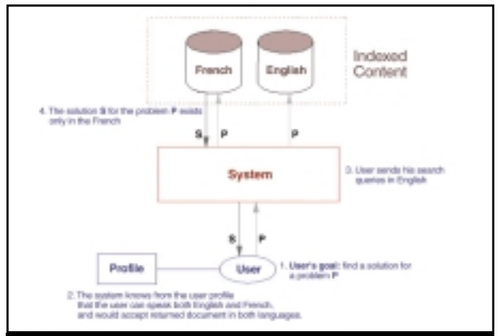


FIG. 5: MULTILANGUAGE SEARCH PROCESS

FIG. 6: REMOTE SITE CUSTOMIZATION OF RESULTS PAGE

Remote Site Customization

Another dBlue feature that addresses corporate needs is Remote Site Customization (RSC). IBM, like any other large corporation, has multiple departments that may want to present search results and technical documents to their customers in different ways, adding their own ads, promotions, and so on. The dBlue system enables this by providing the RSC feature, which allows different departments to define their own layouts for search results and technical documents. The idea of RSC is rather simple: each remote site that wants to present the shared system content in a special format is allowed to store and register its own forms. When the system gets a request that specifies this remote site, it will use the appropriate form to build the customized view of the content. Figure 6 shows the six areas that are available for customization in a results page. To assist departments in customizing the layout of Web pages, dBlue provides a Web-based RSC administrative application, which allows the uploading and testing of customized forms.

Conclusion

dBlue has many advantages. In the near future, customers will be able to ask questions in natural language and the system won't require an exact match of words. In the near future, dBlue will also personalize searching so that once a user fills out a profile, responses will be filtered and ranked based on that profile. Multilanguage searches for documents written in Japanese, Chinese, and French will be supported by late 2002. By 2Q03, it's expected that all languages will be supported from a single Web application consistent with the vision of "one Web" for all regions.

References

- *IBM WebSphere Software Platform Overview:* www7b.boulder.ibm.com/wsdd/products/platformoverview.html
- *WebSphere Everyplace Suite:* www.ibm.com/pvc/products/ wes/index.shtml
- Booth, Alan E. *Extending the Reach of Enterprise Applications with Transcoding and Machine Translation:* www7b.software.ibm.com/wsdd/library/techarticles/0206_booth/booth.html
- *WebSphere Technology for Developers (overview & downloads):* www7b.software.ibm.com/wsdd/downloads/wstechnology_tech_preview.html
- Snell James. *Implementing Web services with IBM WebSphere Version 4.0:* www7b.software. ibm.com/wsdd/library/techarticles/0108_snell/0108_snell.html
- *DB2 Product Family Overview:* www.ibm.com/software/data/db2/
- *About the Internationalization Activity:* www.w3.org/International/about.html
- *Internationalization in Java:* http://java.sun.com/docs/books/tutorial/i18n/
- *The Unicode Home Page:* www.unicode.org
- Park, Youngja; Byrd, Roy J.; and Boguraev, Branimir K. *Automatic glossary extraction: Beyond terminology identification, IBM Research Technical Report RC22421, 2002. IBM T.J. Watson Research. The Talent (Text Analysis and Language Engineering) project:* www.research.ibm.com/talent
- Boguraev, Branimir K. and Neff, Mary S. (2000). *Lexical Cohesion, Discourse Segmentation and Document Summarization.* RIAO-2000. April.
- Park, Youngja and Byrd, Roy J. (2001). *Hybrid text mining for finding terms and their abbreviations.* EMNL-2001.
- Chu-Carrol, Jennifer; Prager, John; Ravin, Yael; and Cesar, Christian. (2002). *A Hybrid Approach to Natural Language Web Services.* EMNLP-2002.

ABOUT THE AUTHOR

Dr. Lev Kozakov is a research staff member at IBM T.J. Watson Research Center and is a member of the dBlue project architecture and research teams. He has worked in many areas, including dynamic systems, applied statistics, information management systems, man-machine interface, medical software, computer telephony, and design patterns. Lev holds a number of patents and is the author of several publications.

E-MAIL

kozakov@us.ibm.com

AN INNOVATIVE APPROACH TO BUILDING PORTALS

Driving down the TCO

BY QI LI AND LILYBETH TAN

ABOUT THE AUTHOR

Qi Li is a senior consultant at Prolifics working in the specialized WebSphere Consulting division. He is responsible for delivering distributed WebSphere and Portal solutions to clients worldwide. Qi has helped organizations such as CIBER, Inc. and GE with their WebSphere and Portal initiatives.

E-MAIL
qli@!prolifics.com

Prolifics has fine-tuned a methodology and architecture for portal solutions that is quite different from the typical portal implementation. While most portal architectures connect a front-end portal interface directly to the back-office systems, we instead utilize a Service Oriented Architecture (SOA), which involves creating an enterprise repository of business services based on the Web services standard. We have found that this portal integration approach maps more naturally to the project phases and team assignments, creating a more positive experience and driving down the application's TCO (Total Cost of Ownership).

Critical to the portal solution – in fact almost the starting point – is building the centralized, technology-agnostic repository that houses the key services, modeling them in a format that is immediately accessible and testable. This serves as insulation; front-end portal developers and back-end programmers can now work against the repository interfaces, eliminating team dependencies. The repository also allows the project team to quickly demonstrate functionality to the business sponsors, enabling them to ensure that the project is right on track. Since the repository is based on Web services, an open and adaptable standard, the application is well-positioned for future growth.

With this approach, the result is a storage area of "components" that are all tested, predictable, documented, and standardized. It is easy to see how reusability becomes a reality, and therefore how overall maintenance costs can be reduced. After all, according to industry analysts, maintenance expenses account for 20–30% and sometimes reach 50%, of an application's TCO.

Figure 1 outlines the steps involved in our methodology for developing portals. For the purposes of this article, we will walk through the building of a sample application to validate and illustrate the effectiveness of this approach. The application was implemented using the following platform and tools:

- Intel Pentium III, 1133MHZ, 1GB RAM
- Microsoft Windows 2000, Service Pack 2
- DB2 Universal Database 7.2 with Fixpack 5
- IBM WebSphere Application Server Advanced Single Server Edition 4.02
- IBM WebSphere Portal Server 4.1.2
- IBM WebSphere Studio Application Developer (WSAD) Integration Edition version 4.1.1
- Portal Toolkit
- IBM HTTP Server 1.3.19.1

For the database, we used WSSAMPLE, which comes with DB2 (optional on install). Figure 2 shows the tables and fields used in the sample application.

Portal Integration Approach

GATHERING AND ANALYZING APPLICATION REQUIREMENTS

The sample application has four requirements:

- To display and filter the list of employees by department
- To view information about an employee
- To edit information about an employee
- To add new employees

MODELING

Modeling is the designing of software applications before coding. This is essential in any project of any size to ensure the user's requirements are addressed, and the business functionality is complete and correct. UML (Unified Modeling Language) is a tool that helps to specify, visualize, and document models. The type and number of models (i.e., use cases, sequence diagrams, class diagrams) used are dependent on the needs and the size of the application to be implemented.

After modeling, our analyst easily scopes out the functions that are needed for the sample application, as well as the function's inputs (arguments) and outputs (results). Table 1 lists the function signatures.

CREATING THE ENTERPRISE REPOSITORY

Here is where we diverge from the traditional design and development of portal solutions. Often a team may go straight from "spec to implementation" or "model to implementation;" we instead create an enterprise repository of business services, published using the Web services standard. The application of Web services implies that UDDI is used for the repository, WSDL for the description of the service interface or signature, and SOAP for invocation.

So, thus far the architect has defined – and received user approval on – the application design, functionality, and interfaces. These interfaces then feed the UDDI and are made into published and available services, albeit with generic "stub" implementations. This becomes the "contract" between those who implement the business services and those applications that consume them. Once this exists, front-end portal developers and back-end programmers may begin the implementation phase, without affecting one another.

We will now detail the steps used to build an Enterprise Repository based on Web services.

CREATING XML SCHEMAS AND SERVICE DEFINITIONS

WSDL (Web Service Description Language) is an XML-based language used to describe any kind of service. Figure 3 shows the WSDL document architecture.

The WSDL document has two main sections. The top section is the abstract service interface definition. This is called a portType and is where you define your input and output operations. The bottom section describes how the service interface is implemented and where you can find it. In practice, you do not need to write WSDL from scratch; there are WSDL tools that generate it for you. However, you are still likely to have to modify the tool-generated WSDL to make it complete, e.g., mapping a message format type to its corresponding type in your implementation.

Furthermore, at present, SOAP providers do not support multiple parts in output messages. Therefore, even if you defined your operation in WSDL to return multiple parts, you will only be able to retrieve the first part in the output message. A simple workaround to return complex messages is to model them using XSD, an XML-based grammar for describing the structure of service messages.

In our sample application, armed with the list of business functionalities defined earlier, we proceed to translate these functionalities into WSDL syntax. Four XML schemas must be created to model employee, list of employees, department, and list of departments accordingly. We also split the service definition into an interface and an implementation binding file. The interface WSDL file, WSDemoServices.wsdl, contains the interface of the service, along with the message and type descriptions. The implementation binding WSDL file, WSDemoServicesJava.wsdl, contains the binding to the service provider along with its location.

WSAD Integration Edition comes with both XML schema and service definition wizards. The wizard-generated interface WSDL file needs to be modified to define the complex types and mapping. Table 1 shows parts of WSDemoServices.wsdl that define XML schemas, set messages to those types, and associate operations to its input and output messages.

GENERATING JAVA SKELETON AND HELPER CLASSES

Java skeletons, as their name implies, are Java functions that have no actual implementation; rather, they are simply a description of an implementation. WSAD Integration Edition comes with a skeleton wizard that generates a Java skeleton class and helper classes from a Java binding WSDL file.

In our sample application, the wizard generates three Java classes:

- WSDemoServices.java (skeleton class)
- Department.java (helper class)
- Employee.java.(helper class)

After running the wizard, we edit the skeleton class to return dummy values as opposed to nulls and empty strings. This provides the service requesters or users with a sample result they can utilize to proceed with their implementation and testing.

DEPLOYING AND PUBLISHING AS WEB SERVICES

A Web service is a component packaged and published in a service registry (UDDI) as a single entity that can be used by other programs. UDDI provides a mechanism for clients to dynamically find other Web services. Using a UDDI interface, businesses can share enterprise services or dynamically connect to services provided by external business partners. There are two main players in Web services:

- ***Service provider:*** Creates and deploys its Web services. It can publish the availability of its services through the UDDI Business Registry and respond to requests to use its services.
- ***Service requester:*** Uses the UDDI Business Registry to find a needed service and bind to that service.

The next step in our approach is to make this service available by deploying the WSDL to WebSphere Application Server as a SOAP service. Using the deployment wizard in WSAD Integration Edition, Figure 4 shows the directory structure after deployment. A SOAP deploy-

ABOUT THE AUTHOR

Lilybeth Tan is a senior consultant specializing in distributed WebSphere architectures and dynamic content management. She is a member of the Prolifics WebSphere Consulting Division, a staff of specialized consultants retained by IBM to solve the toughest of their customers' challenges and deliver training, mentoring, and development services.

E-MAIL
ltan@!prolifics.com

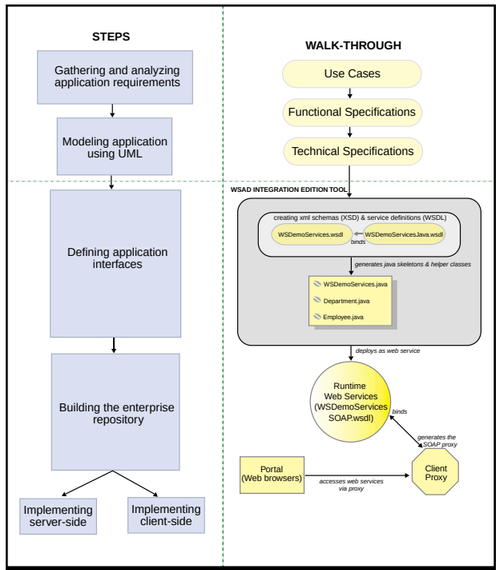


FIG. 1: PORTAL INTEGRATION APPROACH

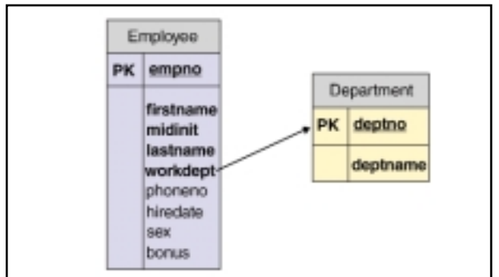


FIG. 2: DATABASE SCHEMA

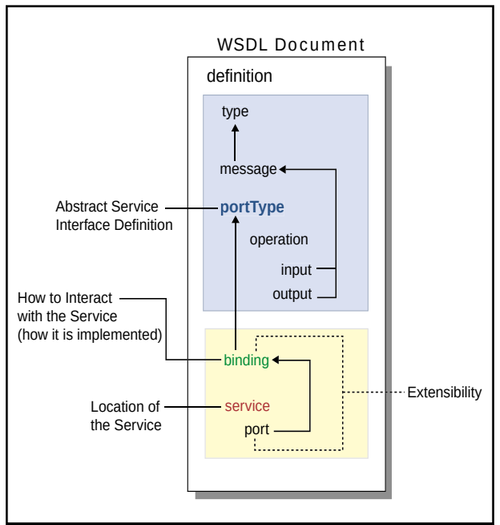


FIG. 3: WSDL DOCUMENT ARCHITECTURE

ment descriptor, WSDemoServicesSOAP.wsdl, was generated in the Web project, WSDemoAppWeb, to inform the SOAP server about the service. It contains servlet initialization and mapping information to run the Web module within an application server. Likewise, the EJB project, WSDemoAppEJB, now contains all the resources for EJB applications, including the session bean WSDemoServicesServiceBean, the remote interface WSDemoServicesService, and the home interface WSDemoServicesServiceHome. An EJB deployment descriptor, WSDemoServicesServiceEJB.wsdl, was also generated to describe the session bean.

CREATING THE SOAP PROXY

A SOAP proxy is used to invoke the SOAP service. The proxy provides a remote procedure call interface to the service. Using the proxy, the application calls a remote method on the service as if the method were a local one. Once the application makes the remote call, the proxy handles all the communication details between the application and the service using SOAP. WSAD Integration Edition comes with a service proxy wizard that generates the SOAP proxy.

Server-Side Development

Now that we have established the repository of method stubs, the application implementation can begin by the front-end portal developers and the back-end programmers. We have achieved a true "subdivision of labor" in which each group focuses on their area of expertise and commits on their deliverable, shielded from the deliverables of other teams.

IMPLEMENTING THE METHODS

Although the methods in the Java skeletons return dummy values, it does not prevent the front-end developers from starting to work on the user interface of the portal. Changes at the server side later on will not require any changes at the client side. Figure 5 shows how the front-end components are hooked to the back-end components through Web services.

Since all of these methods need to query or update the database, we can create a helper class to access the database through a predefined data source. This class acts as a database connection dealer and container of SQL statements so we can keep specific database logic out of business functions and make the implementation more clean and flexible.

By using a data source rather than using a direct JDBC connection, we get access to a pool of connections to the database. It improves application performance and simplifies resource allocation and connection calls. Listing 2 shows the code for getting a database connection from the data source object.

To use a data source, we need to configure the server to recognize it. In WSAD, we add the new data source via the Server Configurations window. For WebSphere Application Server, we add the new data source at the JDBC connection pool in the Administrator's Console. The server looks up a data source object by name from the JNDI (Java Native Directory Interface) server. From a programming perspective, we only need to know the JNDI name of the data source in order to use it. This facilitates application deployment and migration.

As an alternative to writing the database helper class, DataAgent.java, WSAD comes with a wizard that generates database access beans (DB beans) from SQL statements. We use a separate Java class to avoid mixing presentation and database logic in the JSP when using DB Beans.

As soon as the helper class was written, we were able to replace the Java skeleton class, WSDemoServices.java, with actual implementation code. Listing 3 shows the code that implements the getAllDepartments method in DataAgent.java. Listing 4 shows the code in WSDemoServices.java using this method. After implementing and testing the Java skeleton class, the server-side coding is complete. To make these changes transparent to the front end, we only need to republish the enterprise application, WSDemoApp.ear, to WebSphere.

FUNCTION NAME	INPUT PARAMETER(S)	OUTPUT(S)
GetAllDepartments	none	Department[ ]
GetAllEmployees	none	Employee[ ]
getEmployeesIn Department	department_number: String	Employee[ ]
getEmployeeBy Number	employee_number: String	Employee
UpdateEmployee	employee:Employee	Boolean
AddEmployee	employee:Employee	Boolean

TABLE 1: FUNCTION SIGNATURES

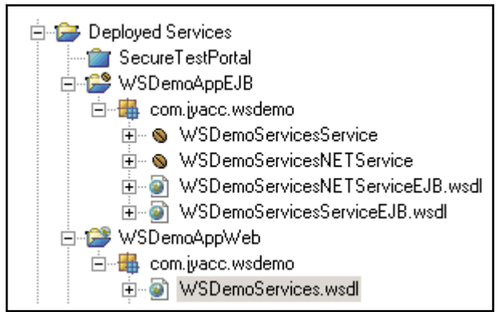


FIG. 4: DIRECTORY STRUCTURE AFTER DEPLOYMENT AS SERVICE

Building the Portal

The power of this approach is making parallel development a reality. While the back-end programmers are putting in the true implementations of the methods, the front-end developers are building the user interface for the portal. The Web services, whether they return dummy or true values, are ready and usable for the front-end developers.

In our portal application, shown in Figure 6, we demonstrate personalization and the use of two portlets that fully interoperate with each other. Using portlet messaging, the selection of an employee's name from the EmployeeList portlet displays detailed contact information for that employee in the EmployeeUpdate portlet, which can then be further updated. We retain the query selection of "department" from the user's last visit as a user preference stored in the PortletData object, which automatically persists in the Portal Server. The next time this user logs in, the EmployeeList portlet will display only employees from the department previously selected. Table 2 provides a brief description of each portlet.

To build the portal, we used WSAD and the comprehensive Portal Toolkit, which IBM provides as a plug-in to WSAD. Our first step was to create the portlet application project. A portlet project is similar to a Web project in a J2EE application. It includes an additional deployment descriptor called portlet.xml in the WEB-INF directory, which defines portlet capabilities to the Portal Server. Creating a portlet project automatically creates one portlet. Note: portlet classes are subclasses of servlet classes, so they appear as servlets in the web.xml file.

The EmployeeList portlet required two JSP files, EmployeeListView.jsp to handle the view mode, and EmployeeEdit.jsp to handle the edit mode. The portlet's doView() method and doEdit() method were set to use these JSPs respectively. Both of these JSPs use the methods from the published Web services to get a list of departments and a list of employees by department. The EmployeeList portlet also implements ActionListener to save user preferences to the PortletData object and to send a message to the EmployeeUpdate portlet notifying it of the selected employee ID.

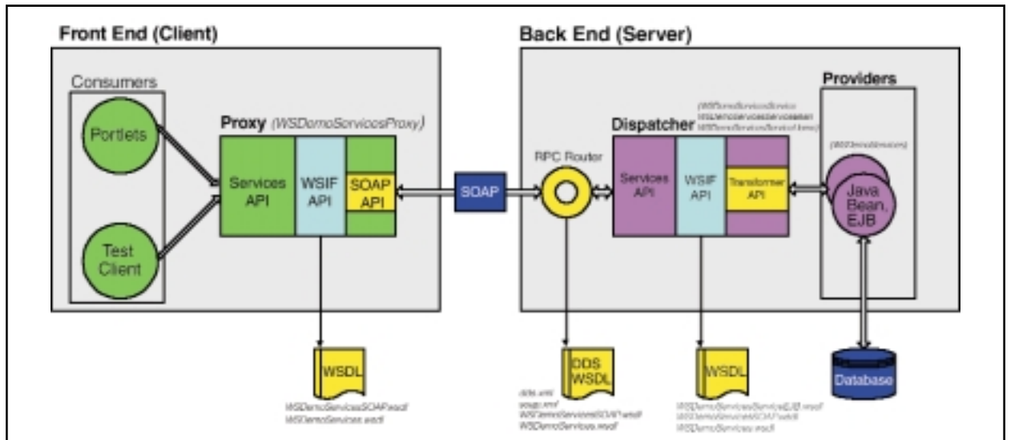


FIG. 5: TESTING ENVIRONMENT FOR DEVELOPMENT WITH WSAD INTEGRATION EDITION

PORTLET NAME	DESCRIPTION	MODE SUPPORTED
EmployeeList	In view mode, it displays the list of employees. In edit mode, it allows the user to filter the list of employees by department.	view, edit
EmployeeUpdate	In view mode, it displays the details of an employee selected from EmployeeList. It allows adding a new employee, and editing the information of an existing employee.	view

TABLE 2: PORTLETS

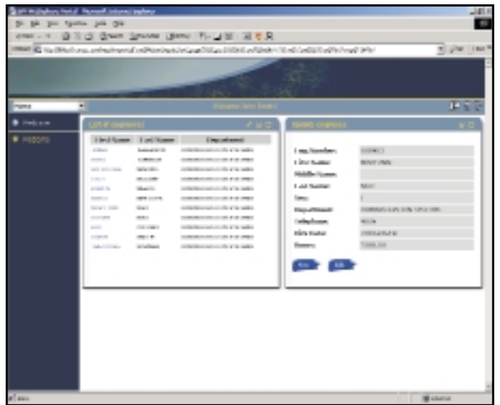


FIG. 6: SAMPLE APPLICATION PORTAL

The EmployeeUpdate portlet lives in the same project so messages can be exchanged between portlets. We set the portlet's doView() method to use EmployeeDetailView.jsp to view and manage the details of an employee. The JSP invokes a method from the published Web service that retrieves employee information. This portlet implements ActionListener to add and update the employee's information, also using methods from the available Web services. It also implements MessageListener to listen for messages from the EmployeeList portlet.

DEPLOYMENT

The final step in any development project is to deploy the application. For a portal solution this involves configuring the deployment descriptors in order to display the portlets as desired, and then building and publishing the final Web pages.

To configure the deployment descriptors, we edited portlet.xml to support both view and edit modes for the EmployeeList portlet, added the EmployeeUpdate portlet to the servlet list in web.xml, added the EmployeeUpdate portlet to the portlet list in portlet.xml, and exported the portlet application to a WAR file.

Next, as administrator, we installed the portlet application (WAR file) to WebSphere Portal Server using the Portal Administration tool. Finally, as shown in Figure 6, we created the Web page, WSDemo, using corporate standards, templates, and logos to display the portlets as desired.

Conclusion

This article discussed a portal integration approach to building portals that is based on Web services. Using Web services you have a truly standardized, open framework enabling corporate IT to adapt to business and market changes. Your repository of key assets now jump-starts any future development initiatives. We demonstrated how this innovative methodology allows for efficient project management and a true "subdivision of labor" that has proven to increase productivity and drive down overall IT and application costs.

Resources

- *XML schema:* www.xml.com/schemas
- *Web services:* www.xml.com/webservices
- *IBM Web services zone:* www.ibm.com/developerworks/webservices
- *IBM WebSphere Studio Zone:* www7b.software.ibm.com/wsdd/zones/studio
- *IBM WebSphere Portal Server zone:* www7b.boulder.ibm.com/wsdd/zones/portal
- *Portlet:* ftp.software.ibm.com/software/webserver/portal/V41PortletDevelopmentGuide.pdf

LISTING 1

```
……
    <types>
        <schema attributeFormDefault="qualified"
             elementFormDefault="qualified"
             targetNamespace="http://wsdemo.jyacc.com/"
             xmlns="http://www.w3.org/2001/XMLSchema"

xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:xsd1="http://wsdemo.jyacc.com/">
             <complexType name="Department">
                 <all>
                     <element name="deptname"
                     nillable="true" type="string"/>
                     <element name="deptno" nillable="true"
                     type="string"/>
                 </all>
             </complexType>
             <complexType name="ArrayOfDepartment">
<complexContent>
                     <restriction base="soapenc:Array">
                         <sequence/>
                         <attribute ref="soapenc:arrayType"
                         wsdl:arrayType="xsd1:Department[]"/>
                     </restriction>
                 </complexContent>
             </complexType>
  ……
         </schema>
    </types>
    <message name="getAllDepartmentsRequest"/>
    <message name="getAllDepartmentsResponse">
        <part name="result" type="tns:ArrayOfDepartment"/>
    </message>
  ……
    <portType name="WSDemoServices">
        <operation name="getAllDepartments">
            <input message="tns:getAllDepartmentsRequest"
            name="getAllDepartmentsRequest"/>
            <output message="tns:getAllDepartmentsResponse"
            name="getAllDepartmentsResponse"/>
        </operation>
  ……
    </portType>
  ……
```

LISTING 2

```
……

protected synchronized static Connection getConnection() {
   try {
            java.util.Properties parms = new
            java.util.Properties();
            parms.setProperty(Context.INITIAL_CONTEXT_FACTORY,

"com.ibm.websphere.naming.WsnInitialContextFactory");
            // Create the Initial Naming Context
            javax.naming.Context ctx = new

            javax.naming.InitialContext(parms);
            // Lookup through the naming service to retrieve a
            DataSource object

javax.sql.DataSource ds =
(javax.sql.DataSource)ctx.lookup(DATA_SOURCE_NAME);

            // Obtain a Connection from the DataSource
            return ds.getConnection();
   } catch (javax.naming.NamingException ne) {
            ne.printStackTrace(System.err);
            return null;
   } catch(java.sql.SQLException se) {
            se.printStackTrace(System.err);
            return null;
   }
}
……
```

LISTING 3

```
……
public Vector getAllDepartments() {
   Vector v = new Vector();

try {
            Connection con = getConnection();
            PreparedStatement pst =
            con.prepareStatement(SELECT_ALL_DEPARTMENTS);
            ResultSet res = pst.executeQuery();

            Department dept = null;
            while(res.next()) {
                     dept = new Department();
                     dept.setDeptno(res.getString(1));
                     dept.setDeptname(res.getString(2));
                     v.addElement(dept);
            }

            res.close();
            pst.close();
            con.close();
   } catch (SQLException sqle) {
            sqle.printStackTrace(System.err);
   }

   return v;

}
……
```

LISTING 4

```
……

public com.jyacc.wsdemo.Department[] getAllDepartments() {
   // user code begin {method_content}
   return (Department[])
   dataAgent.getAllDepartments().toArray();
   // user code end
}
……
```

Revisiting the shopping cart scenario

XML and Relational Data in WebSphere Studio

BY JOAN HAGGARTY AND CHRISTINA LAU

How many applications can you think of that do not involve data in some way? Data is at the core of most businesses since it provides vital information about things like customers, business partners, employees, and inventory. The vast majority of software applications are written for the purposes of data creation, retrieval, exchange, or storage. So we recognize data is everywhere and needs to be managed to make it available when required and easily maintained. However, data exists in various forms: delimited text, structured text such as XML, and relational data, to name a few. The choice of data format may be made based on a number of criteria, including performance and ability to integrate with legacy systems. The challenge is managing different data formats and making transitions between them within an application.

ABOUT THE AUTHOR

Joan Haggarty is a staff software developer at the IBM Toronto Lab. Joan is a lead developer for the XML and data tools in WebSphere Studio Application Developer.

E-MAIL

joan@ca.ibm.com

This article explores the XML and relational data tooling in WebSphere Studio through the construction of a familiar Web application, a shopping cart. WebSphere Studio provides tooling to help developers easily develop applications that require the use of the several forms of data. Perspectives in WebSphere Studio provide combinations of views and options that are most relevant to the developer's task. Examples of WebSphere Studio perspectives include: J2EE, Web, Server, Data, and XML. In the XML perspective, the XML tools provide support for creating XML document structures (XML Schemas and DTDs), for editing XML documents, and for mapping between relational data and XML. In the Data perspective, the relational data tools show database schemas, and allow you to work with them directly within the WebSphere Studio Workbench. These, in combination with the Java and Web tooling, provide complete support for creating a data-intensive Web application like the shopping cart that we will be using as an example. The steps we will illustrate are:

1. Creating an XML Schema to describe an XML document structure for the purchase order
2. Browsing and querying customer and product information stored in a relational database
3. Mapping XML to relational tables so the XML document data can be stored in or retrieved from the database
4. Enabling the DB2 XML Extender to store and retrieve XML data to and from DB2
5. Creating a Web application to process the customer's purchase request
6. Generating JavaBeans from an XML Schema to create an XML stream
7. Decomposing XML to a DB2 database using a Java stored procedure
8. Testing and running the application

Application Scenario and Architecture

We will use a basic shopping cart scenario that everyone should be familiar with. You visit a Web site to buy one or more items. As you browse the site's product catalog, you add the items you wish to purchase to a virtual shopping cart. At the end of your visit you go to a virtual checkout. At this point, the items in your cart are processed, your total cost is calculated, your personal contact information may be collected and updated, and you select a method to pay your bill.

Our scenario begins when the user submits his or her order at checkout. Once the customer has provided all the necessary information, several steps are necessary. The purchase order is received and acknowledged, perhaps via an e-mail to the customer. Next, the purchase order needs to be validated to ensure that payment will be received from the customer's credit card company. Inventory is checked to determine product availability and location. The customer's record is checked to see if there are any outstanding orders that should be combined with the current one. A ship date is scheduled for the order, and the customer is notified of the delivery schedule. Once the order is ready to be shipped, the customer's record is updated and an invoice or receipt is generated. Figure 1 illustrates the flow of this application.

The majority of steps in the flowchart depend on the processing of data. In this application scenario, the data received from the Web page HTML request will be rendered in an XML format. XML will be used in each of the following steps, as it provides a convenient way to integrate and combine different data sources. In the steps in which the relational database is updated, the XML data must be decomposed so it can update different tables in the database. In this article, we will focus specifically on this task.

XML Tools in WebSphere Studio

Let's introduce the XML tools provided in WebSphere Studio before we begin using them to create parts of the shopping cart application. Table 1 shows some of the tools available in WebSphere Studio.

When opening XML-related documents in the WebSphere Studio Workbench the perspective switches to the XML perspective. The XML perspective views include the Navigator view, which shows the structure of the projects in your workbench; the Outline view, the content of which changes depending on which type of editor is open; the Tasks view, which shows any errors or warnings that may have been reported; and the editor view itself. All of the XML editors use the Outline view to display the structure of the document and an editor to work with the document itself. Most of the editors contain both a design view and a source view. The design view provides a graphical user interface for modifying the document structure or content. The source view immediately reflects changes made in the design view and can also be edited directly.

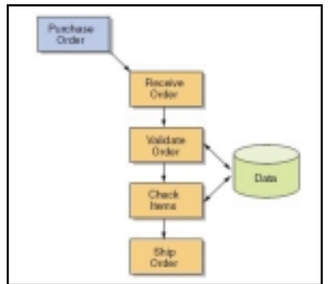


FIG. 1: WORK FLOW FOR A PURCHASE ORDER

Task	Tools
Creating and editing DTDs Creating and editing XML Schemas	DTD Editor XML Schema Editor
Creating and editing XML instance documents	XML Editor
Creating and generating XSL transformations	XSL Editor and XML to XML Mapping Editor
Debugging XSL transformations Defining mapping between XML and relational data	XSL Debugger RDB to XML Mapping Editor
Creating XPath queries Generating JavaBeans from XML Schema or DTD	XPath Wizard Generate JavaBeans wizard

TABLE 1: WEBSPHERE STUDIO TOOLS

Creating the Purchase Order XML Schema

The first task we will demonstrate is to describe the format of the customer purchase order once it is received as a request from the shopping cart Web page. As mentioned, the purchase order will be an XML document. Either an XML Schema or a DTD could be chosen to define the structure of the purchase order. We will choose to use an XML Schema since it has several advantages over a DTD.

Using an XML Schema, data types can be specified for document elements. This allows tighter control of data that is valid for a particular element in an XML instance document. This becomes particularly important when an application will use both relational data and XML data and likely move data between these formats. In addition to specifying data types, XML Schema also has namespace support. Namespaces provide a URI-based method to qualify element and attribute names and therefore uniquely identify them.

Multiple namespaces can be used in an XML Schema. Many applications on the Web will have similar purposes and be identified by the same names, so namespaces have an important role to play. We can also provide a target namespace for the XML Schema so there is a repository for the elements and attributes it defines.

The XML Schema Editor (shown in Figure 2) is used to create the XML Schema. The purchase order contains an identifying key, an order date, the customer name and address information, and the items that the customer ordered. A PurchaseOrderType for the purchase order that contains the shipping information and items is defined. The shipping information contains the customer's address. The Items element is made up of one or more elements of Item type. The Item type has a part number attribute and contains a product name, quantity ordered, price in U.S. dollars, and the date the item will be shipped.

Figure 2 shows the structure of the XML Schema in the Outline view and the Source tab selected in the editor. Notice in the source that each element has a type that is either a simple type such as a string, or a user-defined type such as PurchaseOrderType or Items. As elements and types are selected in the Outline view, they are located and highlighted in the source as shown for the purchaseOrder element. To add to the XML Schema, popup menu items are available in the Outline view.

Based on this XML Schema, an example of an XML instance document for a purchase order might look like Listing 1. The XML editor could be used to edit this instance document within WebSphere Studio if desired (see Figure 3). The XML editor provides powerful features for editing your XML file. The built-in smart editing provides you with constant feedback about the content model of your document and where elements and attributes can be inserted. You can create an XML document that conforms to your grammar (XML Schema or DTD) or create a grammar from an XML instance. We will return to use the XML tools again later, but for now we'll move on to the relational tools side to show how the

ABOUT THE AUTHOR

Christina Lau is a senior technical staff member at the IBM Toronto Lab. Christina is the architect and manager for WebSphere Studio's XML and data tools.

E-MAIL

clau@ca.ibm.com

customer data can be processed now that an XML document exists.

Data Tools in WebSphere Studio

In our scenario a relational database is used to keep a record of our customer and order information. The data tools in Websphere Studio will be useful here. These tools can be used to browse existing databases and to create new tables, views, SQL statements, and stored procedures to query and update the database records. Many database vendors are supported, including DB2, CloudScape, Informix, Oracle, Sybase, Microsoft SQL Server, and MySQL.

The Data perspective in WebSphere Studio houses the relational database tools. The Data perspective allows you to work with your databases either in a connected or disconnected fashion. There are four views of note in addition to the Navigator, Tasks, and Outline views we've already described: the Data Servers view, the Data Definition view, the Output view, and the editor view. The Data Servers view is the way to make connections to your database. A connection wizard is used to make a JDBC connection. Once the connection is made, the corresponding database structure is shown in this view. Using this view, you can browse the database structure and reconnect connections that have become disconnected.

You can use the Data Definitions View to create new databases, schemas, tables, views, SQL statements, and stored procedures. Existing database definitions can also be imported from the database catalog into the Data Definition View for modifications. There are editors for the various objects, such as the Table editor, shown in Figure 4, in which a table named PurchaseOrder is displayed for the CLAU schema.

After creating or modifying database objects, DDL (Data Definition Language) can be generated from database objects. There is an Export to Server wizard that allows you to execute the DDL on the database server to create the database objects. You can create SQL statements with a popup menu option in the Data Definition view and edit them with the SQL Query Builder. Statements can be executed against the database from within the workbench. Wizards and editors are available to help you create stored procedures and user-defined functions. Once the stored procedure or user-defined function is created, you can build, run and debug the stored procedures on the DB2 server.

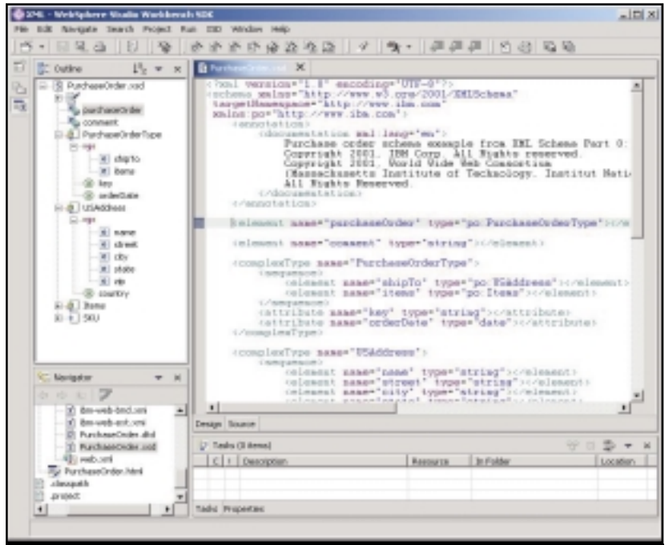


FIG. 2: XML SCHEMA EDITOR

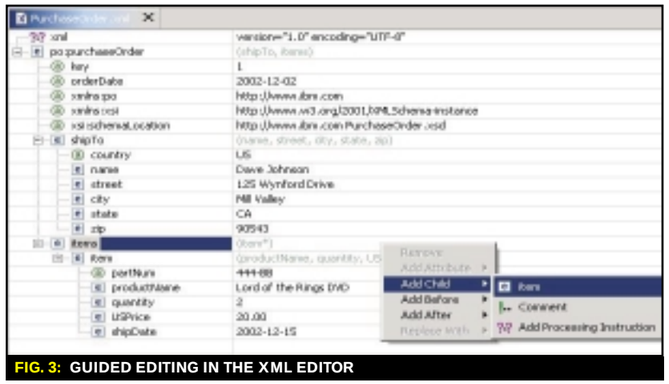


FIG. 3: GUIDED EDITING IN THE XML EDITOR

Mapping XML to Relational Database Tables

Going back to our sample shopping cart application, recall that we have defined the XML Schema that defines the purchase order XML document. The next step is to map the purchase order schema to the relational database tables. We want to be able to store information such as elements and attributes from the XML documents into corresponding corresponding database table columns. In our example, there is a USAddress table keeping track of the customer's address, an Items table keeping track of the list of order items, and a PurchaseOrder table to record the list of purchases made by the customer. The XML document data will be decomposed by the application and stored in these database tables according to a mapping we define.

DB2 provides advanced composition and decomposition functions with the DB2 XML Extender to store

XML data in the DB2 database. The DB2 XML Extender also provides new data types, functions, and stored procedures to manage XML data in DB2. In order to use the DB2 XML Extender a DAD (Document Access Definition) file is needed. A DAD file is an XML file that defines the mapping between the XML document structure and the tables and columns in a DB2 database. It is used by the DB2 XML Extender to compose and decompose XML data.

WebSphere Studio provides a Relational Database to XML Mapping Editor that allows you to visually create the mapping and generate the DAD file. For this example we will map the PurchaseOrder to the Items, PurchaseOrder, and USAddress tables. Figure 5 shows the RDB to XML Mapping editor.

Simply dragging a table column from the Tables pane and dropping it into an element or attribute in the XML pane defines a mapping in the editor. Defined mappings are shown in the Overview at the bottom as they are created. As an example, look at the mapping between the ORDERDATE column in the PurchaseOrder table and the orderDate attribute in the PurchaseOrder XML. Joins can be defined between tables used in the mappings. We will do this for our scenario to relate the list of items for this particular purchase order. Use the Join Conditions wizard to define the following join between the Items and the PurchaseOrder table:

```
ITEMS.ITEMSID = PURCHASE-
ORDER.PURCHASERORDERID
```

Once all of the mapping has been completed, the next step is to generate a DAD file to describe the mapping. Do this using the Generate DAD wizard, which is invoked from a menu option in the RDB to XML Mapping Editor. A few pieces of information related to your DB2 XML Extender and DB2 setup are required, such as the name and path for the generated DAD file. Optionally, you can generate a set of scripts to unit test your DAD file to make sure the mapping is correct.

Enabling the Database to Use DB2 XML Extender

In order to use the DAD file we created to define the mapping from the purchase order XML to the RDB tables, the database needs to be enabled to use the DB2 XML Extender. Use the following DB2 command to enable the database "po" for XML Extender.

```
dxxadm enable_db po
```

In addition, the DAD mapping file we created (PurchaseOrder.dad) must be stored in the database so it is available to the XML Extender to make the mapping. To do this, the following DB2 command is used. (*Note:* The name, po_collection will be used later on in our application to reference the XML Extender collection.)

```
dxxadm enable_collection po
po_collection
PurchaseOrder.dad
```

Creating the Web Application

Now that we are set up with the schemas on both the back end (defined using relational) and the application (defined using XML), as well as the corresponding mapping, it's time to create the Web application that will make use of this infrastructure.

WebSphere Studio provides wizards to help with creating a J2EE Web application. Knowing that the Web application receives a purchase order request via HTTP, we will create a servlet to receive the request, convert the incoming data into XML, make the appropriate updates and queries to the database, and finally respond to the request.

The first step in creating the Web application is to create a Web Project using the Web Project creation wizard. Next, we can use the Servlet wizard to create the servlet that will process the HTML request. We specify a name for the servlet and where to store it, in addition to specifying a superclass, modifiers, and several other options.

Now that we have the generated servlet code, we need to customize it for our application. We'll need a method in the servlet to convert the input from the Web page into an XML stream. The XML stream format will be based on the structure defined in the purchase order schema. Once the XML stream is created, we can pass it to the DB2 XML Extender so it can be decomposed into the database tables using the DAD mapping file we created earlier.

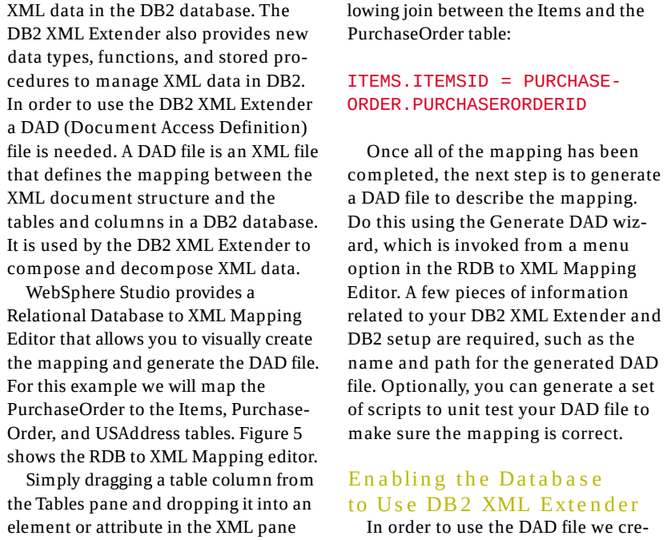


FIG. 4: RELATIONAL DATA PERSPECTIVE

Creating an XML Representation of the PurchaseOrder

An XML stream can be created by using the native Java API for XML Processing (JAXP) APIs to build the DOM tree and then serialize it to a string. Coding to the JAXP APIs can be tedious. WebSphere Studio provides a framework and generator to create JavaBeans from an XML Schema or a DTD. Using the generated JavaBeans with the type-safe APIs can greatly simplify the process of creating and parsing an XML document. For the shopping cart application example we'll use the purchase order XML Schema to generate the JavaBeans. Then we will call the methods in the generated beans to set the appropriate data for the customer address and item information.

FIG. 5: RDB TO XML MAPPING EDITOR

The Generate JavaBeans wizard can be invoked from the popup menu for the purchase order XML Schema file.

The request parameters from the Web page will be used to invoke the setters on the generated JavaBeans. Once all the data is initialized, a getXML() method will return a string representation of the XML stream.

Decomposing the XML Data into the Database

With a string representation of the XML stream, we are ready to invoke the stored procedure db2xml.dxx InsertXML to decompose the XML stream into the corresponding DB2 tables. This will require making a connection to the database, initializing the stored procedure, and invoking the stored procedure. Listing 2 shows the Java code fragment that will perform this task.

Testing the Shopping Cart Application

We have now completed the parts of our shopping cart application that allow purchase orders to be received and the back-end database to be updated. This might be a good time to test the application flow before adding more business logic. To do that, use the Run On Server command to run the application using the integrated WebSphere unit test environment.

Summary

Using a typical Web shopping cart scenario as an example, this article has shown how to create a Web application using WebSphere Studio. We followed the application flow, which takes data input from an HTML page, converts it into XML, and decomposes that into a DB2 XML-enabled database. You have seen how the WebSphere Studio wizards and editors make it easy to manipulate both XML and relational data in Web applications. The power of the generators in WebSphere Studio has been demonstrated here with the generation of JavaBeans for creating an XML stream from a Web form, and the generation of a DAD file to map from XML data to relational data. Finally, the integrated WebSphere unit test environment makes it easy to test and debug your Web application.

Resources

- The complete application is available for download at www.sys-con.com/websphere/sourcec.cfm
- WebSphere Studio Application Developer: www-3.ibm.com/software/ad/studioappdev/about/V5.html
- DB2 XML Extender: www-3.ibm.com/software/data/db2/extenders/xmlext/downloads.html
- XML and WebSphere Studio series on WebSphere Developer Domain: www7b.boulder.ibm.com/wsdd/techjournal/0111_lau/lau.html

LISTING 1: PURCHASE ORDER EXAMPLE

```
    <purchaseOrder key="1" orderDate="2002-12-02">
    <shipTo country="US">
        <name>Dave Johnson</name>
        <street>125 Wynford Drive</street>
        <city>Mill Valley</city>
        <state>CA</state>
        <zip>90543</zip>
    </shipTo>
    <items>
        <item partNum="444-BB">
            <productName>Lord of the Rings DVD</productName>
            <quantity>2</quantity>
            <USPrice>20.00</USPrice>
            <shipDate>2002-12-15</shipDate>
        </item>
    </items>
</purchaseOrder>
```

LISTING 2: CALLING DB2 STORED PROCEDURE TO DECOMPOSE XML

```
// Declare some variables
CallableStatement cs;
int errCode = 0;
String msgText = "";

// Connect to the PO database

Class driverClass =

Class.forName("COM.ibm.db2.jdbc.app.DB2Driver");
Connection conn =
java.sql.DriverManager.getConnection("jdbc:db2:PO",
"db2admin", "db2admin");

// Prepare the stored procedure call
String sql = "Call db2xml.dxxInsertXML(? ? ? ?) ";
cs = conn.prepareCall(sql);

// Indicate which XML collection to use
cs.setString(1, "po_collection");

// Set the XML string
cs.setString(2, xml);

// Initialize the return parameters
cs.registerOutParameter(3, Types.INTEGER);
cs.setInt(3, errCode);
cs.registerOutParameter(4, Types.VARCHAR);
cs.setString(4, msgText);

// Call the stored procedure
conn.setAutoCommit(true);
cs.execute();
```

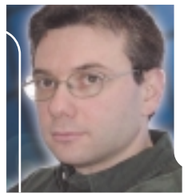

DATA MANAGEMENT

It's all about TCO

Web Services in a Pervasive Computing Environment

BY RON BEN-NATAN

Pervasive computing is taking the world by storm. Industry analysts are predicting that mobile is the "next paradigm shift," and vendors (including IBM) are investing heavily in producing the best toolset for building applications for pervasive computing.



ABOUT THE AUTHOR

Ron Ben-Natan, CTO of ViryaNet, Inc., holds a PhD in computer science in the field of distributed computing and has been architecting and developing distributed applications for more than 15 years. Ron's hobby is writing about how technology is used to solve real problems; he has authored numerous books, including *IBM WebSphere Application Server: The Complete Reference* (Osborne/McGraw-Hill).

E-MAIL

rbennata@hotmail.com

Pervasive devices like mobile phones, PDAs, and pagers far outnumber computers, and the trend is accelerating. Add to that the fact that all sorts of equipment, such as air conditioning units, oil tanks, car surveillance systems, and so on, are being built with serious computing power and a communication link, and you begin to understand the large investment in new computing environments for building applications on devices other than PCs and workstations.

Mobile and pervasive applications aren't new. The explosion of pervasive technology isn't a result of a new business need but rather a result of maturing technology. Users have always been mobile. Because of lack of technology (or lack of reasonably priced technology), mobile users have had to make do with manual processes and paper-based data management.

This isn't to say that there are no mobile systems out there – there are industries in which mobile systems have been a way of life for more than 10 years. One of the best examples is the utilities vertical – companies that are responsible for the distribution of power, such as electric utilities and gas utilities. These companies have been using pervasive computing for many years, including mobile terminals used by field personnel for receiving and completing work orders as well as remote telemetry units (RTUs) that monitor the energy distribution network for faults and provide a constant data feed to the central monitoring stations.

Let's look at an example from the world of field service. In this world, technicians and engineers are deployed in the field and make use of mobile data terminals to receive work notifications. They then use these terminals to see what work needs to be done where and receive additional information such as circuit diagrams, product catalogs, and so on. Traditionally, most of these data terminals have been rugged laptops and/or alphanumeric pagers. Client software running on such a laptop is shown in Figure 1. With the proliferation of lower-cost, high-quality pervasive devices, the field force can now be equipped with a myriad of devices.

These mobile data terminals access the central server using wireline and wireless networks. This is another area in which technology has made much progress. Five years ago a company would almost always have to set up a private radio network in order to deploy such a system. Today, the availability of lower-cost public data networks makes the use of private radio frequencies (RF) a legacy as companies move away from expensive infrastructure. Some geographies fare better than others; for example, carriers in Europe have deployed almost ubiquitous access over GSM and are now rolling out GPRS. The U.S. lags far behind with a combination of different technologies with spotty coverage, causing a reality in which a national deployment will probably need to use a combination of networks such as CDPD, Mobitex, cellular, and possibly private RF and even satellite. In this reality, the system architecture is complex and the total cost of ownership (TCO) high.

Total Cost of Ownership

TCO is what it's all about (at least from a business perspective). The goal of any organization that wants to deploy a mobile data system has to do with achieving the efficiencies and ROI provided by the application while maintaining a low TCO. If you look at a rollout of a mobile data solution you see that there are at least four important categories to consider when assessing TCO:

- Devices
- Networks
- Applications
- Mobile infrastructure

In each of these categories, technology is making rapid progress in reducing the TCO. In terms of devices, operating systems like Win32, Pocket PC, Palm OS, EPOC, and more allow a variety of data terminals with very good tools and applications. In terms of networks, gone are the days of expensive private RF networks. In terms of applications, environments such as J2EE and J2ME are now being applied, allowing highly maintainable and low-cost applications. But in this article I focus on the mobile infrastructure, specifically the use of Web services within these environments and on the components provided by IBM within the WebSphere family of products that can help you roll out a complete mobile infrastructure.

Web Services in the Field

You're probably a bit surprised that Web services can be used in mobile environments, and I don't blame you. After all, Web services are, by definition, closely related to the Web and the Internet, which are all about online connections. Yet mobile environments must function flawlessly in a disconnected or offline mode. This is because no matter how good coverage becomes, it's still spotty. This is especially true if you live and work in the U.S. Unless you live in the middle of a highly populated metropolitan area, you probably frown (as I do) on the "Can you hear me now? Good" commercial.

Even if coverage is perfect, it will not be complete because some people need to work while in the basement, within radio-free areas (like hospitals), etc. Also, wireless networks today are slow. 3G may solve these issues, but right now you really don't want to wait for an end-to-end transaction all the way back to the server with every button press. What you really need is an environment with a store-and-forward capability, which will allow you to do your work, cache all your transaction submissions, and send them up when the network allows.

Why Web services? The reason goes back to TCO. In the same way that the devices and networks have migrated away from proprietary environments, the messaging platforms and formats are converging. From a data format perspective, it's attractive to use SOAP messages and envelopes to carry information that needs to be sent from the client to the server or from the server to the client. By sticking with a standard, which is the claim to fame of Web services, companies can take another step toward lowering the TCO. And while Web services are mostly used in online environments today, they were designed to work in asynchronous environments, so they're actually quite appropriate when it comes to store-and-forward environments.

WebSphere Components for Mobile Infrastructure

Are we there yet? Almost. Most mobile environments today don't make use of Web services, perhaps because they're just too new. But the WebSphere family of products actually has all the components you need to develop and deploy a mobile data solution using Web services. It's based on a combination of the WebSphere Application Server, WebSphere Everyplace Wireless Gateway, and MQSeries Everyplace.

The IBM WebSphere Everyplace Server is a software platform for extending the reach of new and existing applications to the mobile e-business space. The components include:

- IBM Everyplace Wireless Gateway
- Tivoli SecureWay Policy Director
- IBM WebSphere Edge Server
- IBM WebSphere Transcoding Publisher
- IBM MQSeries Everyplace
- Location-based Services
- Intelligent Notification Services
- IBM Everyplace Synchronization Manager
- Tivoli Personalized Services Manager
- Everyplace Suite Manager
- IBM SecureWay Directory

The beauty of the Wireless Gateway is that it makes any network, or combination of networks, look like a simple IP network (a virtual LAN/WAN, if you will). It supports a variety of wireless and dial-up network technologies. All data traffic that flows through the Wireless Gateway uses a mobile

network connection (MNC), a resource assigned to a Wireless Gateway that defines a specific type of network connection. Existing applications that use a TCP/IP interface may use either wireless or wireline networks – it's completely transparent. In fact, a data transmission can start over one network, lose coverage, and be completed over another network, all without the application even knowing about it. Using the Wireless Gateway shields network-specific details from the user application and provides network-specific enhancements, such as data compression, data encryption, data optimization, and authentication.

Supported IP-based and non–IP-based networks include:

- Packet networks
  - -CDPD
  - -DataTAC
  - -Mobitex
- GPRS
- Cellular networks
  - -AMPS
  - -CDMA
  - -GSM
  - -iDEN
  - -PCS 1900
  - -PDC
  - -PHS
  - -SMS
- Private RF networks
  - -Dataradio
  - -Motorola Private
- Dial-up telephone
  - -PSTN
- WAP

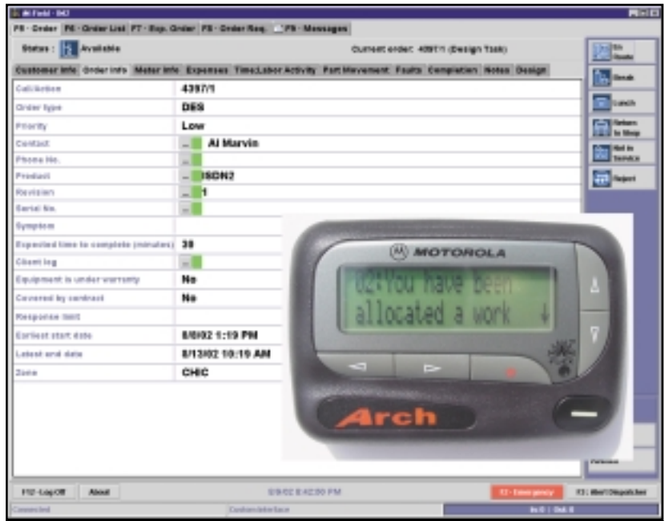


FIG. 1: LAPTOP AND PAGER AS MOBILE DATA TERMINALS

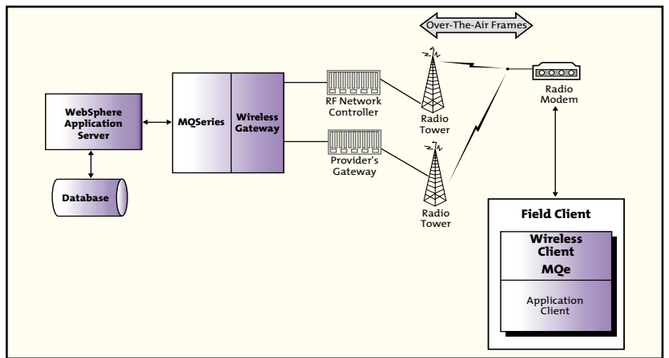


FIG. 2: WEBSPHERE-BASED MOBILE INFRASTRUCTURE

The Wireless Gateway removes the complexity of the underlying network(s) and makes everything as simple as opening a TCP/IP socket and moving data around. However, for the store-and-forward capabilities, you need MQSeries Everyplace (MQe). MQe integrates many of the functions available in MQSeries and supports both reliable and unreliable communications as well as:

- Industrial-strength messaging
- Reliable communications
- Assured, once-only delivery of messages
- Powerful encryption
- Optimized data streams
- Immediate server or mainframe interaction when a link is available (queues messages when it is not)

MQSeries Everyplace is supported on the following platforms:

- EPOC
- Palm OS
- Pocket PC (Windows CE)
- Win32

OK, enough with the marketing fluff, let's put this all together (see Figure 2). The application typically has two components: a server that is running within a WebSphere application server and a client that may be running over pJava, J2ME, or even a non-Java application making use of the MQe C DLLs. These application components communicate by sending SOAP messages to Web service endpoints. The actual SOAP payload is delivered using a set of MQe queues (for example, the MA0R SupportPac for MQ can be used). The delivery of messages from one queue to another is made easy by using the Wireless Gateway. No matter which network is needed, it all looks like TCP/IP when used by MQ.

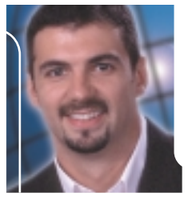
So, go out there, download ~1GB of WebSphere "stuff," and have a ball. And if you need to justify all this great technology, remember: it's all about the total cost of ownership.

CASE STUDY

WEA Trust Relies on Fenway

Solving Performance Issues with E-Business Management

BY FRANK MORENO



ABOUT THE AUTHOR

Frank Moreno is the product marketing manager for Dirig Software, a leading developer of award-winning enterprise performance management solutions. Frank has over 10 years of experience in product marketing, product management, and strategic alliances in the networking and software industries, and has written multiple articles on e-Business performance management.

E-MAIL
fmoreno@dirig.com

Traditional network, system, and application management products deliver incredible amounts of information about the status of a company's infrastructure. However, if everything is up and running and a visitor still can't purchase products or access information (for example, personnel, financial, or health data) from the Web site, these tools have failed to guarantee a positive end-user experience. The evolution of the Internet, specifically e-business, requires even more scrutiny and detail about application performance. To truly ensure the complete performance and functionality of e-business solutions, new management products are needed to manage the individual components and connection points within an application server.

A new breed of management tools that oversee the individual components of an application server – the heart of the e-business infrastructure – has taken precedence in the market. Application servers provide a platform to develop "Web applications" that connect the front-end Web site to a back-end database through a series of smaller software objects called components.

Components, in their simplest form, are small pieces of software that operate within an application server. Each component is responsible for a very explicit task, such as verifying credit card transactions, checking to see if credit card history is valid, confirming inventory levels, delivering customized content to an end user, or allocating connection pool utilization. If any one component fails, the transaction will not successfully complete. E-business management solutions enable IT managers to unleash the full power of an application server, helping to provide maximum uptime combined with instant notification of a transaction failure and identifying the root cause of an e-solutions performance problem.

One organization that has invested in technology to ensure their Web site meets the expectations of their client base is WEA Trust. Having problems bringing his WebSphere-based test e-business infrastructure into production, Jim Struve, assistant manger of Information Support Services at WEA Trust, engaged the services of IBM Global Services (IGS) and Dirig Software's e-business management product, Fenway, to pinpoint and solve the performance issues.

Who, or What, Is WEA?

Headquartered in Madison, Wisconsin, WEA Trust is a not-for-profit organization dedicated to building financial security for Wisconsin's public school employees. Created in 1970 by the Wisconsin Education Association Council, the statewide educators union, WEA Trust provides insurance and retirement investment services to Wisconsin public school employees and their families. A Board of Trustees – all public school employees – oversees the Trust operations to ensure that members get high quality and great value in insurance and financial services. Over 80,000 members use services from WEA Trust.

The WEA Trust Web site is utilized by its members to review policy information, refill prescriptions, and enroll in new services. It is the main method of communication between the Trust and its members. Jim Struve and his IT team needed to ensure that each transaction and information request was completed successfully and decided to implement several additional technology pieces to raise the performance levels of the e-business infrastructure.

Based on stringent critical criteria, WEA Trust's Information Systems group selected IBM's WebSphere platform as the backbone of the Web site that serves their members. They chose WebSphere application servers because they met strict criteria, including flexibility, integration, simplification, and scalability. For example, WebSphere Application Servers run on a variety of operating platforms, providing WEA Trust with flexibility to use Unix, OS/400, Linux, or other server operating systems in the future. WebSphere also integrates well with the existing Lotus Domino environment, DB2 and iSeries infrastructure. In short, the addition of WebSphere was simplified because all the parts integrated well with the existing infrastructure. The WebSphere/ Domino environment provides the opportunity WEA Trust requires today and the means to execute its e-business strategy in the future.

The Problem

WEA Trust was experiencing two main problems in preparing their new infrastructure for production – inconsistent WebSphere availability in the new three-tier environment and the performance problems with specific Web applications within that environment. (The three-tier environment consists of Web servers, WebSphere Application Servers, and legacy databases.) To help WEA address these immediate, critical issues, IGS and Dirig were called upon for a joint on-site engagement to help identify and resolve WEA Trust's unknown performance issues.

IGS initially estimated it would take three weeks to examine the three-tier Web infrastructure to determine the problem. After consulting with WEA Trust staff members, who needed a faster solution, they suggested implementing Fenway, a partner's solution that would reduce the time frame to one week or less. Greatly improving on this time estimate, within less than one day of being installed Fenway pinpointed several problems within the three-tier Web infrastructure that were causing the performance issues (see "The Solution" below for in-depth details).

What Is Fenway and How Does It Work?

Dirig Software provides proactive e-business performance management solutions for production environments that help organizations protect revenue streams, resulting in increased profits, brand loyalty, and a better user experience. Dirig's Fenway Management Extensions provide application and component-level management for managing both J2EE and .NET Web application servers, as well as Java Management eXtention (JMX)– enabled Java applications.

The Fenway management solution gives users the ability to monitor and proactively manage every Web application function, connection point, and component to validate and ensure the integrity of each transaction request, execution, and completion. Fenway manages the entire e-business environment from the front-end Web server, through the application server and middleware applications, to the backend database. With Fenway, Dirig delivers e-business management solutions that both protect and help generate corporate revenue. By employing Fenway, companies can increase transaction success and improve end-user experiences, resulting in a positive impact on revenue and market share, as well as customer and brand loyalty. Fenway can have a direct impact on an organization's bottom line. From small to medium-size organizations to large enterprises, a simple 1% increase in successful transactions can mean a significant increase in profits.

The Solution

IGS and Dirig installed Fenway agents on WEA Trust's WebSphere servers. A synthetic protocol transaction was configured to test a designated component for measuring availability and response time within the Web environment. The test ran for one afternoon and overnight to get a baseline on the three-tier Web infrastructure.

One glaring statistic on both WebSphere Application Servers was the "Connection Pool Percent Used" value for two separate applications. (Connection pools consist of a series of dedicated connections through which the

WebSphere Application Servers communicate to the back-end DB2 databases. Connection pools are allocated to application transaction components as necessary.) The connection pool statistics appeared to be abnormally high, between 50% and 100% continuously for the actual number of user sessions. In fact, there were only 1–5 user sessions actually occurring, which meant that the "Connection Pool Percent Used" value should have been around 0%. Web sessions were failing because the excessive connection pool utilization (i.e., the lack of a connection for an application to connect to the back-end legacy database) prevented information from the DB2 databases from flowing to the WebSphere applications.

Once the session availability issue was identified as the problem, IBM technical support was contacted for potential patches and software updates available to resolve the connection issue. WebSphere and DB2 patches were downloaded and installed and the connection pool issue resolved, increasing WEA Trust member Web site availability and performance.

Within the next two days, the IGS and Dirig team performed additional inquiries on WEA Trust's environment as well as configuring Fenway with a variety of management policies and thresholds. IT managers set thresholds to monitor the usage amount of a certain aspect within the infrastructure, such as memory, CPU utilization, disk space, etc. Going forward, if the usage amount surpasses the set threshold, IT is notified that there is a problem within the infrastructure and in many cases an automated corrective action fixes the issues without human interaction.

The IGS and Dirig teams analyzed alerts and reports from different test settings to ensure that additional configurations would not pose performance issues. For any configuration, the team developed additional thresholds, policies, and corrective actions to head off future problems that WEA Trust may incur.

The Aftermath

In evaluating the WebSphere infrastructure being tested for production, Fenway was able to identify that connection pool statistics were excessive. Fenway also recognized that the connections were not being released back to the pool and that the memory on both WebSphere Application Servers was running near capacity. With the detailed information on what components were causing problems within the infrastructure, WEA was able to change the memory partitions on the WebSphere Application Servers to achieve greater performance. In addition, the software patches from IBM's technical support helped to solve any remaining performance issues. The IGS and Dirig teams helped WEA Trust accelerate the implementation of its new Web architecture, improving their overall e-business initiatives and ensuring the Web site remained available 24x7.

Future Plans

WEA Trust was impressed with the comprehensive management capabilities of Fenway, and Jim Struve decided to implement it on a permanent basis. WEA Trust now uses Fenway to constantly monitor the company's Web infrastructure and member Web site. Jim and his team have also recognized that Fenway can contribute as a capacity planning tool to help manage the equipment resources available. WEA will be able to effectively predict growth and activity on the Web site using Fenway. In addition, through monitoring resource use, WEA Trust will have the ability to efficiently plan and budget for additional computing components needed as the system expands to meet increased Web activity.

One additional problem WEA Trust plans to manage with Fenway relates to the use of JVM (Java Virtual Machine) technology within their infrastructure. WEA has experienced occasional problems with applications running in the JVMs. Several applications can be running in a JVM, but if one of them fails, it is very likely that the failure will go unnoticed by the IT team. A user's entire Web experience can be affected by that failed application. Fenway monitors the applications running in the JVMs for any problems and alerts the IT team to any issues and their locations within the infrastructure.

In the future, WEA Trust will use the reporting tools within Fenway to run various reports to track the site's performance levels. In addition to monitoring the historical data, WEA Trust will use the system to diagnose small problems. As many of these small problems could quickly escalate into larger issues affecting Web site performance, Fenway will alert WEA Trust to the nonevident crisis that is building. Through managing the small details and implementing a quick solution, they will avoid any costly downtime and Web site outages.

Conclusion

Through a detailed services engagement with IBM Global Services and IBM alliance partner Dirig Software, WEA Trust was able to pinpoint the small details within its Web infrastructure that were causing numerous performance issues. With Fenway installed, WEA can now proactively manage and monitor its Web infrastructure to ensure that members are able to access the site and the information that's important to them, and complete all of their transactions successfully.

Many businesses are relying upon Web sites to communicate and support their customers and partners, sell goods, and relay their company's market positioning. When a Web site fails and customers cannot access what they're looking for, they will turn to a competitor, and revenue will be lost. An ineffective Web site impacts the bottom line, but equally important, it can tarnish the company's image overnight. As demonstrated in WEA Trust's case, the use of e-business management solutions using application component technology helps companies of all sizes maximize their Web site performance and, in the end, increase revenue and customer retention.

VIEWING CORPORATE APPLICATION DEVELOPMENT AS A PRODUCT DEVELOPMENT LIFE CYCLE

BY ROBERT COSTELLO

Meeting the new challenges

ABOUT THE AUTHOR

Bob Costello works for MS2 Corporation, provider of a product life cycle automation solution. Bob has over 15 years as an application architect.

E-MAIL

rcostel@comcast.net

Managing the software application development life cycle is more challenging than ever. With the introduction of e-business technologies such as IBM's WebSphere solutions architecture, organizations face inherent complexities. Many more issues must be considered before launching any type of e-business application. In most cases, cross-functional teams need to be formed so that there is representation from each of the organizations participating (see Figure 1).

In most large organizations, application development is usually done by one or more organizations focused only on that task. When a solution is required, the business owners contact that organization and the project begins. However, if it is across functional lines, which is the case with e-business projects, the business owners provide the external funding. With this scenario, the internal application development group morphs into not just an internal IS group, but also a product development group that executes a product development life cycle.

Figure 2 is an example of a product development life cycle for a typical e-commerce application offering. It is fairly complex, as are most e-business projects. Listed below are some of the complexities not documented in the diagram:

- Constantly changing environment
- Escalating competitive climates
- Increasingly sophisticated user requirements
- Budget constraints
- Management of development processes across multiple organizations
- Working with dispersed geographic teams while simultaneously interacting with contractors and customers
- Many late-to-market projects and cost overruns
- No central repository to store project information, loss of valuable knowledge when personnel leave the project
- Poor documentation or no standard documentation
- Inability to share best practices, creating silos of information

Most companies have spent millions on software development processes that are supposed to help solve many of the problems stated above. However, just putting them into a binder and distributing them isn't the answer. What is needed is an application that can automate the life cycle – enforcing the processes, collecting the metrics to see if they're successful, and enhancing communication between the teams.

This article defines the key criteria to look for when assessing a new solution for automation of the software development life cycle. The four issues that are most critical to successfully bringing applications to release are:

1. Assisting development organizations with defining and executing a consistent life cycle process
2. Giving team members quick and easy access to all life cycle information
3. Providing real-time visibility into the applications portfolio, to make investments decisions and determine the status of all programs
4. Bringing the extended development team, including partners and business users, into close communication with the core development team

An effective automation solution must help the organization accomplish its objectives while minimizing the impact on the development team members. MS2 Accelerate is being used by application development organizations to help them address these challenges and deliver better applications faster.

Application Development Practices

It's a widely accepted fact that accessing information and the applications that manage it is strategic to an organization's ability to maintain its competitiveness in the marketplace. Managing a portfolio of applications through the development life cycle and delivering the right applications at the right time are key to driving a company's revenue and growing its market share. Organizations that can effectively plan and execute their application delivery strategy and deliver the right applications faster and cost effectively have a marked advantage in today's competitive business environment. Yet, in spite of its importance, application development management remains a very complex endeavor with limited tools and technologies to manage it.

In a recent survey of 140 high-technology executives, 99% agreed that it is hard or very hard to repeatedly and predictably deliver applications on time. The challenges of effective application life cycle management are exemplified by the following:

- Nearly 75% of all development programs miss their target release date or never ship at all
- More than 50% of all development programs take twice as long as expected to complete, or are canceled altogether.
- On average, top performing application development organizations hold a tenfold productivity advantage over weaker performers in the same industry
- At least 7 of the 10 most common signs of application development failure are present before the design is created or a single line of code is written

Fewer than 5% of all application development programs have successfully identified the complete set of customer requirements before beginning application design.

Figure 3 illustrates this point. The earlier in the life cycle you can identify problems and propose a solution, the easier it is to correct them. As simple as this may seem, most organizations fail in this, many times because the team lacks visibility into the overall solutions.

Complex Challenges Facing Application Development

The application development life cycle has evolved from the traditional waterfall process into a highly flexible one requiring multiple decision check points and input from not only the engineering teams, but often multiple business teams, often located on separate continents. These projects are usually under severe time constraints, requiring teams to execute quickly on a common objective. Even though most development activities are interdependent, the time-compressed environments that software projects exist in dictate that they must occur in parallel in order to meet the requirement window. The complexity of applications being developed and the need to leverage core competencies and advances in software technology, such as Web services and component architectures, requires that many of these projects be conducted in conjunction with joint-development partners. As the number of applications under development and the number of partners involved in the process increase, however, the resulting complexity is overwhelming.

CHANGE IS CONSTANT

The only constant in software development projects is change. Managing the application development life cycle would be difficult even if all requirements were known up front and never changed during the development process. But the reality of today's competitive business

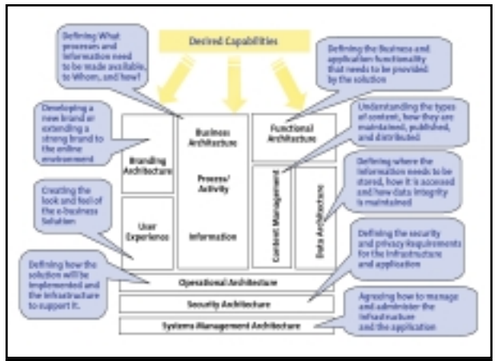


FIG. 1: ORGANIZATIONS INVOLVED IN AN E-BUSINESS APPLICATION

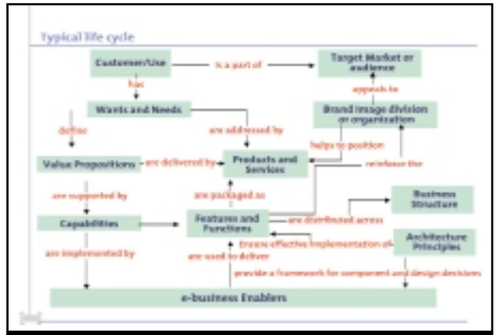


FIG. 2: PRODUCT DEVELOPMENT LIFE CYCLE EXAMPLE

environment guarantees that there will often be significant changes in customer requirements and available technologies, not to mention employee turnover on the team. The ability to anticipate and effectively respond to changes in the market separates the winners from the losers.

TIME TO MARKET IS CRITICAL

As the competitive environment intensifies, the time between new application releases continues to shrink. The speed at which technology changes also affects how quickly applications must be delivered. The window of opportunity between an application's feasibility and its ultimate obsolescence is narrowing. And because being late could be disastrous, extreme pressure is placed on application development teams to move quickly. To survive, organizations absolutely must develop the right application and deliver it at the right time.

BUSINESS USERS ARE MORE SOPHISTICATED

With the development of Java applications, object-oriented programming, and browser-based applications, business users demand an application that can be easily adapted to their specific needs. Therefore, the creation of cross-functional business teams and the ability for them to communicate is critical to getting business validation of the application plan before the development process begins, as is continuously monitoring business feedback throughout the development process.

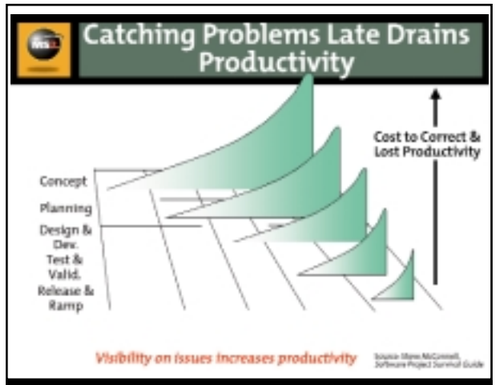


FIG. 3: THE IMPORTANCE OF IDENTIFYING PROBLEMS EARLY

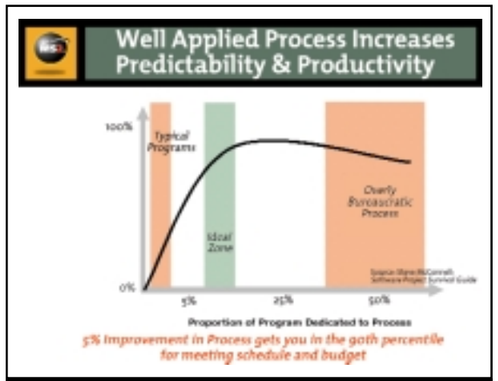


FIG. 4: WELL-APPLIED PROCESS INCREASES PREDICTABILITY AND PRODUCTIVITY

RESOURCES ARE SCARCE

The most valuable resources in an organization are always scarce. Finding, training, and retaining application team members with highly specialized skill sets are continual challenges. It's often difficult to be sure that the most strategic development programs have been allocated sufficient resources for success. The allocation of these scarce resources is at the core of an effective application portfolio investment strategy.

Assessing an Organization's Current Approach

Executive managers can enlighten themselves by asking the following questions about their own application organization and development process:

- Are you making the right investments in your application portfolio to yield the greatest return?
- Do you have a well-defined process to consistently and predictably deliver applications on time? If so:
  - –Are your teams using it?
  - –How do people in your organization learn the process?
  - –Can you ensure that cross-functional application development groups are working together as a team, based on a common view of what the business wants and what should be built?
  - –Do you have real-time visibility into the detailed status of all applications under development?
  - –Are you communicating quickly and effectively with your extended development team, including contractors, joint development partners, and business users?

If you answered no to some of these questions, you're not alone. Application development organizations have implemented many different tools to manage some of these complexities. Spreadsheets, program management software, source-control systems, and even home-built Web sites are being used in an attempt to assess a company's ability to design and build the right new applications. While these tools are all useful, they often create islands of information that quickly become out-of-date or obsolete, making it difficult for the team to execute on an integrated application strategy. If you can automate the processes and create process improvement, the payback is enormous. (see Figure 4)

The Need for a Solution

An automated solution is required to provide the information application development organizations need to answer these questions confidently. It's important to consider what capabilities and features are needed in such a solution. A sound set of assessment criteria will ensure the best selection.

Criteria for Automating the Application Life Cycle

The increasing challenge of defining and developing winning applications on time has led to the need for an application to automate the application development life cycle. A viable Product Life Cycle Automation (PLA) solution must meet the criteria explained below.

A Successful, Repeatable Application Life Cycle Process

An application organization typically has an application development life cycle that it would like to follow for every program. The life cycle process is often long, detailed, difficult to execute, and even harder to repeat every time across multiple programs. Even when the entire process is fully documented, the plans may remain in a binder on someone's desk, where they're often forgotten or skipped. Many of the improvements or lessons learned never get repeated because there isn't consistent use of the defined process. A study of a dozen companies found that a viable solution for automating the application development life cycle must meet the following criteria:

- Enable application teams to capture and implement a repeatable process for use on all development projects
- Create and maintain a single, well-organized, and easy-to-access location to manage the application life cycle process and all deliverables
- Provide a live application roadmap with visibility into the application portfolio and status of development projects
- Enable timely and clear communication with the extended development team

Application development organizations spent less time on administrative activities and more time focused on application development. The result: business units that embraced this approach reduced their average development times by 30–50%.

The first step toward ensuring a consistent and effective new application life cycle is to create a process. A PLA solution must allow the application development organization to capture its best practices in the form of processes and templates. Ideally, it should provide out-of-the-box best practices for organizations that haven't had time to create them. These processes and templates should be embedded into the work environment and the tools that team members use, thus avoiding their binder problem. Each new program should be able to leverage these best practices without requiring team members to dig up documentation on the process. Finally, successful execution of the process means that every individual on the team knows his or her role in pursuing the goal: What deliverables am I responsible for? When do I create them? Who must review and approve them?

Single Source for All Application Life Cycle Information

The complexity of development programs and the interdependencies of each team member's work require a common framework for interaction. This interaction, however, can consume an inordinate amount of time and energy.

A recent study found that engineering resources, on average, spend 70–75% of their time attending meetings, correcting mistakes, and traveling, leaving only 25–30% of their time for value-adding activities such as designing, developing, testing and solving problems.

Project managers often spend the majority of their time creating "executive dashboards" for weekly review meetings, in an attempt to give project visibility to executives. On large projects this may involve the consolidation of multiple-project plan information and numerous phone calls and e-mails. A pattern begins to emerge; by the time the project manager has finished the review materials, the cycle must begin again, never allowing the project manager to truly manage the project.

Application development organizations have tried the following alternatives to address this problem:

SHARED DRIVES OR FOLDERS

Many organizations use shared drives or folders to allow team members to share documents on a given development program. Because the structure is generally ad hoc, however, most find that it is still cumbersome for team members to locate the right deliverables. Even when they can find deliverables, it's often unclear whether they have the latest version and whether it's a draft, a review copy, or the final approved version.

SOURCE-CODE CONTROL SYSTEM

Most software development organizations use a source-code control system to store deliverables related to a given program. While this provides structure and manages document versions, it often causes difficulty for many team members who don't have access to (or don't know how to use) these systems. Such systems also make it difficult to share information with extended development teams that include partners and business users.

HOME-BUILT WEB SITES

Some organizations hire individuals full-time to create and maintain a Web site for tracking development programs. This alternative provides team members with easy access to deliverables and status, but there is a significant cost associated with it. The owners of the Web site not only spend the time to create and maintain it, but often spend a great deal of additional time tracking down team members in order to get them to provide status and the latest versions of their deliverables for posting to the site. These resources could otherwise be used to help drive the application through the life cycle. The effectiveness of these Web sites depends upon the promptness of the Webmaster in posting team members' updated documents, data, and status reports.

An application development life cycle automation solution must provide a single source for all status and deliverables. That solution must be:

- Easy to create and maintain
- Easily accessible by team members and development partners
- Well organized
- Always up to date
- Able to give executive visibility

Thus, application team members will be able to spend their time developing applications, as opposed to creating and managing Web sites or searching for the right document.

Real-Time Visibility into the Application Portfolio

Making sound investment decisions across a large number of application development programs may be the single most critical challenge faced by an application organization executive.

To maximize expected return on investment, executives must have the information to clearly assess the strategic value of each program, its status, and its apparent risks. An application life cycle automation solution must provide a live view into the status and future plans for all development programs, allowing executives to quickly assess portfolio status. The solution must allow them to manage by exception and take action where appropriate to ensure that the most critical applications yielding the highest ROI are delivered on time and with customer-valued functionality. Too often, executive managers use a stale application roadmap presentation to make feature content or application release date commitments to business users. Since the last executive roadmap review, the application team may have moved the release date and added or deferred key features. Understanding the complete status of a development program is a difficult task. Collecting and reviewing status reports from each team member can be time-consuming and may not provide the key information necessary to determine whether a program is really on track. In a recent study of software development programs, researchers found that at least 7 of the 10 most common signs of application development failure were present before the design was created or a single line of code was written. Visibility into these leading indicators is critical to identifying potential problems and attempting to remedy them.

Globally Dispersed Application Teams

Application development teams are increasingly global. Resources, including employees, contractors, partners, and business users, are geographically dispersed and often have limited hours during the business day to communicate. All too often, critical information gets distributed to these different participants by request only. Moreover, the information delivered to different constituents may be inconsistent and can quickly become stale, limiting the ability to effectively collaborate on the program.

An effective PLA solution must effectively manage geographically dispersed constituents through a scalable and highly available architecture. It must allow collaboration among team members without the need for continual meetings.

Summary

In today's business environment, investment in technology, especially software development, is being led by ROI. Gone are the days of unlimited IS budgets. Executives need to understand where and how their software development budgets are being spent. The first step to understanding this is managing the process and managing the people. Neither can be accomplished effectively without some type of tool that ensures the process is being repeated, and that can provide a central location for the collection and storage of metrics around the process.

Message-Driven Beans and WebSphere 4.0

Using a pseudo MDB

BY MARCEL HEIJMANS

One of the major innovations in the EJB 2.0 specification, the Message-Driven Bean (MDB), is not supported by WebSphere Application Server 4.0. There are several ways to circumvent this shortcoming – you can wait for WAS 5.0, use another application server, or concoct some sort of pseudo MDB. Since the first two are rarely feasible, I'll describe the last.

The Java Message Service (JMS) provides the means for asynchronous communications via Message-Oriented Middleware (MOM). In a J2EE environment, with Enterprise JavaBeans, JMS can be used to produce messages asynchronously. However, EJBs aren't suitable for consuming messages asynchronously. The MDB provides an infrastructure for consuming JMS messages within a J2EE environment. This new EJB type (EJB 2.0) presents the same robust component-based infrastructure that session and entity beans benefit from.

WebSphere Application Server 4.0 supports the latest JMS Specification (1.02b) and is often used in combination with MQSeries 5.2.1 as its MOM. The fact that WAS 4.0 supports JMS doesn't necessarily imply that it also supports MDBs, since MDBs are part of the EJB 2.0 specification and not part of JMS.

WebSphere 4.0 does support more functionality than specified in EJB 1.1, but it lacks several important features of EJB 2.0; for example, the MDB container is not implemented in WAS 4.0. Although MDBs will be available in WAS 5.0, this shortcoming in WAS 4.0 is a nuisance.

Recently, many companies have migrated from WAS 3.5 to 4.0, IBM's mature J2EE platform. Most have invested in the development of mission-critical applications or the migration of their legacy applications to this platform. Due to these investments, the application server market is not as flexible as the desktop market, where new versions are accepted every few months. IBM realized this and came up with an intermediate solution to cope with this deficiency in WAS 4.0. This article provides a brief outline; for a thorough description of the solution, see the IBM Redbook SG24-6283-00.

The proposed solution consists of two parts: a so-called JmsMonitor, which is a queue-monitoring process outside the WebSphere Application Server, and a MessageReader, which is implemented by an ordinary stateless session bean (see Figure 1). Note that the monitor only browses the queue; it doesn't actually read the message. When a message is available on the queue, the monitor invokes the readMessage method of the MessageReader, which reads the message from the queue and invokes its onMessage. The apparent reason for this complicated method is to enable scalability by pooling and container-managed transactions.

What's wrong with this solution? Well, the system administration's having external processes isn't contributing to the maintainability of the system. IBM's solution comes from a Redbook that targets z/OS and OS/390. A solution like this is very natural in these mainframe environments because of the way processes are treated. On other platforms (AIX, Linux, NT, etc.), however, a pure WebSphere solution that can be distributed, deployed, and administered with a single EAR file is preferable. The JmsMonitor doesn't use an event-driven mechanism, it uses a one-time scan through the queue. This "batch" behavior could be extended to some sort of pooling method, but it certainly doesn't fit into an object-oriented J2EE design.

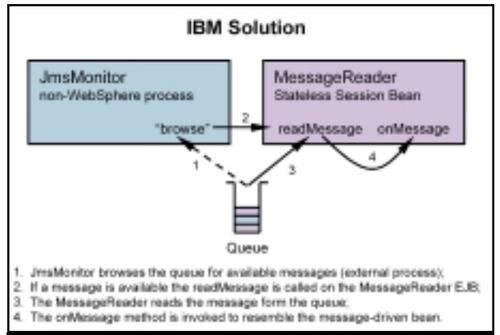


FIG. 1: IBM PSEUDO MDB SOLUTION

ABOUT THE AUTHOR

Marcel Heijmans is a senior software engineer and founder of Mnemonics. He created the J2EE Development Coaching concept, which trains and supports novice developers and architects within their projects while minimizing the project risks.

E-MAIL

marcel.heijmans@mnemonics.nl

After having tried to circumvent message-driven beans altogether, I came to the conclusion that the only way to develop a high-quality application with asynchronous communications is to brew your own MDBs.

Requirements

The first step in the design of an MDB surrogate is to compile a wish list of requirements the pseudo MDB should meet. As with any surrogate, resemblance to the real thing is key. WebSphere Application Server 5.0 will provide a genuine MDB as specified in EJB 2.0. A close match with the MDB will simplify migration later.

A real EJB 2.0 MDB is very easy to use; just implementing the MDB interface will suffice. The application server's MDB container handles the JMS-specific issues (connection, session, receiver, and message listener). Hence, a successful pseudo MDB must be just as easy to use.

An MDB has no home or remote interface that can be invoked by other components. The application server (the MDB container) is responsible for the creation and invocation of the MDB methods. Because of this the application server must start a pseudo MDB and invoke its onMessage method when a message is available on the queue.

Similar to other types of EJBs, MDBs are designed in such a way that they can be pooled to enable scalability and used within container-managed transactions. Again, the pseudo MDB solution must support these vital qualities to be a viable alternative.

I came up with the following rudimentary list of requirements that a pseudo MDB must meet. It must:

- Resemble a real message-driven bean
- Be invoked by the arrival of messages on a queue
- Be easily used by J2EE applications
- Be created during server startup
- Support pooling
- Support transactions
- Be a pure WebSphere solution

The last item is added to provide the previously mentioned single EAR.

Message Monitor and Collector

Since the number of J2EE component types is rather limited, there are only a few possible solutions for building a pseudo MDB. The only thing that will start in WebSphere 4.0 at application server startup is a servlet, and the closest you can get to an MDB is a stateless session bean. A servlet that is started by the application server and implements a MessageListener in combination with a stateless session bean, which receives the message from the servlet, would do the trick (see Figure 2). However, pooling and container-managed transactions aren't supported by this straightforward solution – this was actually a prototype to see whether this works at all. I've included the source code for this prototype servlet/EJB solution for educational purposes, but I won't elaborate on it here (source code for this article is available at www.sys-con.com/websphere/sourcec.cfm).

The golden rule in computer science is, "A layer of indirection can solve every problem." In the final solution, the purpose of the StartMonitor servlet is solely to start a QueueMonitor stateless session bean, which contains the monitor functionality. In turn, this QueueMonitor EJB invokes another stateless session bean (QueueCollector), which serves as the pseudo MDB (see Figure 3). The code is nearly identical to that of the servlet/EJB solution; the only difference is that the monitor functionality is separated from the bootstrap mechanism.

Only one StartMonitor servlet is necessary within WebSphere, which can bootstrap all QueueMonitor EJBs at server startup. The initialization parameters of the servlet determine which QueueMonitors should be activated (see Figure 4). Although the QueueMonitor EJBs are different beans (classes) with different home and remote interfaces, there's an EJB polymorphism technique to create different EJBs in a generic way.

This can be done by extending all the remote interfaces of the QueueMonitor EJBs from both EJBObject and IQueueMonitor and the home interfaces from EJBHome and IQueueMonitorHome (see Figure 5). Since these are interfaces, use of multiple inheritance is allowed. The home interface looks like the following:

```
public interface MyQueueMonitorHome extends
  EJBHome,
    IQueueMonitorHome {
    public IQueueMonitor create() throws
      CreateException,
        RemoteException;
}
```

The IQueueMonitor return type of the create() (which is not the regular remote interface, but its super interface) is crucial. This allows the StartMonitor servlet to create different QueueMonitor EJBs like this:

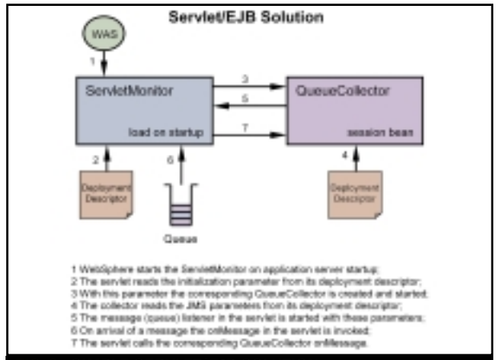


FIG. 2: SIMPLE SERVLET/EJB SOLUTION

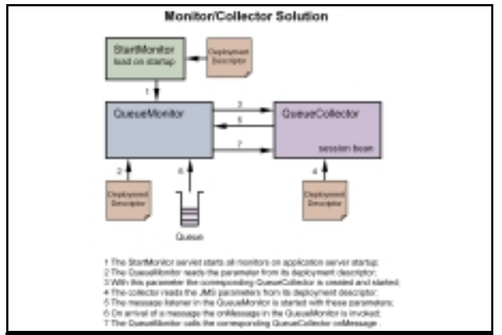


FIG. 3: EJB MONITOR/COLLECTOR SOLUTION

```
IQueueMonitorHome queueMonitorHome =
   (IQueueMonitorHome)
   PortableRemoteObject.narrow(obj,
IQueueMonitorHome.class);
IQueueMonitor queueMonitor = (IQueueMonitor)
   queueMonitorHome.create();
```

(More details on polymorphism and EJBs can be found in the August 2002 issue of ***Java Developer's Journal*** [Vol. 7, issue 8].)

After the StartMonitor server is started and all QueueMonitors are activated, this servlet is done.

The QueueMonitor does most of the work. This stateless session bean handles all communication with the JMS provider (e.g., MQSeries). It connects to the JMS provider, maintains the session, creates the receiver, and implements the message listener and the exception listener. The exception listener provides the means to recover from connection failure with the MOM.

Your QueueMonitor EJB needs to be derived from the class QueueMonitor. To let the QueueMonitor do its work, it must be initialized in the ejbCreate() and destroyed in the ejbRemove(). The combination of the StartMonitor and the QueueMonitor provides functionality similar to that of the JmsMonitor in the IBM solution (and, of course, the real MDB container). The QueueCollector is merely an empty base class that provides the pseudo-MDB characteristics. It runs in the StartMonitor/QueueMonitor environment, just like real MDBs run in the MDB container.

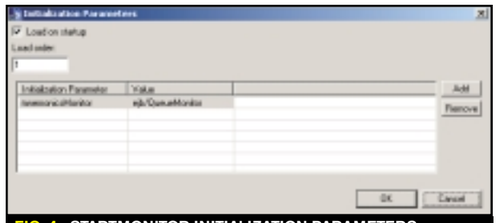


FIG. 4: STARTMONITOR INITIALIZATION PARAMETERS

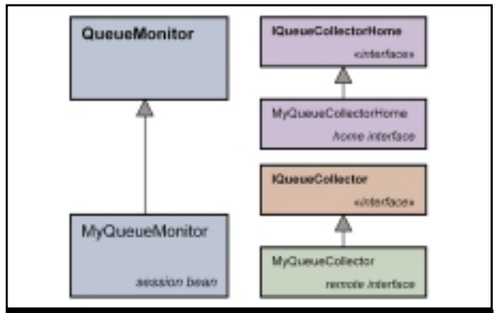


FIG. 5: EJB AND INTERFACE INHERITANCE

FIG. 6: OVERVIEW OF REFERENCE CHAIN

The configuration of a real MDB includes a destination, which is JMS terminology for a queue. Since the deployment descriptor of a stateless session bean is very different from that of the new MDB, there is a certain limit to how well it can mimic the real thing. In order to let the QueueCollector resemble the real MDB, it will store the JMS destination in its deployment descriptor. During startup the QueueMonitor retrieves the JMS destination information from the QueueCollector by calling the startCollector() method. The QueueMonitor initializes its JMS resources with the configuration information that this method returns.

The QueueMonitor uses the same EJB polymorphism described earlier to create the QueueCollector. Similar to the StartMonitor, the QueueMonitor stores the name of the QueueCollector in the deployment descriptor. The "queueCollector" environment variable of the QueueMonitor EJB holds the reference name of the QueueCollector EJB. The result is a rather long chain of names, references, and JNDI names, which is somewhat error-prone (see Figure 6).

Another difference between the pseudo MDB and the real MDB is that there's no way to protect the remote interface of the session bean QueueCollector from being invoked by some other component. A real MDB doesn't have a home or remote interface. There is no adequate solution to this problem – just tell developers not to alter the remote interface of the pseudo MDB.

The only function of the QueueMonitor is reading messages from the queue and calling the QueueCollector to process them. If there are multiple messages on the Queue and the processing in the QueueCollector takes longer than just reading from the Queue, then the QueueMonitor will create multiple QueueCollectors (depending on the configuration, of course). This also indicates that every QueueCollector (EJB) type needs its own QueueMonitor (EJB) type.

If applicable, both stateless session beans need to take part in the same transaction. If you need some form of transactional processing in the QueueCollector, involving the QueueMonitor in the same transaction is mandatory. Otherwise, a rollback would result in loss of the message.

Run on Server

The environment in which MOM operates is often quite complex. The included EAR files need some configuration before they'll run on WebSphere 4.0. I use IBM MQSeries as the underlying JMS provider, which is very common in combination with WebSphere. To enable MQSeries to work with JMS, you must download the JMS Support Pac from the IBM Web site (MA88 Support Pac).

My implementation uses LDAP for the storage of the JMS destinations. I've included the LDIF and the LDAP configuration file for OpenLDAP. More information on the administration of JMS resources (JMSadmin) can be found in the documentation that comes with the MA88 Support Pac and IBM MQSeries. I've also included a WebSphere Studio Application Developer (WSAD) server project that I used during development. The following configurations must be made to the supporting software to run the enclosed solution:

- MQSeries must be installed and running with a Queue Manager, QM1, and Queue, Q1.
- The JMS Support Pac (MA88) must be installed in the default MQSeries folder.
- LDAP (OpenLDAP or IBM SecureWay) must be running with the settings according to the LDIF file.
- The application server (or WSAD test environment) must be configured according to the svr.zip file.

You can change the settings, but remember that there's a relationship between the deployment descriptor of the QueueCollector, the LDAP entries, the MQSeries resources, and the application server settings.

If you experience problems while running the solution on WebSphere, check your configuration. I found that problems are usually caused by small typos in EJB references, JNDI names, or LDAP entries. Moreover, make sure that your supporting middleware is running (e.g., MQSeries QueueManager and Listener service, slapd, etc.).

Conclusion

The solution described here complies with the requirements stated earlier. It simulates a real MDB to the available J2EE components in WebSphere 4.0 as closely as possible. It's very easy to use by simply inheriting from the QueueMonitor and QueueCollector class (the pseudo MDB) and configuring the beans with the deployment descriptor. The servlet bootstrap mechanism works at server startup and is a pure WebSphere solution. The combination of QueueMonitor and QueueCollector provides the means to comply with both the pooling and container-managed transaction requirements.

The QueueMonitor and QueueCollector solution operates in the point-to-point communication domain. This means that there's a one-on-one relationship between sender and receiver. A real MDB can be configured by its deployment descriptor to operate in either the point-to-point domain or the publish-and-subscribe domain. The publish-and-subscribe domain allows multiple subscribers to receive messages by subscribing to a topic. It's easy to develop a topic-based pseudo MDB along the lines of the solution described above. Just replace queue with topic, sender with publisher, and receiver with subscriber, and you're as good as finished. A combination solution that supports both point to point and publish and subscribe like a real MDB is also possible, but it overcomplicates things and there's no substantial gain.

Resources

- Roman, E., et al. (2001). *Mastering Enterprise JavaBeans* (2nd edition). John Wiley & Sons.
- *Java Message Service Documentation:* http://java.sun.com/products/jms/docs.html
- Monson-Haefel, R., et al. (2000) *Java Message Service*. O'Reilly.
- IBM Redbook SG24-6283-00, "Enterprise JavaBeans for z/OS and OS/390 WebSphere Application Server 4.0." http://publib-b.boulder.ibm.com/cgi-bin/searchsite.cgi?query=SG24-6283-00
- IBM MQSeries SC34-5456-07, "Using Java." www.ibm.com/software/mqseries
- Lenz, B. and Musser, J. (2002). "Making Your EJBs Polymorphic." *Java Developers Journal*. SYS-CON Media, Inc. Vol. 7, issue 8.

Monitoring key WebSphere components throughout the development and production life cycle

Ensure Business-Critical Performance

BY MARY HALL

ABOUT THE AUTHOR

Mary Hall, advisory marketing manager of WebSphere Performance Monitoring Solutions, has worldwide responsibility for marketing and sales support for Candle Corporation's middleware and Web management products. Mary has an extensive background – at ISOCOR, she led Internet global marketing efforts for the company's award-winning Web server products. She also spent six years at LexisNexis. Mary holds a bachelor's degree from Miami University in Ohio, and a juris doctor from the University of Toledo.

E-MAIL
mary_hall@candle.com

Which department is not considered "mission-critical" to your organization? To the finance department, immediate access to information is essential to projecting revenue accurately. From the sales perspective, every second takes on added importance when prospects are waiting to download requested product information from your Web site. For a developer, making sure your application infrastructure can support customers and internal departments is imperative. In addition to formalized unit and integration testing processes, it's equally important to gain a clear understanding of how the application will perform early in the development cycle in order to identify and eliminate potential problems.

Achieving the best possible WebSphere application performance isn't just about optimizing your Java code; it's about understanding the different factors that influence modern Web-based applications. Often the variables that impact performance are relatively invisible and typically not key concerns for developers…until bad application performance impacts business productivity. Performance problems are often compounded because many developers don't have the tools to view granular performance activities and pinpoint the source of application slowdowns.

Tracking Key Performance Components

The WebSphere Application Server is a very complex environment. In essence, it's a complete operating system that resides on top of a Java Virtual Machine (JVM). It provides application loading and dispatching services, database connectivity and pooling, execution of remote services and messaging, and external communications via a variety of protocols. The smooth execution of all of these interrelated parts plays a huge role in application performance. In order to truly understand application execution, developers must be able to drill down and view the details of all components.

Take a relatively simple interaction, for example, a customer record lookup. This application runs in a JavaServer Page (JSP) servlet and accesses data via a container-managed persistence Enterprise JavaBean (EJB) backed by a DB2 database. Response time from the user perspective will depend on many factors, and most of the performance variables aren't included in the application code.

For example, servlet execution will depend on the servlet engine's creating and dispatching an execution thread. Accessing the EJB will depend on a Java Naming and Directory Interface (JNDI) lookup, and EJB method execution will require ORB pool resources. If you're using a typical session-facade pattern, you may be invisibly aggregating multiple EJB method invocations, thus compounding the strain on resource usage. Finally, accessing the data will require database connection pool resources. Even in this relatively simple example, there are many areas to monitor. You first need to track Web container thread pool usage, database connection pools, the servlet request and error rate, session creation, and invalidation statistics. You should also monitor performance within JVMs.

EJB performance can have a major impact on the overall execution of your application, and many areas may need to be addressed. EJB pool hit rates, EJB pool usage, activation/passivation, and create/destroy cycle rates are all key performance metrics that highlight problem areas. Additionally, we want to keep statistics for individual methods, including method invocation rate and method elapsed time.

ORB thread pool usage is another critical metric to understand and track. Because every EJB request requires a thread, the ORB pool must be sized appropriately to prevent waits and time-outs under heavy load. Bear in mind that both servlet-to-EJB requests and EJB-to-EJB requests that may be hidden behind a session facade will use the ORB thread pool.

Other important items to track for performance are database connection pools and transaction rates. When monitoring database connection pools, look at the percentage in use and size the connection pool to sufficiently service transactions without causing waits. Additional areas to monitor include threads, average connection-use time, and prepared statement cache utilization.

Developers are able to tune capacity to enhance performance based on information gathered from cataloging the frequency and duration of threads that exceeded the pool capability. By applying a load generator to the application view, they can conduct trend analysis under conditions that simulate production environment traffic to more accurately determine how an application will respond under real-world conditions.

Tracking performance inside JVMs is crucial. Over time, graphing free memory in the JVM will produce an approximately sawtooth pattern, as memory is allocated and then freed by garbage collection. Observing the trend of the peaks and valleys, as well as the amount of memory allocated during each garbage collection cycle, can reveal memory usage patterns that might indicate problem areas in your applications. Monitor heap size to strike a balance between too-frequent garbage collection caused by heap size allocations that are too small versus less frequent but longer-running garbage collections. A tool that will let you view a memory usage graph in relation to transaction rate or servlet request rate can provide a very good visual indication of the overall working of your applications.

WebSphere tuning is a complex and continuous process. Thousands of pages of contradictory tuning advice have been published, which often leaves the developer without a clear path to resolution. It is important to use development guides in the context of your own applications and with the aid of a good monitoring tool.

Developers need to give performance measurement and monitoring significant attention throughout the development cycle. Often, tuning is something that's done after deployment or in response to a performance crisis. Waiting until an application has been created before testing significantly limits the options to rectify the problem. Redesigning parts of the application flow, for example, would be too cumbersome and complicated to be a viable solution. Concentrating on application performance early in the development cycle has an additional benefit because it helps to increase a developer's understanding of the effects of design and implementation decisions. This knowledge, in turn, increases overall software quality.

Optimizing Through Continuous Monitoring

With organizations making large human resource and software investments to deploy WebSphere applications, it's critical to create a foundation for new applications that will support high-performance benchmarks and scalability. In many cases performance tuning begins only after deployment, which is tantamount to waiting until building construction has been completed before inspecting the structural integrity of the foundation.

To ensure that an application is designed to withstand real-world demands, developers must test application performance throughout the life cycle to detect, isolate, and eliminate performance problems, preventing them from impacting overall application health. By tracking metrics of critical components within the application and across the WebSphere environment, developers will have the information required to create and deploy new applications that will readily maintain peak performance and maximize corporate IT investments.

Sample components, workflow, and discovery in the life sciences

IBM Life Sciences Framework PART 2

BY MICHAEL NIEMI

To help address the challenges faced by pharmaceutical, biotech, research, and medical organizations, the IBM Life Sciences Framework uses industry-standard technologies (J2EE, XML, Web services, etc.) and protocols and data representations from standards efforts such as the I3C (Interoperable Informatics Infrastructure Consortium), OMG-LSR (Object Management Group-Life Sciences Research), HL7 (Health Level 7), and the Bio* projects.

ABOUT THE AUTHOR

Mike Niemi is a senior software engineer in the IBM Life Sciences organization. He has held various positions in both software and hardware development over the past 24 years. His experience includes development in voice systems, WebSphere Application Server and SiteAnalyzer, TCP/IP, and mainframes.

E-MAIL

niemi@us.ibm.com

In my previous article (***WSDJ***, Vol. 1, issue 7) I showed how to invoke XEMBL, a Web service that takes an accession number as an input parameter and returns a nucleotide sequence. Web services, based on SOAP and WSDL standards, facilitates the development of loosely coupled, distributed systems. In this article we will look at developing workflows and the use of a registry to publish and discover Web services for the life sciences.

XEMBL is an example of one component of a Web services architecture. It uses the SOAP protocol to accept parameters from the service requester and returns an XML document containing the nucleotide sequence. In order to create useful applications it is necessary to have a robust set of components orchestrated in some sort of workflow.

SOAP-enabled components may be publicly available, for example XEMBL and BQS (hosted by the European Bioinformatics Institute), or caBIO (hosted by the National Cancer Institute Center for Bioinformatics [NCICB]). They may be implemented as wrappers to legacy applications, for example a wrapper to the NCBI BLAST (Basic Local Alignment Search Tool) program. They may be included in the future as standard API interfaces within vendor applications (early examples of this trend are found in Documentum, Plumtree, Spotfire, WebSphere Portal, and Google).

Workflows will be based on coming standards such as BPEL4WS (Business Process Execution Language for Web Services) and be supported by tools such as MQSeries Workflow.

In this article we will show a use case from the life sciences; discuss the components, vocabularies, and protocols to address that use case; look at the use of workflow, registries, and ontologies to support that solution; and finally, consider some overall solutions.

Sample Use Case

Use cases are a handy way to focus on a system's design and development. They help flesh out the requirements, as well as informally validate the design and implementation. Their use can be formalized as in the case of UML (Unified Modeling Language) and the HL7 methodology, or less formal, such as in the case of the I3C (Interoperable Informatics Infrastructure Consortium) Roadmap. I will briefly sketch out a life sciences scenario that we will use as our guinea pig. It is from the I3C Registry Group project.

In this scenario a scientist interested in studying the role of one of the proto-oncogenes in neuroblastoma (brain cancer)

- Conducts a PubMed search for "Dbl related novel genes and neuroblastoma"
- Retrieves an article
- Clicks on related articles, obtaining 319 articles
- Retrieves the accession numbers from articles that provide them
- Submits these to XEMBL, which returns a BSML (Bioinformatic Sequence Markup Language) document
- Passes this sequence to the BLAST algorithm, which filters the sequences based on percentage identity
- Uses geneid to know the gene structure
- Passes the protein sequence to BLAST against SWISS-PROT
- Uses Bio-Dictionary to identify relevant articles from PubMed, then uses text mining to categorize the results and look for concepts
- Uses ClustalW and Tereisisas to look for common patterns in the BLAST results (this is called multiple sequences alignment)

At the end of this the scientist can predict the functionality of the novel proto-oncogene and hypothesize its probable role in brain cancer.

Now we will look at the components we need to implement a solution for this use case.

Components and Vocabularies

Some of the components needed for this use case are shown in Figure 1. Note that this list is a work in progress. See the I3C Registry Group for more details. Other components would be used to create the graphical user interface. One of the objectives of standards organizations and industry consortiums is to identify and define standard XML vocabularies for data interchange between components such as these. A good example of this is the MAGE-ML vocabulary developed by the Microarray Gene Expression Data Society and the OMG-LSR (Object Management Group-Life Sciences Research). Another good example of vendors coming together to work on a common data model and XML vocabulary is the V3 CDA (Clinical Document Architecture) XML vocabulary of HL7. See the Resources section for other examples of common vocabularies in the life sciences and some of the standards organizations working on them.

Registries, Ontologies, and Workflow

Just to review the Web services basics (see Figure 2), the service requester binds to the service hosted by the service provider at design time or run time. The service registry is optional. The requester may have discovered the service in any number of ways (see Figure 3).

Possibilities range from e-mail to a registry. At the upper end of the spectrum, the service provider publishes WSDL describing their services in a UDDI registry. The service requester can search the registry at design time (by the system developer) or at run time (by the end user or an automated process). A UDDI registry could describe a service using taxonomies and categories (see the UDDI overviews in the references below). A taxonomy or ontology might be used to describe an algorithm or entry in a database.

Services might also be categorized using name-value pairs. The values could be concepts of an ontology; for example, the following might be used to describe a BLAST service:

```
input object type = {BSML,
  AGAVE}
output object type =
  NCBI_BlastOutput
service type = BLAST
```

This is the sort of characterization that the BioMOBY project is attempting to utilize. The BioMOBY project is an international research project involving biological data hosts, bio-logical data service providers, and coders whose aim is to explore various methodologies for biological data representation, distribution, and discovery.

Ontologies can also be used to describe databases. The ontologies developed by the GO (Gene Ontology) Consortium are a leading example of this in the life sciences. The consortium has developed three types of related ontologies, for molecular functions, biological processes, and cellular components.

A GUI or automated process might then use this information (the nodes of a graph) to help the user construct a workflow for the problem he or she is trying to solve. Capturing and saving successful workflows is a way for an organization to build a knowledge base. This is one of the goals of myGrid, which aims to design, develop, and demonstrate higher- level functionalities over existing Web services and grid infrastructure to support scientists in making use of complex, geographically distributed resources.

For the use case described above, XEMBL would be called with an accession returning a BSML XML document (my previous article

Service	Description	Input(s)	Output(s)
XEMBL	Retrieves Nucleotide Sequence(s) and associated information	Accession Number	BSML 2.2 or AGAVE
BQS	Access to MEDLINE		
Blast	Invokes NCBI blastall	BSML 2.2 or AGAVE	NCBI_BlastOutput
BodyMap			
ClustalW			
GenBank			
GeneID			
HMMER			
PubMed			
Teiresias	Pattern Discovery		
MUSCA	Multiple Sequence Alignment		
Bio-Dictionary™	Protein Annotation		
Genes@Work	Analyze patterns in gene array chip data	MAGE-ML	

FIG. 1: PROSPECTIVE COMPONENTS FOR LIFE SCIENCES SERVICES–BASED SYSTEM

included code samples that do this). The BSML document can then be passed to the BLAST service. This would be captured as a small portion of a workflow. An ontology describing the BLAST service might help guide the developer or end user. A registry is consulted as described in the hypothetical systems below.

Possible Systems

So, let's try to tie this all together in three hypothetical systems that use workflow and a registry to describe and discover services.

First, we might imagine a user interface in which the developer is able to select from a palette of components to construct an overall application. The registry might be static – listing the components of one or more vendor applications together with stable public components that have been brought inside the firewall. The developer might use pieces of custom code as necessary to supplement the standard palette. In this case, the developer is an expert in the field. When the application is deployed, the end user may only need limited flexibility in customizing the application. This is an example of constructing a static workflow using a private UDDI registry. The UDDI registry is a way for the enterprise to stabilize the components used within the enterprise.

Next, we might imagine a GUI that the end user could use to construct workflows dynamically. The GUI uses the registry to guide the user in constructing viable workflows. Again the user may supplement the standard components with code or scripts as necessary. Ontologies may be included to help the user to develop higher-quality workflows – guiding the user to a successful solution. The palette of components might change from day to day – reflecting new components continually brought into the organization and shared among the users. The dynamic nature of this ebb and flow of components provides the competitive advantage to the organization. The end users are sophisticated bioinformaticians. Successful workflows are saved and shared among the enterprise.

Finally, we might consider systems in which parts of the workflow are largely automated. This would require high-quality ontologies and smart agents.

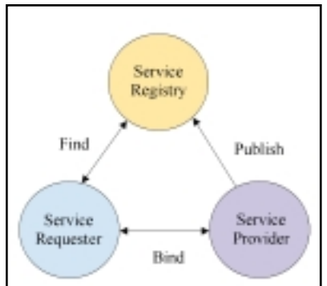


FIG. 2: WEB SERVICES BASICS

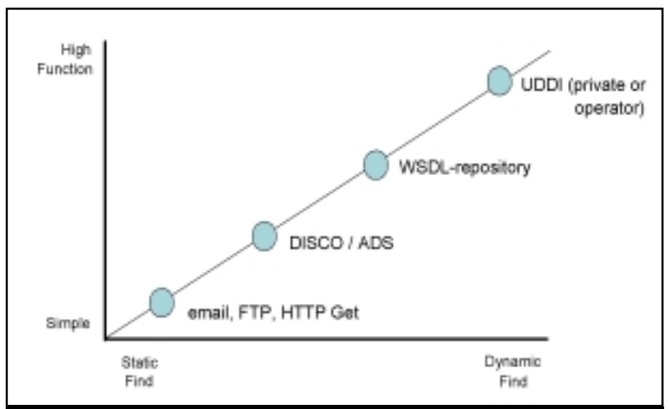


FIG. 3: SERVICE DISCOVERY METHODOLOGIES

Conclusion

In my previous article we looked at invoking one Web service in the life sciences. In this article we have filled out the palette of components and touched on registries, ontologies, and workflow to bring those components together. For more information, please have a look at the BioMOBY project (www.biomoby.org), myGrid (www.mygrid.org.uk), and the I3C Registry Group (on www.i3c.org).

References

- *List of XML in Bioinformatics:* www.xml.com/pub/rg/Bioinformatics
- *I3C:* www.i3c.org
- *OMG-LSR:* www.omg.org/homepages/lsr
- *HL7:* www.hl7.org
- *CDISC:* www.cdisc.org
- *MGED:* www.mged.org
- *GO Consortium:* www.geneontology.org
- *Web services:* www.ibm.com/developerworks/webservices
- *BPEL4WS:* http://xml.coverpages.org/ni2002-08-12-a.html
- *MQSeries Workflow support for Web services:* www.ibm.com/software/ ts/mqseries/workflow/sp-wa07. html
- *I3C Use Case Roadmap (draft at the time of this writing):* www.i3c.org/workgroups/technical_architecture/resources/I3C_Roadmap-Working_DRAFT_J1.doc
- *BioMOBY:* http://biomoby.org
- *I3C Registry Group:* www.i3c.org/workgroups/technical_architecture/registry/index.html
- myGrid: www.mygrid.org.uk
- Berners-Lee, T., et al. "The Semantic Web." *Scientific American:* www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21
- Wroe, C., et al. "A Suite of DAML+OIL Ontologies to Describe Bioinformatics Web Services and Data," draft to be published in the *Journal of Cooperative Information Science:* www.cs.man. ac.uk/~wroec/mygrid_ontologies_paper_draft.doc

DIRIG ANNOUNCES AUTOMATED PERFORMANCE MANAGEMENT SOLUTION

(Nashua, NH) – Dirig Software, a developer of enterprise performance management solutions for e-business, has introduced PathFinder, which automatically identifies transaction dependencies across Web servers, application servers, and databases. PathFinder changes the way enterprise IT departments utilize system management tools, allowing them to communicate the effect Web application performance has on other parts of the organization, such as development and line of business.

PathFinder allows IT to gather the relevant statistics for a Web application–based transaction and aggregates the data to deliver success-to-failure ratios and overall health.
www.dirig.com

NEW IBM SOFTWARE SPEEDS INTEGRATION OF LEGACY SYSTEMS

(Somers, NY) – IBM has announced the release of new software to help companies extend their legacy assets to make the most of Web services and an open e-business infrastructure. The new versions of IBM WebSphere Host Integration solutions are available immediately. New features include support for IBM WebSphere Portal, enabling access to legacy applications in a portal environment without rewriting code; a new product called WebSphere Host Access Transformation Server; and several improvements to the Host Access Client Package, streamlining administration and increasing end-user productivity for users and IT administrators.

The IBM Host Access Client Package and the WebSphere Host Integration Solution offer a fast, cost-effective way for businesses to provide secure access to legacy applications via a browser, whether to corporate users inside a firewall or supply-chain partners accessing legacy applications across the Internet.
www.ibm.com

SINC ANNOUNCES GRAND OPENING OF WEBSPHERE INNOVATION CENTER

(Montreal) – Solution Integration of New Concepts (SINC), a company specializing in networking, Internet connectivity, e-business development, and infrastructure deployment, has announced the opening of an IBM Business Partner Affiliated WebSphere Innovation Center (WIC) in Montreal. The center will be the 18th such center worldwide and the first WIC in Canada.

WIC provides "hands-on" access to a variety of hardware platforms and operating environments, allowing business partners to test-drive technological solutions in a reproduction of their IT infrastructure. The center will also provide proof-of-concept support and customer technical briefings.
www.sinc.ca

SERENA EXTENDS SUPPORT FOR WEBSPHERE STUDIO PLATFORM

(San Mateo, CA) – Serena Software, Inc., an industry-leading supplier of software that automates change to enterprise code and content, has announced it will extend its support for IBM's WebSphere Studio platform, providing customers with a comprehensive Enterprise Change Management solution. Serena ChangeMan DS will support WebSphere Studio Version 5, the latest release of IBM's end-to-end J2EE application development tool.

The product, which supports WebSphere Studio v.4, enables customers to accelerate next-generation development while increasing the quality and reliability of their enterprise applications.

Serena ChangeMan DS also offers companies migrating from VisualAge for Java a complete, fully automated solution that integrates seamlessly into the existing IT environment.
www.serena.com

CONTEXT MEDIA INTRODUCES CONTENT MANAGEMENT SOLUTION

(Providence, RI) – Context Media, Inc., has released Interchange Suite 3.0, its enterprise content management software solution that enables companies to organize, manage, and distribute content across and outside the enterprise by integrating disparate and distributed installed content management systems. Interchange Suite delivers enhanced ROI without replacing existing systems, disrupting established workflows, or requiring content objects to be moved or replicated into a separate centralized repository.
www.contextmedia.com

COMMUNIX LAUNCHES APPSYNERGY, A RAPID PROVISIONING SOLUTION

(London) – Communix Limited, a value-added reseller of e-business infrastructure products and services, has launched AppSynergy, a solution that dramatically reduces the cost and time it takes to procure, install, and configure IBM WebSphere on workstations and servers for development, test, and production environments.

"Normally, it takes a single skilled person one to three working days to set up a single J2EE development workstation or test server; it's a tedious, error-prone job that entails multiple downloads, tricky installations, and configurations of application server, database, Java IDE, JDBC drivers, and their latest software patches – assuming the hardware and its operating environment are ready," says Leo Tsesmelis, technical director of Communix. "This effort can escalate to, and delay a project by, weeks or even months in a medium-sized team that requires a dozen standard-build workstation or test server environments. With AppSynergy, this time can be reduced to just hours, with an equally great reduction in cost."
www.communix.com/appsynergy

Web Commerce Performance Management

At the top of CIOs' holiday lists

BY MARY HALL

While the hullabaloo of the dot-com era is now a whisper, the e-commerce phenomenon, a critical plank in the prophesied e-revolution, continues to gather momentum. In B2C and B2B markets, purchasers come to the virtual storefront for convenience and leverage.

The holiday season is a stressful time of year. Many of us, in search of the perfect gift, spend hours battling traffic and fighting for parking. Today, the Web is the preferred mall of millions who wish to accomplish this feat in a matter of minutes.

As growing numbers rely on online purchasing, new traffic jams are clogging the virtual storefronts. When crowds hit popular sites all at once, performance suffers and shoppers are left waiting to locate items. As online shopping volume increases, the representative/customer ratio declines, leaving potential purchasers frustrated and impatient at checkout. According to research, eight seconds is the magic response time – when customers are forced to wait longer for online selections, they click away to another store. While poor performance erodes customer satisfaction, site crashes that take stores offline can be catastrophic.

In the brick-and-mortar world, customers are vexed to find stores closed during business hours. Lack of availability is more damaging in the online world. Shoppers quickly click elsewhere to satisfy their requirements. If they have a good experience at another site, they may never return.

We are witnessing a Darwinian dynamic across global e-commerce markets. Purchasers vote with their credit cards, eliminating the constraints of physical proximity and currency exchange. Only the fittest organizations will survive, as customers reinforce their demand for "immediate" gratification. Sites with rapid content loads and transaction speeds generate customer loyalty, high sales volumes, and profitability.

Beneath the facade, commerce infrastructure solutions are clearly complex environments. A customer visiting www.rollingstones.com is looking for music and video clips and is focused on buying concert tickets in one quick visit. To serve up rapid, intuitive, seamless functionality, organizations must bring together and synchronize a tremendous amount of technology. This reality is driving significant demand for Web server and application infrastructure monitoring and management solutions.

Holistic Approach

To achieve success in application server management, organizations must embrace a holistic approach. This means monitoring the full range of "engine room" applications and resources that support response, serving, and transaction functionality – operating systems, middleware, application servers, back-end databases, overall Web architecture, and so on. At the same time, they must monitor customers' experiences against the eight-second rule. This dual approach empowers organizations to detect issues and drill down to understand and tune systems to improve the customer experience.

Clinical Diagnostics, Prioritization, and Remedy

Monitoring and management solutions empower organizations to take a proactive stance on performance and availability. For example, reports empower organizations to identify the most frequently requested applications and map performance to specific resources and demands, including Enterprise JavaBeans (EJBs), servlets, available memory and utilization, the number of applications being processed, and the number of concurrent sessions. Typically, there is an inverse relationship between number of sessions and response-time performance. Managers can leverage application priority, performance, and dependency information to drill down and tune at the component level.

Dashboard Approach to Success

To put first things first amid the complexity, many users set up a top 10 list of things to monitor on a regular basis. It's important to select a tool that empowers the manager to structure a proactive, personalized, hourly, daily, or weekly list.

Intuitive Reporting Requirement

In light of the complexity of Web commerce environments and increasing interest from line-of-business and executive managers in associated revenues and performance, management solutions are no longer purely technical tools. Organizations require intuitive reports that are easy to understand, navigate, and communicate. It's important to consider that these reports will play a key role moving forward in defensive investment justification and offensive investment-request scenarios.

Web commerce is clearly here to stay. As we count down to the holiday shopping season, increasing numbers will elect to skip the day-after-Thanksgiving crush and instead shop online. As the trend blossoms, robust Web performance monitoring and management solutions will become an increasingly common feature on the holiday lists of CIOs and business-focused management executives.

ABOUT THE AUTHOR... Mary Hall, advisory marketing manager of WebSphere Performance Monitoring Solutions, has worldwide responsibility for marketing and sales support for Candle Corp.'s middleware and Web-management products. Mary has an extensive marketing background – at ISOCOR, she led Internet global marketing efforts for the company's award-winning Web server products. She also spent six years at LexisNexis. Mary holds a bachelor's degree from Miami University in Ohio, and a juris doctor from the University of Toledo.

E-MAIL
mary_hall@candle.com