SYS-CON MEDIA

www.SYS-CON.com/WebSphere

WebSphere ®
JOURNAL

The World's Leading Independent WebSphere Resource

FEBRUARY VOLUME 3 ISSUE 2

Community Integration
with WebSphere Business Integration Connect

BY SCOTT **SIMMONS** PAGE 6

Keep the little problems from turning into big ones

diagno*Sys*

Learn how to improve your
Web applications at H&W's
free Web seminar.
Visit www.hwcs.com/wsj1.asp

You might not even know you have a leak, but that little drip can turn into a big deal if it's left unattended. Performance problems with your Web applications work the same way.

Hung threads, memory leaks, and faulty network connections can cause your application to perform poorly or even fail when it matters most – in production. Customers click away and profits plummet. But how do you find the problems if you don't even know they are there?

DiagnoSys is a complete software solution that allows you to analyze the performance of your Web applications through all phases of the life cycle. With true statistical correlation, in-depth analysis, and root cause identification, it helps your teams work together to find problems while they are still little.

The end result? With DiagnoSys, you build vital applications that optimize uptime and keep the bottom line healthy. No problem.

H&W

www.hwcs.com | 1.800.338.6692

Bridging the Enterprise Computing Gap Since 1979

IT Needs to Saddle Up for World's Biggest Roundup

BY JACK **MARTIN**

Some new opportunities are appearing in the world of technology, some of them quite unexpected. Faced with the specter of mad cow disease appearing in the U.S., the American rancher and the entire chain of U.S. food suppliers are about to embark on a far-reaching program that will affect every American consumer who eats beef – and eventually any animal product.

The government has set a deadline for all growers and food processors to track down the whereabouts of every head of cattle in the country – from birth to slaughter – and to identify all of its offspring and any other cattle that have come into contact with it in its entire life span. This government regulation mandates that this tracking and identification of cattle be able to be accomplished within 48 hours.

The industry is eying RFID (radio frequency identification) as the most likely cost-effective solution to accomplish this. The system works simply: put an RFID tag with a serial number on each cow, store the information in a database, and scan in all the data, including everything that's been fed to the cow and all its medications, right up through processing.

Adoption of this new technology may for the first time give consumers the ability to scan a porterhouse steak in their favorite food store and see the entire history of the cow that the steak comes from – from calving and weaning to processing.

There are approximately 96 million cows in the U.S., worth about $70 billion. America's ranchers and food processors are not the most technologically adept folks. They will require enormous amounts of consulting and education to put in place this crucial safeguard for America's food supply. Engagements will range from helping a small family-run ranch to comply with the new regulations all the way up to assisting the largest meat packers in the world to become compliant with these same regulations.

This will be an enormous undertaking. The economic consequence of noncompliance is the placing in jeopardy of the $3 billion export market. Currently, Japan, Taiwan, and South Korea have all banned the importation of American beef products, all because one cow – which entered the U.S. from Canada in a wholesale transaction – has been diagnosed with mad cow disease.

RFID is starting to appear in a myriad of daily interactions, highlighted by Wal-Mart's corporate edict in January requiring their 100 top suppliers to start RFID-tagging all products shipped to Wal-Mart by Dec 2004. Companies unable to comply will be removed from Wal-Mart's vendor list.

RFID is poised to become the single most pervasive passive technology in our society – a fact that has not been missed by our friends in India. Recently, Infosys Technologies, India's largest outsourcing company, entered into an alliance with the Massachusetts Institute of Technology (MIT) to work on RFID inventory systems. The reason for this alliance is simple: Indian tech companies have discovered that they do an abysmal job when it comes to anticipating what U.S. and European corporations want. In an effort to change that, they're trying to develop business expertise in industries like banking, retailing, manufacturing, and energy.

In my opinion, every reader of this editorial should contact Charles Vest, the president of MIT, and ask him why this alliance is in the best interest of the U.S. – and how it will benefit the large number of MIT alumni who possess information technology degrees.

Help like this is exactly what the American worker does not need. If MIT firmly believes that assisting offshore companies is in the country's best interest, due to their reduced cost, I ask Mr. Vest why he hasn't significantly reduced tuition rates for the current MIT student body. By taking this path, he is assuring that they will have fewer opportunities and a reduced standard of living. 🌐

ABOUT THE AUTHOR... Jack Martin, editor-in-chief of *WebSphere 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. Jack is coauthor of the upcoming book, *Understanding WebSphere*, from Prentice Hall. **E-MAIL...** jack@sys-con.com

SYS-CON MEDIA

Two years without a vacation.
The application's up. It's down.
It's up. It's down.

I'm to blame. Steve's to blame.
Someone's always to blame.

Not any more.

Get Wily.™

Enterprise Java
Application Management
1 888 GET WILY
www.wilytech.com

Community Integration with WebSphere Business Integration Connect

Extending integration between trading communities

BY SCOTT **SIMMONS**

Community integration elevates collaborative commerce to a new level of integration between enterprises. In the past, traditional B2B solutions have fallen short of market expectations for community integration due to scalability issues, lack of transaction visibility across the trading community, and minimal partner participation in community management. To solve the tough challenges of B2B, new integration solutions must address these requirements to provide an optimal foundation for trading partner collaboration.

ABOUT THE AUTHOR

Scott Simmons is the worldwide technical lead architect for B2B integration for Worldwide WebSphere Business Integration and is an IBM Certified Senior IT Architect. Scott joined IBM in March 2002 from Peregrine/Extricity, where he was the director of Solution Technologies for Peregrine's Office of the CTO. In this role, Scott specialized in B2B solutions for the technical manufacturing sector. Scott has over 20 years of experience in the IT industry.

E-MAIL

scottsim@us.ibm.com

This article discusses the concept of community integration and offers an architectural overview of how WebSphere Business Integration Connect can enable the rapid implementation of trading communities.

What Is Community Integration?

Community integration is about extending integration beyond the enterprise. It defines a document exchange framework that optimizes business processes across enterprise boundaries. Within organizations there are numerous interenterprise relationships with external partners, some manual and others automated. Over the past 15 years, EDI (Electronic Data Interchange) has been established as a proven foundation for automating these relationships. The ongoing maturation of integration standards and the emergence of the Internet as an open communication framework have resulted in these relationships being extended to Internet-based document exchange such as XML over HTTP and RosettaNet. However, automating these relationships does not provide community integration in itself.

Each partner interaction is inherently specialized in terms of communication transport, document exchange formats, and quality of service requirements. As a result, the framework for B2B integration must be both flexible and scalable, and should allow for the shared management of the relationship at the hub and the partner/participant layer. Community integration is realized when the trading hub provides management access and visibility across the community to the hub and participant constituencies.

Traditional B2B solutions have been of two basic varieties: VAN (Value Added Network)–based offerings built on FTP/File Transfer communication interchanges, and point-to-point solutions for Internet-mediated B2B. In both cases the integration is predicated on point-to-point connections, with each connection configured, implemented, and managed separately.

These solutions offer only a limited ability to enable partners to provide community management and have not provided global visibility into hub operations. Additionally, the effort required at the hub to manage these communities in the face of technological, economic, and business changes becomes nearly impossible. More important, both of these solution architectures promote a "ready, fire, aim" mentality and result in a reactive approach to building and managing trading partner relationships.

Community integration provides a foundation for scaling trading communities into the hundreds and thousands of participants. By providing filtered visibility for partners into hub operations and an integrated approach to alert/exception management for both the hub and the partner community, this solution architecture enables shared operational responsibility across the community. This shared management focus allows the hub to rapidly add processes and partners to the community while at the same time reducing the overall incremental cost to partners to operate the trading community.

Community Integration and WebSphere Business Integration Connect

WebSphere Business Integration Connect is a community integration solution that was launched by IBM in partnership with Viacore, a service provider offering a comprehensive suite of community integration services for private trading communities. Through its partnership with Viacore, IBM offers a complete community integration package: The WebSphere Business Integration Connect software, plus a range of community integration services to accelerate the planning, implementation, and operation of the integration with trading partners.

As an example, a major U.S. electronics distributor embarked on a major initiative to align a complex supply chain. The organization needed to quickly implement a private trading network to exchange real-time business

processes with its suppliers and customers. This initiative required the synchronization of external document exchanges with internal business processes to support integrated order management, forecasting, and supply chain planning. By utilizing a community integration framework from Viacore, the company successfully implemented an integration solution providing collaborative management and transaction visibility across the trading community. The use of Viacore's BusinessTone solution allows the organization to react more quickly to excep-

- *Advanced Edition:* Enables the creation of larger trading communities that require more sophisticated community management tools. It is ideal for trading communities that will start small and grow over time. The Advanced Edition enables integration over numerous transports and offers a rich set of tools for deep community integration.
- *Enterprise Edition:* The solution for deploying large trading hubs with support for unlimited connection definitions. The Enterprise Edition offers the same core

"The framework for B2B integration must be both flexible and scalable, and should allow for the shared management of the relationship at the hub and the partner/participant layer"

tions, shortages, and trading-partner issues and, as a result, drive increased customer satisfaction and enable a more proactive real-time organization.

WebSphere Business Integration Connect is a Java-based software component architecture that offers three editions designed to meet the needs of different-size businesses:

- *Express:* Designed for small to medium-size businesses that want to integrate small trading communities with an intuitive, Web-based integration solution. The solution leverages AS2 and HTTP standards for transmitting documents securely over the Internet

technology as the Advanced Edition, differing only from a licensing perspective.

Advanced and Enterprise Editions: An Architectural View

WebSphere Business Integration Connect 4.2 is implemented on the embedded version of WebSphere Application Server (v5.0.2) and inherits all of the functions of WebSphere Application Server. WebSphere Business Integration Connect provides support for standards-based transports (HTTP, HTTPS, SMTP, FTP, JMS/WebSphere MQ), protocols (RNIF 1.1, RNIF 2.0, AS1, AS2), and a wide

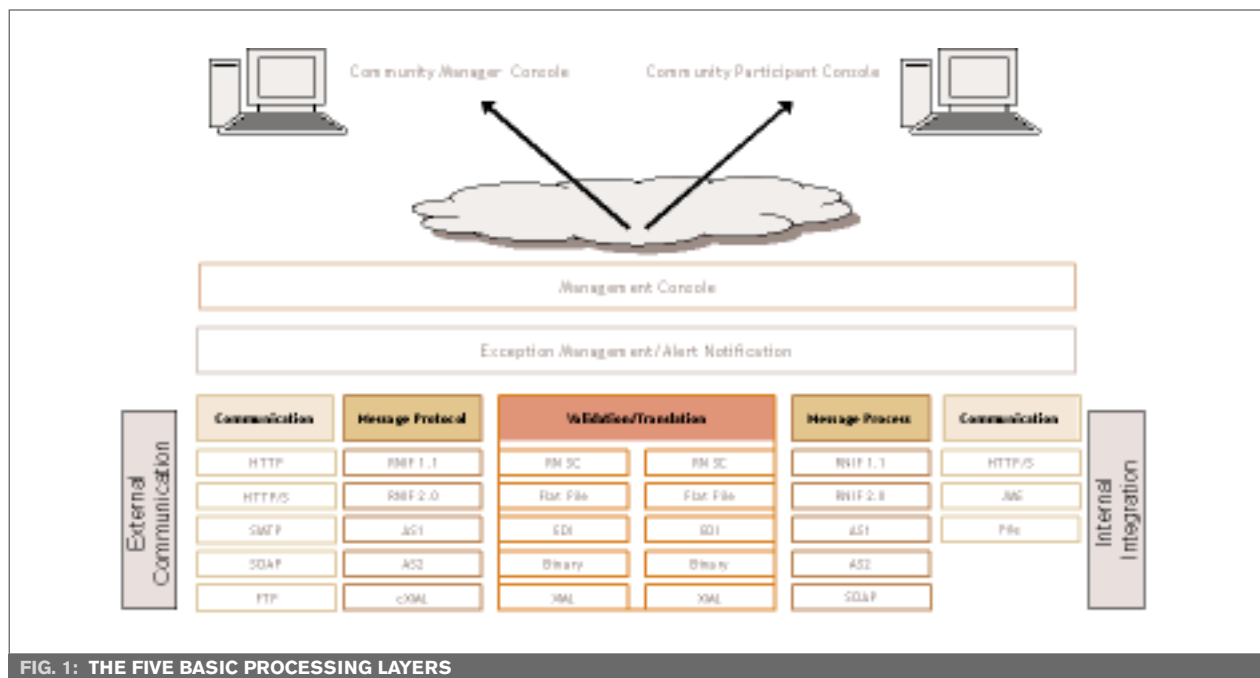| External Communication	Communication	Message Protocol	Validation/Translation		Message Process	Communication	Internal Integration
	HTTP	RNIF 1.1	RN SC	RN SC	RNIF 1.1	HTTP/S	
	HTTP/S	RNIF 2.0	Flat File	Flat File	RNIF 2.0	JMS	
	SMTP	AS1	EDI	EDI	AS1	File	
	SOAP	AS2	Binary	Binary	AS2		
	FTP	cXML	XML	XML	SOAP		

FIG. 1: THE FIVE BASIC PROCESSING LAYERS

WEBSPHERE BUSINESS INTEGRATION CONNECT 4.2 PLATFORM SUPPORT

WebSphere Business Integration Connect Advanced/Enterprise 4.2 currently runs on Intel platforms. The software runs under RedHat Linux AS 2.1. The WebSphere Business Integration Connect Advanced/Express 4.2.1 release will support IBM pSeries (with AIX 5.2) and SUN Sparc (Solaris V8). Additionally, the 4.2.1 release will support SUSE Linux ES V8 and Windows 2000 on Intel platforms.

For the 4.2 WebSphere Business Integration Connect Advanced Enterprise release, the recommended Intel architecture is a 2GHz Intel Xeon processor with 300MB available disk space for the application and additional disk space as needed for document storage. Additional servers can be added for capacity and/or redundancy – multiserver installations require implementation of a shared file system to support distributed document access.

Software requirements for WebSphere Business Integration Connect Advanced/ Enterprise include:
- Database: DB2 8.1 FP 2 or Oracle Database Server, version 9.2 with Oracle JDBC thin driver (Oracle support is included in the 4.2.1 release)
- WebSphere MQ, v5.3 or later
- Web browser: Microsoft Internet Explorer, v5.0 or higher; Netscape, v6.0 or higher for Community Console access
- Simple Mail Transport Protocol (SMTP)-server for e-mail alert delivery and SMTP-based message delivery

WebSphere Business Integration Connect Express 4.2 requirements are as follows:
- 1.4 GHz or faster Intel Xeon processor
- At least 512MB of random access memory (RAM)
- At least 100MB of available hard disk space
- Microsoft Windows 2000 operating system, with Service Pack 3 installed
- Microsoft Internet Explorer, version 5.5 or higher; or Netscape, version 6.0 or higher for Express Console access
- A Simple Mail Transport Protocol (SMTP)-based e-mail relay server for delivering e-mail alerts

Note: WebSphere Business Integration Connect Express 4.2.1 will also provide support for Red Hat Linux AS v2.1 and SUSE Linux Enterprise Server v8 with SUSE Linux, kernel 2.4.

range of payload formats (EDI, XML, RosettaNet, cXML, binary/raw payloads). SOAP/HTTP/HTTPS and cXML (Ariba) support will be part of the next product release in December. SOAP integration includes support for both RPC and document styles and uses the mediation/proxy support functionality found within the IBM Web Services Gateway to coordinate and manage internal/external service endpoints.

Examining the product from a document perspective (see Figure 1), there are five basic processing layers that control document flow. Inbound documents (external communication) are handled via a network/transport management layer. Following document receipt, additional message format processing is performed, which includes message format–specific processing, packaging/depackaging, and/or content validation. WebSphere Business Integration Connect allows for optional validation (using XSD or DTD instances) and document normalization (via XSLT) integration. After validation and normalization, the document can be converted into another message format. As the final step, the document is made available for delivery into the hub via HTTP/HTTPS, JMS, or file-based integration. The end-to-

end management of the software platform is provided by the event/alert notification layer and the management console, which provides transaction visibility to the hub as well as to trading partner participants.

From an architecture perspective, WebSphere Business Integration Connect consists of three major components:
- *Receiver (REC):* The receiver provides the inbound message listener service and initiates the document processing flow.
- *Document Manager (DOC):* The document manager is the main processing engine and is responsible for document routing, state management, message format packaging, delivery management, and logging, as well as validation and normalization functions.
- *Community Console (CON):* The console application is a Web-based J2EE application for the operational management of the community hub.

All three components run in separate WebSphere Application Server instances providing container management, as shown in Figure 2. The use of WebSphere Application Server as the foundation for the product enables WebSphere Business Integration Connect components to be deployed on separate servers. This architectural feature facilitates deployments to meet high-end performance requirements as well as support for custom high-availability/failover requirements.

The decoupled architecture is supported through components' interactions with the database, shared file storage, and the messaging system. The database holds the community metadata, validation documents (known as "guidelines"), message history (nonrepudiation), process state management history, and system/document exchange event data. The shared file service is responsible for the physical storage and management of the actual documents. The messaging system provides the event service that enables components to communicate with each other.

As a result of this component architecture, a receiver never talks directly to a document manager. This decoupled architecture allows an organization to establish multiple receivers and/or multiple document managers depending on load and performance requirements. The ability to implement distributed topologies provides architectural flexibility and allows the product to be configured to support complex, multilayered extranet/DMZ environments. Figure 3 shows a possible configuration using a split-server topology.

Document security is enabled via encryption/decryption of payloads utilizing PKI standards to secure and validate the authenticity of documents. From a document exchange perspective, WebSphere Business Integration Connect enables the hub to restrict document interactions and SOAP operations to specific trading partners.

From an external access perspective, transport-layer security provides both server-based and client-based certificate authentication at the level of the receiver. Internally, the security integration within WebSphere Business Integration Connect utilizes an access control permission model to configure and enforce user/developer access rights within the Community Console. Partner administrators registered into the system can create user accounts for their organizations as well as groups to pro-

"Community integration provides a foundation for scaling trading communities into the hundreds and thousands of participants"

vide differential access to specific target roles.

Support for high-availability solutions is fully enabled. Support for highly available receiver and console components requires network dispatch services. Document managers fully support placement on different machines without the need for additional clustering software. Cluster Fault-Tolerant Solutions (e.g., Linux Red Hat and SUSE, AIX/HACMP, Sun Clusters, and Windows solutions) can be used as well to support high availability. Figure 4 shows the pod-based solution deployment architecture, which can be used to support active-active configurations to support high-availability functions as well as load balancing.

WebSphere Business Integration Connect enables retry processing at the transport and the process levels. For a transport-level retry, the hub will attempt to send a document a configurable a number of times with a configurable retry interval between attempts. Failure to complete the message delivery within this set range of attempts results in quiescencing of the outbound gateway and persisting of the messages to a configured gateway message queue.

When the gateway is activated, documents flow in a FIFO order from the gateway message queue. Process-level retries occur according to the specific business protocol (such as RosettaNet or AS2) and are configurable by partner and process. Based on a defined number of attempts and a defined retry interval, process retries continue until the message is acknowledged and/or appropriately responded to per the protocol. In the event of a transport- or process-level retry, hub and participants are notified via the alert management facility.

WebSphere Business Integration Connect provides integration with enterprise applications via JMS, HTTP and HTTP/S POST, SOAP, and file-based interactions. It is optimized to work with WebSphere Business Integration or WebSphere Application Server offerings but is flexible to work with other middleware products as well.

For example, a new customer order could trigger an event via a WebSphere Business Integration Adapter and the subsequent routing to the WebSphere Business Integration Server. The WebSphere Business Integration Server hosts business processes that drive integration with other enterprise applications as well as interactions with external trading partners. The customer order could, for example, result in an inventory shortage and an automated interaction with a partner might be needed to provide an "available-to-promise" date to the customer. During WebSphere Business Integration Server processing, the integration process can issue a request/reply call to the supplier via WebSphere Business Integration Connect.

WebSphere Business Integration Connect also provides a complete solution for EDI integration. Using WebSphere Business Integration Connect for transport integration (e.g., AS2 and FTP), traditional EDI processing can be enabled via WebSphere MQ integration with WebSphere Data Interchange or via file-based/message-based integration with alternative existing legacy EDI transformation packages, e.g., Harbinger/Inovis, Gentran, and others.

Internal integration packaging can be invoked with or without transport-level attributes. Transport-level attributes are used to support additional automated routing of the document. Both HTTP and JMS attributes are formatted/managed as x-aux fields (defined in XML) for message/transport information, e.g., sender_id, receiver_id, process version, protocol,
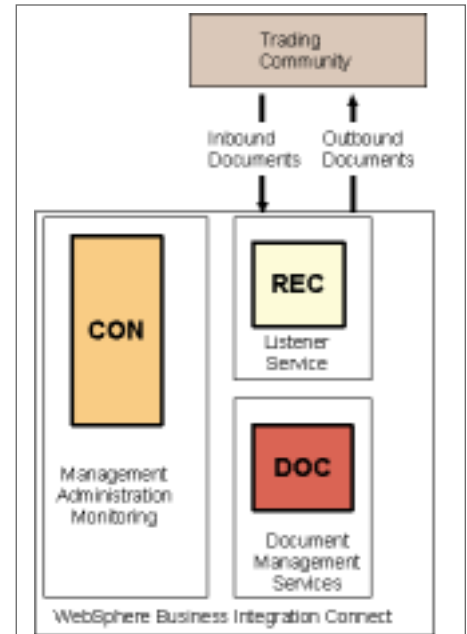

FIG. 2: THE THREE MAJOR COMPONENTS OF WEBSPHERE BUSINESS INTEGRATION CONNECT RUN IN SEPARATE INSTANCES OF WEBSPHERE APPLICATION SERVER
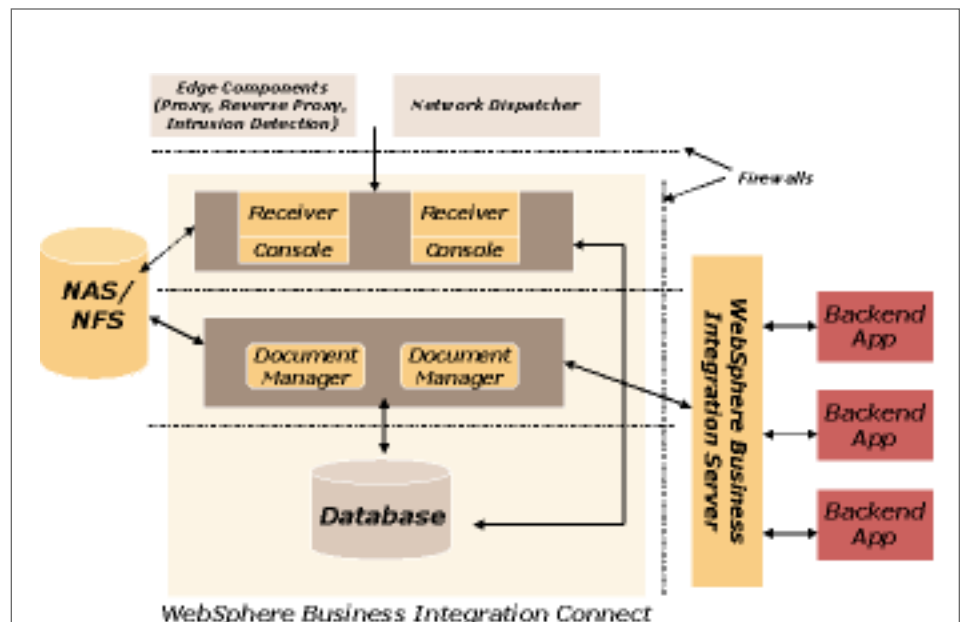

FIG. 3: A POSSIBLE CONFIGURATION USING A SPLIT-SERVER TOPOLOGY

WEBSPHERE DATA INTERCHANGE

In 2001, more than $2 trillion U.S. in transactions were traded via traditional EDI architectures. EDI is established in 95% of Fortune 500 companies, and many of these enterprises have been reluctant to extend this EDI solution into XML-based implementations. As a result, there is a need for a robust EDI interchange solution that is parallel to the WebSphere Business Integration Connect technology. EDI integration is enabled through the use of WebSphere Business Integration Connect to provide a transport architecture and WebSphere Data Interchange to provide the EDI transformation solution. WebSphere Data Interchange is a data translation application supporting EDI standards (e.g., X12, EDIFACT, UCS, VICS, Tradacoms, and others) and XML formats. It operates on multiple platforms, including Windows 2000, AIX, and z/OS. The main components of WebSphere Data Interchange include:

- WebSphere Data Interchange Client, which runs on Windows 2000. This is used to create and deploy maps, import EDI standards, and add trading partner EDI profiles.
- The Data Translation application (the WDI "engine"), which transforms application data to EDI (ANSI X12, EDIFACT), XML, and non-EDI formats (user data) providing an any-to-any mapping solution. The Data Translation engine can run on Windows, OS/390, z/OS, or AIX.
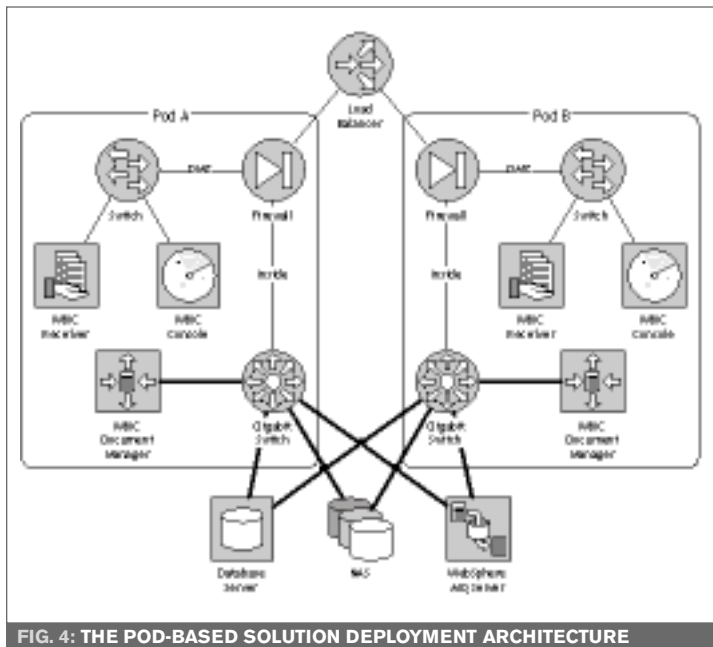
FIG. 4: THE POD-BASED SOLUTION DEPLOYMENT ARCHITECTURE

FIG. 5: THE WEB-BASED COMMUNITY CONSOLE

instance_id, and other fields. The WebSphere Business Integration Message Broker provides direct integration to message queues. Integration with WebSphere Interchange Server requires the WebSphere JMS Adapter v2.3 (or higher) to format/parse the attribute information.

WebSphere Business Integration Connect ships with an extensive set of predefined exception events to enable alerts to be automatically raised during hub operations and notifications to be sent to multiple parties at both the hub and participant levels. From a community integration perspective, the Community Console provides a Web-based interface to provide community visibility into the event history and document exchange interactions.

The Community Console gives hub managers an aggregated view of the activities of the community, allowing organizations to pinpoint "weak spots" in the community. Through this Web-based interface, community participants are provided a secure, searchable, Web-based view of their specific exchange activities, and the ability to update partner-specific information, including profile information as well as participant users and group metadata. The Community Console also provides the interface for the administration of the hub, including creation and maintenance of partner profiles, certificate management, and hub administration/configuration tasks. Figure 5 shows a screenshot of the Web-based Community Console.

Conclusion

Interenterprise integration requirements are fluid and dynamic, which imposes unique requirements on the implementation and operation of trading ecosystems. The need to manage these architectures more proactively is positioning community integration solutions as the core requirement for building trading communities. By providing event and transactional visibility to community managers and participants, trading community benefits can be realized more rapidly.

WebSphere Business Integration Connect enables community integration regardless of transport, data format, or trust model. With the WebSphere Business Integration Connect technology, customers can scale from a small trading community consisting of 3–5 partners to a complete trading community consisting of tens, hundreds, and thousands of trading partners based on a WebSphere Application Server foundation. The product also leverages JMS for WebSphere Business Integration connectivity and offers HTTP as well as file-based integration solutions. WebSphere Business Integration Connect provides a robust solution for building interenterprise solution architectures.

Acknowledgments
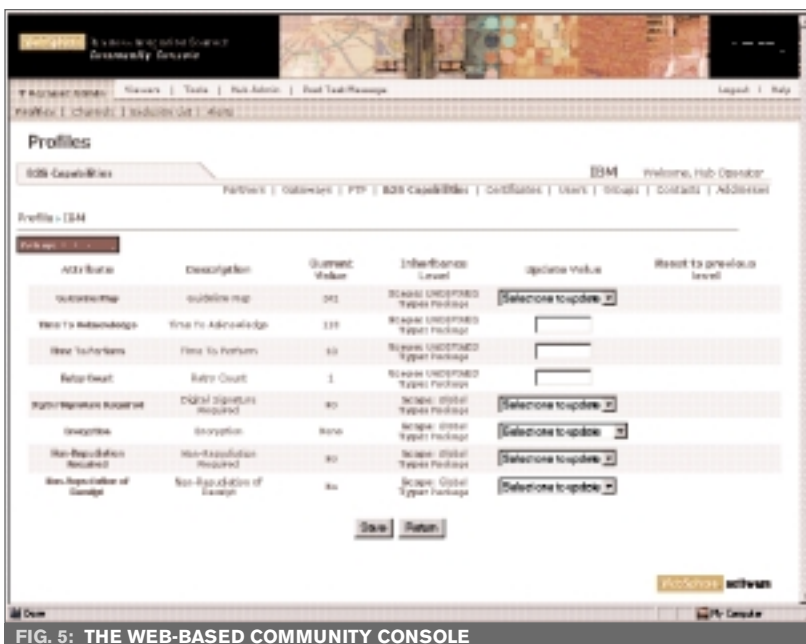
Cyanea™

What's Obstructing Your Application Flow?

With Cyanea/One™, your company will gain valuable insight into its J2EE applications. With a few mouse clicks, you can quickly drill down into your applications, all the way down to the method level. Isolate, diagnose and proactively resolve bottlenecks and resource consumption issues. Accomplish all these without touching your code or even requiring knowledge of your application.

...Finding Methods in the Madness.

Find methods at:
www.cyanea.com/wsdj/nomoremadness.html
1-877-CYANEA8 | sales@cyanea.com

Seeing your systems through a hacker's eyes

Attack Trees: It's a Jungle Out There

BY MICHAEL S. **PALLOS**

Computer security is an important aspect of any IT architecture. The requirement for security vigilance is especially critical, given the widespread availability of technology that potentially enables novice hackers to penetrate corporate IT defenses simply by using a tool available on the Internet.

ABOUT THE AUTHOR
Michael S. Pallos, is a senior solution architect for Candle Corp. (www.candle.com) with 19 years' experience in the IT industry. He is a consultant to some of Candle's largest corporate customers and a featured speaker at industry conferences. Michael is also an IBM-certified e-business designer, e-business technologist, and WebSphere specialist.

E-MAIL
michael_pallos
@candle.com

Attacks can range from the theft of credit card or other sensitive customer information to embarrassing defacements of corporate Web sites. The ramifications of breached enterprises can negatively affect corporate valuation and brand equity, as well as customer and partner relations, not to mention the legal liabilities for security breaches.

Figure 1 outlines the types of malicious tools available and illustrates how their availability has lowered the bar for would-be hackers. The chart is presented by David Dittrich, senior security engineer, University Computing Services, University of Washington, and is based on information gathered by the Computer Emergency Response Team (CERT) Coordination Center (www.cert.org).

IBM WebSphere security at the application level requires a functional assessment as the application is being developed. IBM Redbooks, such as *IBM WebSphere V5.0 Security,* offer best practices for securing important WebSphere components, which include the Web server, application server, servlets, and Enterprise JavaBeans (EJBs). Another excellent cross-platform Redbook displaying how to integrate and secure the enter-

prise is *WebSphere MQ Security in an Enterprise Environment.* Once the application is deployed, a production risk assessment is also required to evaluate application security in real-world environments.

The WebSphere universe includes multiple third-party products to strengthen application infrastructure security. WebSphere MQ security products offered by Candle Corp. and other vendors, for example, help organizations secure messages before, during, and after transportation by leveraging authentication, authorization, nonrepudiation, and encryption techniques. This allows organizations to secure the back-end WebSphere Application Server and WebSphere MQ interfaces, as well as other aspects of the WebSphere environment.

"Security," according to security expert and *Secrets and Lies* author Bruce Schneier, "is only as strong as the weakest point." Often, the weakest point does not involve technological vulnerabilities, but instead may be related to enterprise procedures or physical security. Encryption algorithms are reasonably secure, and Public Key Infrastructure (PKI) Certificates, RSA key length, and

secure phone lines are not always the target penetration points for attackers.

One organization, for example, created three levels of physical security to protect its business-critical servers. The organization, however, had connected the servers to unsecured direct dial-in lines for technical support. These unsecured phone lines could provide hackers with open access to sensitive organizational data.

One approach for securing a system is to consider an attacker's viewpoint. Examining the "How can I penetrate this system?" mindset offers a perspective into the production system that is usually not considered by architects and developers. For example, would a potential attacker be able to retrieve sensitive information such as source code or network architecture blueprints from the corporate dumpster?

Identifying Risk

Attack tree analysis, created by Schneier, quantifies the security or vulnerability of a system based on the goals of the attacker. For example, if an information systems manager were responsible for an order fulfillment system containing credit card details and related customer information, an attacker may have the goal of stealing the customer credit card data. This goal – "steal credit card data" – is the starting point, or root node, of the attack tree. The attack tree is then extended, building branches down the tree to identify the different subgoals and penetration points available to the attacker. The branching process continues as the means of penetration are decomposed, or expanded, to the lowest level of intrusion, known as the leaves.

An attack tree can represent each opportunity for an attack against a computer system. Computer systems potentially contain numerous penetration points and vulnerabilities. A

single attack tree represents each single penetration point. The consolidation of numerous attack trees is known as an attack forest.

The attack-tree approach allows analysts to rethink system vulnerabilities from the attacker's perspective. This perspective expands the software or system vulnerabilities model, as defined by I.V. Krsul in his thesis, "Software Vulnerability Analysis," to include elements other than the software application. One example is the previous example of an attacker rummaging through a dumpster searching for password information and other sensitive documents.

Attack trees are represented graphically and textually. A graphical representation is usually built with the root node, or goal, on the top. The tree then descends branches and subgoals until the leaves are finally reached at the bottom level. Figure 2 is the conceptual model of an attack tree represented in a graphical format.

The textual representation of an attack tree follows a numeric outline

structure. The root node, or goal, is represented at the first level with no indentation. Each subgoal is then numbered accordingly and indented one unit per level of decomposition. The representation below presents the textual view using the same example content found in Figure 2.

1. Goal (root node)
   1.1 Leaf 1
   1.2 Sub-Goal
      1.2.1 Leaf 2
      1.2.2 Leaf 3

When applying attack-tree logic to a production WebSphere application, you can gain insights into the potential penetration points that an attacker may leverage. You start by building the higher-level nodes, then expand downward.

Utilizing "The Twenty Most Critical Internet Security Vulnerabilities – The Experts' Consensus," developed by the SysAdmin, Audit, Network, Security (SANS) Institute, you can build an attack forest by focusing on the Unix vulnerabilities illustrated in

FIG. 1: ATTACK SOPHISTICATION VERSUS INTRUDER TECHNICAL KNOWLEDGE
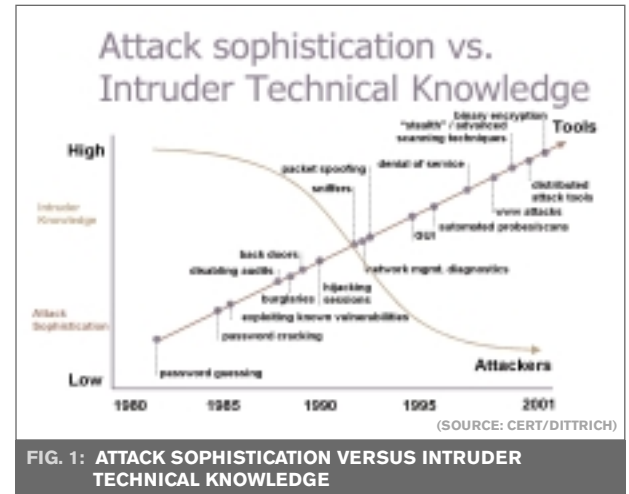
Figure 2. Each of the nodes in Figure 3 can be extended, creating a unique attack tree.

Figure 4 represents a graphical attack tree that examines one of the Unix vulnerabilities reported by SANS. According to the experts, password vulnerabilities for users, systems administrators, and applications include accounts with no passwords,
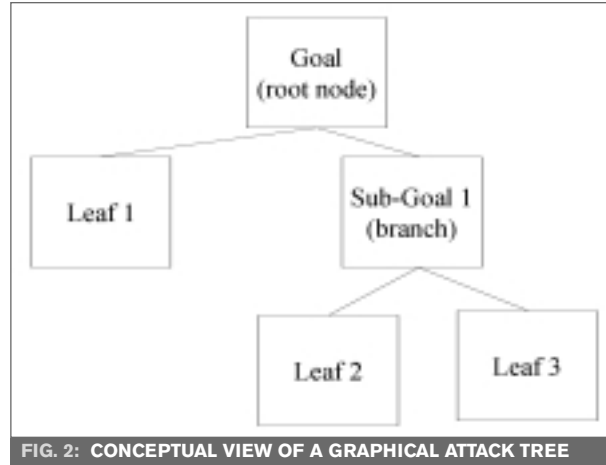
FIG. 2: CONCEPTUAL VIEW OF A GRAPHICAL ATTACK TREE
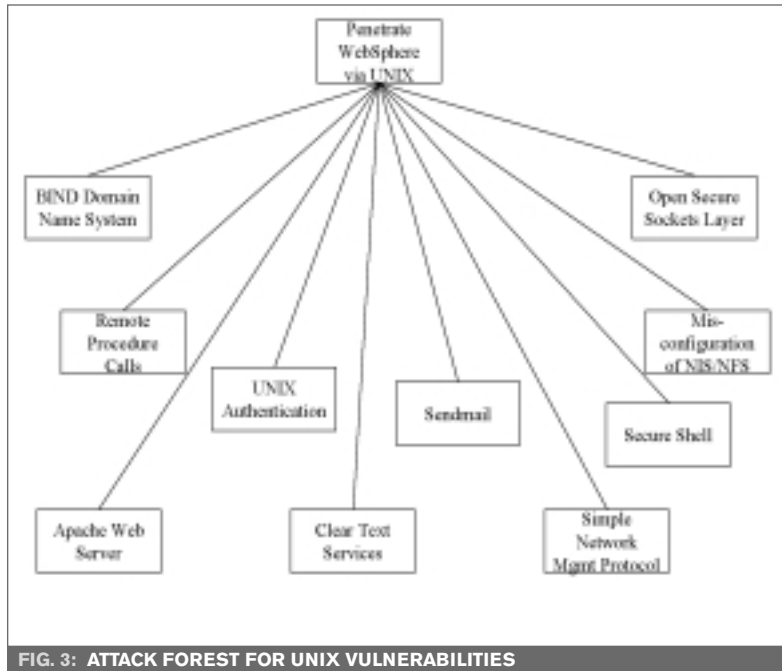
FIG. 3: ATTACK FOREST FOR UNIX VULNERABILITIES
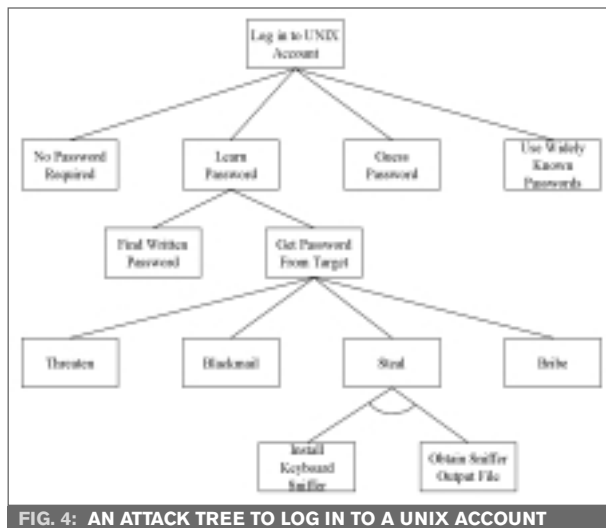
FIG. 4: AN ATTACK TREE TO LOG IN TO A UNIX ACCOUNT

weak passwords, commonly known passwords (such as your company name), and weak hash algorithms.

The attack tree in Figure 4 also aids you in considering alternative ways in which a node can be achieved. Analysts are forced to ask themselves questions from an attacker's perspective, such as "How can I steal passwords?" By taking a broader view of information security, WebSphere Application Server security expands from permissions granted on an EJB to the possibility of installing keyboard sniffers on WebSphere Application Server administrators' computers. This perspective is far different from a WebSphere developer's perspective for designing and building secure EJBs.

Comprehensive WebSphere security encompasses more than the specific WebSphere Application Server application environment. Enterprise architects, information system managers, system administrators, security experts, and WebSphere team members must consider additional aspects of vulnerabilities and penetration points that computer attackers can exploit outside of the WebSphere framework. Attack tree analysis offers a systematic methodology for identifying penetration points and system vulnerabilities not considered from the application design perspective.

Resources:
- Davies, S., et al. (2003). *WebSphere MQ Security in an Enterprise Environment:* www.redbooks. ibm.com/redbooks/SG246814. html
- Kovari, P., et al. (2003). *IBM WebSphere V5.0 Security:* www.red books.ibm.com/redbooks/ SG246573.html
- *Software Vulnerability Analysis:* www.acis.ufl.edu/~ivan/articles/m ain.pdf
- *Attack Modeling for Information Security and Survivability:* www.sei.cmu.edu/publications/do cuments/01.reports/01tn001.html
- *The Twenty Most Critical Internet Security Vulnerabilities:* www.sans.org/top20
- Schneier, B. (2000). *Secrets & Lies: Digital Security in a Networked World.* John Wiley & Sons.
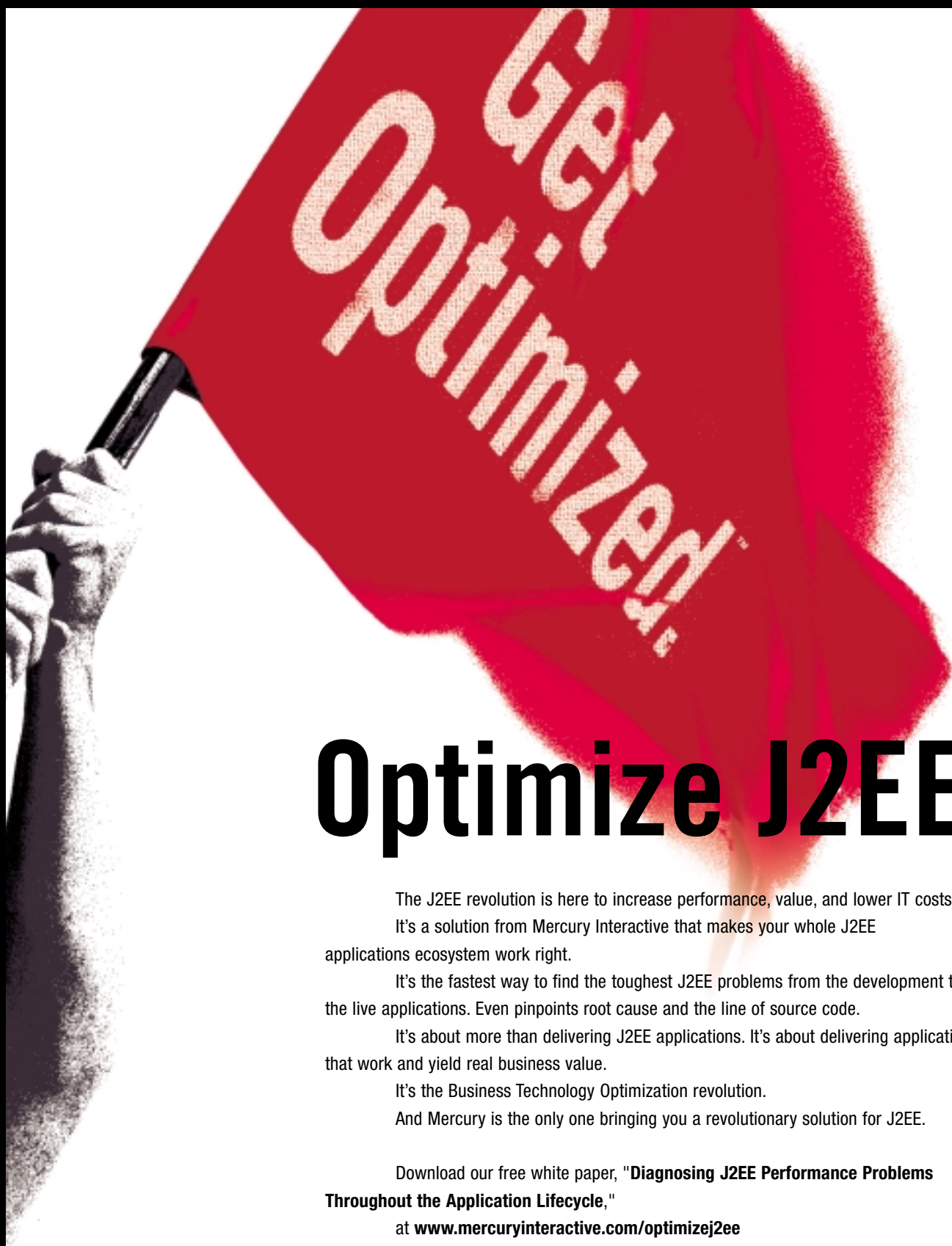
Optimize J2EE.

The J2EE revolution is here to increase performance, value, and lower IT costs.

It's a solution from Mercury Interactive that makes your whole J2EE applications ecosystem work right.

It's the fastest way to find the toughest J2EE problems from the development to the live applications. Even pinpoints root cause and the line of source code.

It's about more than delivering J2EE applications. It's about delivering applications that work and yield real business value.

It's the Business Technology Optimization revolution.

And Mercury is the only one bringing you a revolutionary solution for J2EE.

Download our free white paper, "**Diagnosing J2EE Performance Problems Throughout the Application Lifecycle**,"
at **www.mercuryinteractive.com/optimizej2ee**
Get Optimized.™

MERCURY INTERACTIVE

Developing Java Client Applications Using Java Web Start and WebSphere Studio

Using JWS avoids the problems associated with traditional applets

BY JOE **WINCHESTER**,
GILI **MENDEL**,
& GUNTURI **SRIMANTH**

ABOUT THE AUTHOR

Joe Winchester,
Desktop Java Editor
of Java Developer's
Journal, is a software
developer working on
WebSphere develop-
ment tools for IBM in
Hursley, UK.

E-MAIL
winchest@uk.ibm.com

ABOUT THE AUTHOR

Dr. Gili Mendel is a
technical lead for
IBM's Visual Editor
for Java in Raleigh,
NC.

E-MAIL
gmendel@us.ibm.com

Java Web Start (JWS) was created as part of JSR 56 and is included with JRE 1.4. The idea was to provide a way to distribute a Java application that would run in a JVM on the client, but avoid the problems associated with traditional applets. JWS does this by incorporating the features shown in Table 1.

I f you have JRE 1.4.1 or higher, then you already have Java Web Start installed. If not, you can obtain and install a JRE from http://java.sun.com/j2se/down loads.html. This article describes how to create and deploy a JWS application using WebSphere Studio Application Developer 5.0.

Hello World JWS

In keeping with the tradition of articles that introduce new ways to perform GUI programming, the first step is to create a Web Start application that shows a dialog with the words "Hello World". This is done in two steps, the first on the server side, where the class is created and deployed onto a Web server, and the second on the client side, where a PC accesses a URL that causes it to be down-loaded and invoked locally through JWS.

THE SERVER

1. The HelloWorld class is created, packaged into a JAR, and imported into a dynamic Web project.
2. A file with the suffix .jnlp is created that contains XML tags describing the application to be run.
3. A WebSphere server configuration is created that has the .jnlp file extension associated with the MIME type to cause the browser to run JWS.

CREATING THE HELLOWORLD CLASS

The source code for the HelloWorld class is shown below.

```
import javax.swing.*;
public class HelloWorld {
    public static void main(String[] args) {
        JFrame frame = new JFrame("Hello World");
```

```
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(200,100);
        frame.show();
    }
}
```

When executed using RunAs>JavaApplication, a Swing JFrame is displayed whose title is Hello World, as shown in Figure 1.

Running an application using RunAs>Application allows you to test the application; however, it does so by creating a JVM on the same machine as WebSphere Studio and running the program there. For JWS the next steps are to create a JNLP file and deploy the HelloWorld class into an instance of the WebSphere Application Server, so that it can be run th rough a browser connection on another PC.

A dynamic Web project should be created for the pur-pose of creating and deploying the JWS application. This is done using the menu option from the File menu or New pulldown list on the toolbar.

We'll call the example project "JavaWebStartTest". The HelloWorld class should exist in the Web project within a JAR file. This is done by using the Export pop-up menu option for the HelloWorld.java file, selecting JAR file, and specifying a location in the file system. Having done this, now import the JAR into the JavaWebStartTest project using the menu option File>Import.

Note: If you change HelloWorld.java and wish to re-deploy it to the Web project, you will need to re-export it as a JAR and import it back into the JavaWebStartTest project. To simplify this scenario, you can download the zipCreation utility created by Tim DeBoer, which allows you to configure a project to be automatically re-JARred into another project www7b.boulder.ibm.com/wsdd/library/techarticles/0112_deboer/deboer2.html.

CREATING THE JNLP FILE

The next step is to create the JNLP file that will be used to download and launch the HelloWorld class. This will be called HelloWorld.jnlp, and placed in the WebContent directory of the project. There is no specific wizard for JNLP files, so use the File>New>Other and select Simple>File to specify the name HelloWorld.jnlp. A JNLP file is XML and contains a number of tags, as shown in Listing 1. The JNLP tag specifies the codebase where the code that is going to be run is located. Listing 1 hard-codes

it to be the URL of the Web server (which happens to be our IP address), the port number, and the name of the Web project.

Later we will show you how to use a servlet to avoid having to hard-code the Web server URL, but for simple examples it can be explicitly set. The information tag specifies details such as the title of the application, the vendor, and some descriptions that will be shown to users when they download and invoke the application, and that are also used by JWS administration. The tag <offline-allowed> specifies that once the application has been downloaded by JWS and cached locally, users can relaunch it from their client without the need for a connection. Including the tag <offline-allowed=false> would prevent users from relaunching without a connection. For the application to run on the client, we need to specify the JRE level required for it. In our case we need 1.3 or higher, and the JAR file containing the application must be specified in the <resources> tag. The name of the Java class that will be run through its main(String[] args) method is entered in the <application-desc> tag. We'll go into more detail about this tag later.

The location of the JAR file in the <resources> tag is relative to the codebase specified in the <jnlp> tag. The full syntax of a JNLP file can be found at http://java.sun.com/j2se/1.4.2/docs/guide/jws/developersguide/syntax.html. In this article we will cover the most commonly used tags.

Listing 2 shows the code for an index.html page with an anchor tag pointing to the JNLP file, which allows the user to click the link on a browser and start HelloWorld through JWS.

The locations of the three files in the Web project are shown in Figure 2. Because the HelloWorld.jnlp file is in the WebContent folder of the JavaWebStartTest project, the codebase of the deployed EAR is going to be ServerURL /JavaWebStartTest. For the WebSphere test environment the server URL is the machine name:port number.

CONFIGURING WEBSPHERE

To configure WebSphere, the file extension .jnlp must be associated with the application MIME type application/x-java-jnlp-file. To do this, select the server in the Servers view and edit its configuration using the Open option on its pop-up menu. Select the Web tab, and use the Add button to enter the MIME type for JNLP extension as shown in Figure 3. If you don't have a server already configured, use the menu option New>Server and Server Configuration from the pop-up menu in the Servers view, selecting WebSphere version 5.0 and Test Environment, as shown in Figure 3.

THE CLIENT

To test the HelloWorld application, run the index.html file on the test environment by right-clicking on the file in the Project Navigator and selecting Run on Server from the pop-up menu. If there is more than one server configured, make sure that the one with the MIME type extension for JNLP is the one launched. Once the test environment has started up, select the anchor tag in the index.html file. This will download the HelloWorld.jnlp file to the client browser. In Figure 4 the browser embedded within WebSphere Studio is used, so the same PC is acting as the server (running the WebSphere Studio test environment) and the

client (running the Web browser).

After you've tested the JNLP file this way, a good next step is to go to a PC other than the one running the WebSphere test environment, and try to access the JNLP file from there (typing in the full URL rather than localhost, e.g., http://9.20.217.67:9080/JavaWebStartTest/index.html). Having downloaded the application, JWS will show details about the title and vendor of the application, together with a progress bar as it downloads the JAR file, caches it, and executes it.

FIG. 2: LOCATIONS OF JAR, JNLP, AND
INDEX PAGE IN A WEB PROJECT

Because of the HTTP attribute in the <jnlp> tag, the HelloWorld application will be installed in the JWS Application Manager. This allows a shortcut to the program to be automatically installed onto the desktop, and the user is prompted to choose whether or not to create the shortcut, as shown in Figure 5.

Having downloaded the JAR file, the HelloWorld class is then launched. This application is run in a window that identifies itself to the user as a Java Application Window, as shown in Figure 6. This is so that Trojan horse applications masquerading as other programs can be distinguished as having come from JWS by the status line message at the bottom. If the JNLP file was using a signed JAR with a certificate, then the status message is not shown. I'll go into more detail on security later.

Administering the Client

Once the client PC has downloaded and run the HelloWorld application, a local copy is kept. You might notice that subsequent attempts to run the application are faster, and also that you won't get the same sequence of launch dialogs; instead you get just a single splash screen.

The Application Manager shown in Figure 7 allows a client PC to administer all of its JWS applications. The program javaws.exe is located in the javaws directory beneath the JRE, e.g., "C:\ProgramFiles\Java\j2re1.4.2\javaws\javaws.exe". You can also obtain it from the following URL, which is itself a JWS-enabled version of the application manager, http://java.sun.com/docs/books/tutorial/information/player.jnlp.

Because the href="HelloWorld.jnlp" attribute was specified in the <jnlp> tag, the program is listed in the application manager. If no href attribute is used, the client application is not included in the application manager, and the user will not be prompted to create a shortcut. The icon used is the one specified in the <icon href=""> tag in the JNLP file, and as none was specified in HelloWorld.jnlp, the default coffee cup is shown. To install a shortcut to the application on the desktop, use the menu option Application>Create shortcuts, which will place a desktop icon so you can launch HelloWorld directly without going through a Web browser. The icon used is the one specified in the <icon href=""/> tag in the JNLP file, which defaults to the Java coffee cup logo.

You can also use the menu option Application>Remove shortcuts to take a shortcut away, and the menu option Application>Remove application to delete the locally

ABOUT THE AUTHOR

Gunturi Srimanth is a software developer working on IBM's Visual Editor for Java in Research Triangle Park Lab, NC.
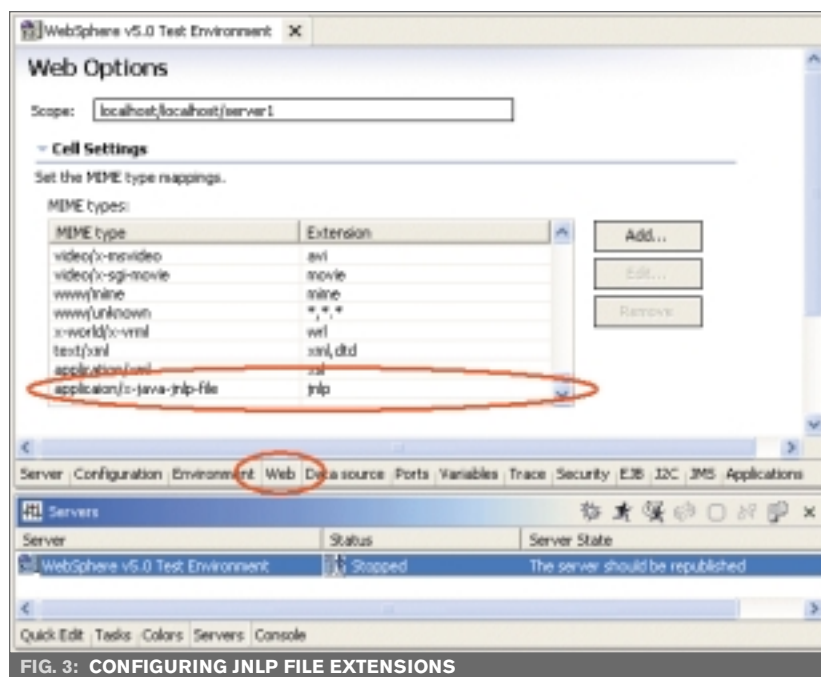
E-MAIL
sgunturi@us.ibm.com

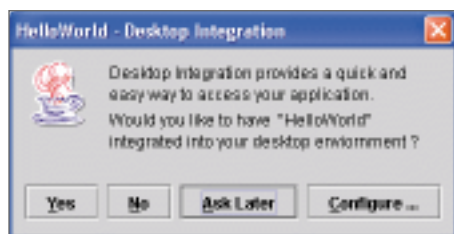**FIG. 3:** CONFIGURING JNLP FILE EXTENSIONS

FIG. 4: HELLO WORLD LAUNCHED THROUGH JWS
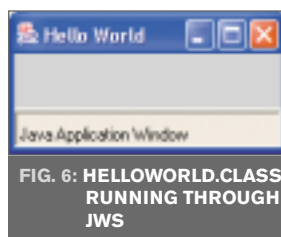

FIG. 5: PROMPT FOR DESKTOP INTEGRATION

FIG. 6: HELLOWORLD.CLASS RUNNING THROUGH JWS

installed by default, but in an environment like the Internet it might not be fair to assume that all of the clients are going to be at a particular JRE level. To deal with the inherent uncertainty of Internet environments, a piece of JavaScript can be written to detect whether or not JWS is installed, and if not, prompt the user to download and install it. A description of the scripts to do this is in the JWS documentation at http://java.sun.com/ j2se/1.4.2/docs/guide/jws/developers guide/launch.html#intro. Listing 3 shows how the index.html file can be rewritten to perform automatic detection of whether the client has the correct JRE level and JWS installed, and if not, prompt them to download a version. The first step is to see whether the browser is Netscape or IE, and if Netscape, inspect the MIME types to see whether JWS is installed. The second piece of VBScript runs on IE, and by using a COM object that is part of the Web Start DLL, it tries to detect what version, if any, of JWS is present. The final piece of JavaScript either links the page to the JNLP file, or else prompts the user with a link to download and obtain JRE 1.4.2 from Sun's Web site. The file indexTestingJreLevel.html includes all of the HTML and JavaScript shown in Listing 3 and is part of the downloadable EAR file included with this article. (All of the code referenced in this article can be downloaded from www.sys-con.com/websphere/sourcec.cfm.)

Security

Some JWS applications may be required to access services on the client PC such as IO for reading and writing files or opening sockets and making connections. For security purposes you don't want malicious programs to have unlimited access to these facilities on your PC, so JWS uses the Java security model and JAR file certification. By default a JWS virtual machine is a restricted sandbox denying access to these services. If you try to go outside the permissions of the security manager, e.g., to perform some file IO, then an exception is raised.

To be able to use restricted Java services you must ask the client to grant access. This is done with the <security> tag in the JNLP file, which is a child of the <jnlp> tag. If the program is to have unrestricted access to the machine resources, the tag would be:

```
<security>
    <all-permissions/>
</security>
```

The JAR file must be signed in order for security to be used; failure to sign the JAR file will result in a security error being raised. To sign a JAR file you can use the key-tool command that is supplied with the JDK, although if you are going to put your JWS application into full production it is recommended that you obtain a certificate from an authority such as Thawte or VeriSign. For more information on Java security, see http://java.sun.com/docs/

installed program. To control whether or not JWS applications are automatically installed as desktop shortcuts, you can use the menu option File> Preferences and open the Shortcuts tab. The preferences dialog also lets you configure details such as whether you are connecting through a proxy server.

Automatic Detection of Client Java Level

The index.html file we created earlier assumed that the client browser had JWS installed. If the client PC has JRE 1.4.1 or higher, then JWS should be

Improving Deployment and Configuration

In the steps described above there were two steps that can be simplified with a simple servlet. They are the inclusion of the explicit URL for the codebase in the JNLP file, and the changing of the MIME type associations for the WebSphere test environment.

To automate these steps, obtain the jnlp-servlet.jar file and add it to the WEB-INF/lib folder of the Web project. The file is included with the JNLP developer's pack, available at http://java.sun.com/products/javawebstart/download-jnlp.html.

Then update the web.xml file in the WEB-INF folder as shown in Listing 4. XML tags for <servlet>, <servlet-mapping>, and <mime-mapping> should be inserted inside the <web-app> tag. These tags should be placed before the <welcome-file-list> tag, and WebSphere Studio validation of the web.xml file will report errors if these tags are incorrectly positioned.

The <servlet> tag tells WebSphere Studio how to invoke the servlet class com.sun.javaws.servlet.JnlpDownloadServlet, which is part of the jnlp-servlet.jar. The <servlet-mapping> tag tells the container when to invoke the servlet, and that we wish it to run for files with a suffix of .jnlp. Finally, the <mime-mapping> tag allows you to specify the application MIME type to associate with the JNLP file without having to change the configuration of WebSphere. This allows you to more easily create a packaged EAR that can be deployed into an existing server that contains all the information required to run within it, and doesn't require the server administrator to update its configuration.

Now that you have included the jnlp-server.jar file in the /lib directory and updated the web.xml file to specify the MIME type and JnlpDownloadServer, the JNLP file itself can be simplified. When we created HelloWorld.jnlp earlier, we hard-coded the explicit URL of the Web server in its JNLP tag. What the JnlpDownloadServlet does, however, is run over all files with the .jnlp extension and allow this information to be substituted. Therefore, the phrase "$$codebase" can be soft-coded into the JNLP file and it will become the codebase of wherever the JNLP file is deployed, as shown below:

```xml
<?xml spec="1.0+" version="1.0" encoding="utf-8"?>
<jnlp codebase=$$codebase>
     ... remainder of jnlp file
</jnlp>
```

Also, because the <mime-mapping> tag has been set in the web.xml file, the explicit association of the .jnlp file suffix with the application in the Web tag of the server configuration can be removed.

Client Detection of Changes

The client PC caches the application JAR file, as well as the date and time stamp of the original JNLP file, in a folder specified on the Advanced tab of the Application Manager.

When an application is launched using JWS, it first checks the cache to see whether it has accessed the JNLP file with the specified name from the same URL. If so, it then checks the date and time stamp of the JNLP file on the server against the one stored in the cache. A match means that the local version is not stale, and JWS will use it to launch the application without re-downloading the JAR files. If the date and time stamp don't match, then JWS will download the latest version from the location specified in the JNLP file, recache it, and continue launching the program. This approach, in which JWS lazily pulls the new version of software from the server when it is required, allows you to administer changes in a single

DEPLOYMENT	Running a JWS application involves selecting an HTML link and is little different from opening a new page. The difference is that the Web server page is a JNLP file that launches the JWS plug-in that downloads and executes the application locally on the client.
CACHING	Once the application has been downloaded it is cached on the client PC. Each time the application is relaunched, JWS checks the server to make sure there is no more recent version to download. If not, then the local version is run (avoiding the need for a re-download), and if a new version is available it is automatically retrieved so the administrator never has to worry about stale clients connecting to an updated server program API.
INDEPENDENT OF BROWSER JRE	JWS allows you to specify precisely which client JRE level you want it to run in, independently of how the browser has been configured. This avoids the problems associated with applet tags and browser JRE levels (such as with Internet Explorer). Unlike applet tags that share a single JVM instance for the browser, each JWS application has its own independent JVM and is launched through its main(String[] args) method.
SECURITY AUTHENTICATION	The Java security model is used to authenticate the source of a JWS application and to restrict access to vulnerable system services such as IO or communications. Explicit access to these must be given to each JWS program by certifying the signed JAR file.
DESKTOP INTEGRATION	Once a JWS application has been downloaded from the JNLP file it can be integrated onto the desktop as a stand-alone icon. The application can then be relaunched without the need for a Web browser.

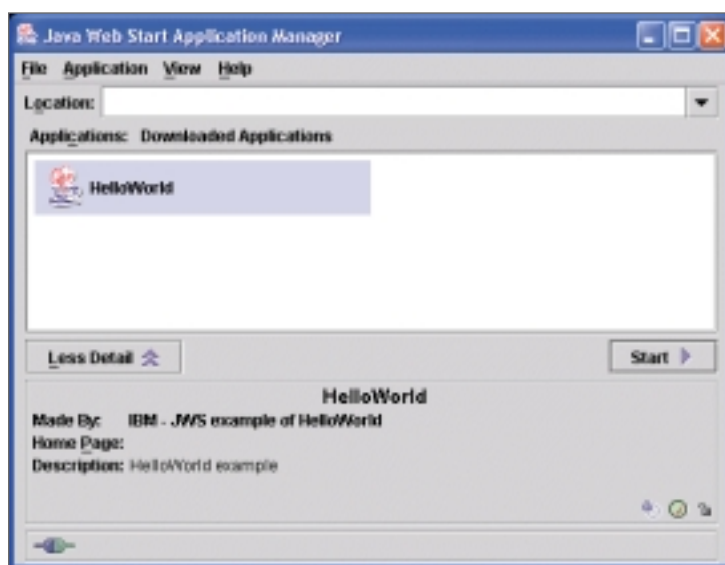TABLE 1: JWS FEATURES THAT AVOID PROBLEMS ASSOCIATED WITH TRADITIONAL APPLETS

FIG. 7: THE JWS APPLICATION MANAGER

location while ensuring that clients will be automatically updated.

One problem with this approach – in which the date and time stamp of the JNLP file are used as a trigger to force clients to redownload a version – is that the same JNLP file may be replicated onto several servers for load balancing. In this case, the client will detect that the file has changed and download a new version of the program when the cached version would have sufficed. To avoid this it is possible to have an explicit time stamp included in the JNLP file by starting the first line with TS: followed by an ISO 8601 date and time stamp. For example, if you wanted to time-stamp your file to be August 13th at 3 p.m. and 30 seconds, a line would be inserted before the opening <?xml> tag, as shown in Listing 5.

The JnlpDownloadServlet is responsible for recognizing the line beginning with TS and replacing the date and time stamp in the response returned to the client, so for this approach to work, the steps outlined earlier to use the jnlp-servlet.jar must have been taken.

With the TS: line included, the JNLP file can be moved between servers and JWS on the client will always pick up the encoded date and time stamp as the application's signature. This means that as the provider of the application you are responsible for explicitly changing the date and time stamp when you wish to deploy a new version of the program, but it has the advantage that the file can be moved across servers without clients falsely detecting that it has changed.

Conclusion

This article has shown how Java Web Start can be used to distribute rich Java applications from within WebSphere to be run on client PCs. This is done through the use of JNLP files that specify information about the application. This article has shown how these are cached on the client PC and can be installed as desktop shortcuts. When the application needs to be updated, it is changed centrally in a single location on the server and all clients will fetch the new copy when they next connect, based on the JNLP file date and time stamp.

The article also showed how the JnlpDownloadServlet can be used to softcode the codebase of the JNLP file and to give an explicit date and time stamp, thereby allowing the JNLP file to be replicated across multiple copies of WebSphere in a distributed environment but still appear to the client as a single program. I hope that you find JWS to be a powerful and flexible way to administer true rich Java client applications, and I welcome all feedback. ⊕

Resources
- *Java Web Start Technology:* http://java.sun.com/products/javawebstart
- *Java Web Start demos, including SwingSet and simple draw program:* http://java.sun.com/products/javawebstart/demos.html
- *The Saturn Times: Daily Web Start News From Around The Globe:* http://lopica.sourceforge.net/times
- *Deploy an SWT application using Java Web Start:* www-106.ibm.com/developerworks/opensource/library/os-jws

LISTING 1: JNLP FOR HELLOWORLD

```
<?xml version="1.0" encoding="utf-8"?>
  <jnlp spec="1.0+" codebase="http://9.20.217.67:
    9080/JavaWebStartTest/" href="HelloWorld.jnlp">
   <information>
    <title>HelloWorld</title>
    <vendor>IBM - JWS example of HelloWorld</vendor>
    <description>HelloWorld - Example of JWS</descrip
     tion>
    <description kind="short">HelloWorld
     example</description>
    <offline-allowed/>
   </information>
   <resources>
      <j2se version="1.3+"/>
       <jar href="HelloWorld.jar"/>
   </resources>
   <application-desc main-class="HelloWorld"/>
   </jnlp>
```

LISTING 2: HTMP FOR INDEX.HTML LINKING TO HELLOWORLD.JNLP

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
  Transitional//EN">
<HTML>
    <HEAD>
        <TITLE>index.html</TITLE>
    </HEAD>
    <BODY>
      <A href="HelloWorld.jnlp">Press here to launch Hello
        World using JWS</A>
```

```
    </BODY>
</HTML>
```

LISTING 3: AUTOMATIC DETECTION OF JRE LEVEL BY JAVA AND VB SCRIPT

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
    Transitional//EN">
<HTML>
    <HEAD>
        <TITLE>Index testing JRE level</TITLE>
    </HEAD>
<BODY>
<SCRIPT LANGUAGE="JavaScript">
var javawsInstalled = 0;
var javaws12Installed = 0;
var javaws142Installed=0;
isIE = "false";
if (navigator.mimeTypes && navigator.mimeTypes.length) {
    x = navigator.mimeTypes['application/x-java-jnlp-file'];
    if (x) {
        javawsInstalled = 1;
        javaws12Installed=1;
        javaws142Installed=1;
    }
}
else {
    isIE = "true";
}
</SCRIPT>
```

```
<SCRIPT LANGUAGE="VBScript">
on error resume next
If isIE = "true" Then
  If Not(IsObject(CreateOb
  ject("JavaWebStart.isInstalled")))
   Then
     javawsInstalled = 0
  Else
     javawsInstalled = 1
  End If
  If Not(IsObject(CreateOb
  ject("JavaWebStart.isInstalled.2")))
  Then
     javaws12Installed = 0
  Else
     javaws12Installed = 1
  End If
  If Not(IsObject(CreateOb
ject("JavaWebStart.isInstalled.1.4.2.0"
))) Then
     javaws142Installed = 0
  Else
     javaws142Installed = 1
  End If
End If
</SCRIPT>
<SCRIPT LANGUAGE="JavaScript">
if (javawsInstalled ||
(navigator.userAgent.indexOf("Gecko")
!=-1)) {
    document.write("<a href=
        HelloWorld.jnlp>Press here to
        launch Hello World using
        JWS</a>");
} else {
    document.write("WebStart is not
        Installed, Click ");
    document.write("<a href=http://
    java.sun.com/j2se/1.4.2/down
    load.html>here</a> ");
    document.write("to download and
        install JRE 1.4.2 and try
        to run the app again.");
}
</SCRIPT>
</BODY>
</HTML>
```

LISTING 4: WEB.XML SPECIFYING THE JWS SERVLET MAPPINGS

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE web-app PUBLIC "-//Sun
Microsystems, Inc.//DTD Web
Application 2.3//EN"
"http://java.sun.com/dtd/web-
app_2_3.dtd">
<web-app id="WebApp">
    <display-name>JavaWebStartTest</
        display-name>
  <servlet>
        <servlet-
name>JnlpDownloadServlet</servlet-name>
        <display-name>JnlpDownlo
          adServlet</display-name>
        <servlet-class>com.sun
          .javaws.servlet.JnlpDownload
          Servlet</servlet-class>
        <load-on-startup>-1</load-on-
          startup>
    </servlet>
  <servlet-mapping>
        <servlet-name>JnlpDownloadServ
          let</servlet-name>
        <url-pattern>*.jnlp</url
          -pattern>
    </servlet-mapping>
  <mime-mapping>
        <extension>jnlp</extension>
        <mime-type>application/x-ja
          va-jnlp-file</mime-type>
    </mime-mapping>
    <welcome-file-list>
        <welcome-file>index.html</
          welcome-file>
        <welcome-file>index.htm</
          welcome-file>
        <welcome-file>index.jsp</
          welcome-file>
        <welcome-file>default
          .html</welcome-file>
        <welcome-file>default
          .htm</welcome-file>
        <welcome-file>default
          .jsp</welcome-file>
    </welcome-file-list>
</web-app>
```

LISTING 5: JWS FILE WITH DYNAMIC TIMESTAMP

```
TS: 2003-08-13 15:00:30
<?xml version="1.0" encoding="utf-8"?>
<jnlp spec="1.0+" codebase="$$codebase"
  href="HelloWorld.jnlp">
    <information>
        <title>HelloWorld</title>
        <vendor>IBM - JWS example of
          HelloWorld</vendor>
        <description>HelloWorld -
          Example of JWS</description>
        <description kind="short">
          HelloWorld example</
          description>
        <offline-allowed/>
    </information>
    <resources>
        <j2se version="1.3+"/>
        <jar href="HelloWorld.jar"/>
    </resources>
    <application-desc main-
class="HelloWorld"/>
</jnlp>
```

WebSphere JOURNAL Coming Next Month...

SECURITY
Secure Messaging in a Dangerous World
BY JEFFREY L DEMENT

PERSONALIZATION
A Comparison of WebSphere Commerce 5.5 Personalized Marketing and WebSphere Portal Content Publishing 5.0 Personalization
BY BHADRI MADAPUSI

DATA
Keeping Your Finger on the Pulse: Archiving User Traffic and Sales Data Using WebSphere Commerce
BY SAPNA MAHWAL & RAHUL KITCHLU

WEBSPHERE STUDIO
Using JAXB in Developing Enterprise Applications With WebSphere Studio V5
BY TILAK MITRA

Part 2: Deploy and verify a user-defined taxonomy in the WebSphere UDDI Registry

UDDI and User-Defined Taxonomies

BY ROB **BREEDS**
& YEN **LU**

UDDI allows providers to publish information about their services in a common registry. A key part of this process is to describe services in an accurate and consistent way so potential users can easily find them using UDDI inquiry functions. Classifying services with user-defined taxonomies is an effective way to improve search results.

ABOUT THE AUTHOR

Rob Breeds is a staff software engineer at IBM Hursley Laboratories, UK, and is lead developer for user interfaces and tooling in IBM WebSphere UDDI Registry. He has worked on development projects since WebSphere 2.0. Rob has over 20 years of experience in software development and has worked in technical sales, marketing, and publishing roles.

E-MAIL
rob_breeds
@uk.ibm.com

Part 1 of this series, "Create and Test a User-Defined Taxonomy in WebSphere Studio" [*WJ*, Vol. 3, issue 1], described how you can define and test taxonomies for UDDI in IBM WebSphere Studio's Unit Test environment. This article explains how to deploy and use user-defined taxonomies with WebSphere UDDI Registry in WebSphere Application Server Network Deployment Edition (version 5.0.2 and above). To try the examples you must have IBM WebSphere UDDI Registry deployed and started.

Deploy the User-Defined Taxonomy to a WebSphere UDDI Registry

Deploying a new taxonomy to a WebSphere UDDI Registry takes two easy steps: load the data in the UDDI registry database, and publish a categorization tModel. Before loading the taxonomy data, let's look at a row from the file created in Part 1 (actually the row you added in Part 1) to see how its data fits in with the examples

that follow.

```
procclas#15.05.05#Manage Customer
    Addresses#15.05
```

With a delimiter of "#", "procclas" is the taxonomy name, "15.05.05" is the value, "Manage Customer Addresses" is the description for that value, and 15.05 is the parent code for the 15.05.05 value.

To load taxonomy data into WebSphere UDDI Registry you will use the UDDITaxonomyTools.jar command-line tool. It's worth taking the time to configure the tool correctly for your UDDI Registry installation. Typically, you need only do this once. We'll assume an IBM DB2 database setup, with the driver classes located at c:/Program Files/SQLLIB/java12/db2java.zip. Adjust the instructions accordingly for your environment.
1. Locate the UDDITaxonomy Tools.jar file. You will find it in the <installed location of deployment manager>/UDDIReg/scripts directory.

2. Copy UDDITaxonomyTools.jar to a convenient location. For this example, we'll assume the JAR file is stored in c:\tools and the taxonomy data files are in c:\taxonomies.
3. Create a configuration properties file, also in the c:\taxonomies directory, and call it "userTaxonomy.properties".
4. Edit the configuration file and complete the details as follows, changing them where necessary to suit your environment (using forward slashes in file paths):

```
classpath="c:/Program Files/SQL
    LIB/java12/db2java.zip;c:/tools/
    UDDITaxonomyTools.jar"
database.driver.className=COM.ibm
    .db2.jdbc.app.DB2Driver
database.url=jdbc:db2:UDDI20
database.userName=db2admin
database.password=db2admin
column.delimiter=#
string.delimiter=\"
```

This example is for a DB2 version 7 setup. If you are using DB2 version 8, the classpath will be similar to:

```
classpath="c:/Program Files/IBM/
    SQLLIB/java/db2java.zip;C:/
    tools/UDDITaxonomyTools.jar"
```

For Cloudscape, the classpath would be something like (assuming the WebSphere lib directory is under c:/websphere/appserver):

```
classpath="c:/websphere/appserver/
    lib/db2j.jar;c:/tools/UDDITaxon
    omyTools.jar"
```

Take care to get the database driver JAR correct in the properties file, as the Cloudscape db2j.jar is easily confused as belonging to DB2! The database.driver.className for Cloudscape would be:

```
database.driver.className=com.ibm
    .db2j.jdbc.DB2jDriver
```

Also note that for Cloudscape, the database.url value will be "jdbc:db2j:<directory of UDDI database>", and that database.userName and database.password are ignored.

Remember to enclose the classpath property in double quotes if the path includes space characters. If using the tool on a Linux platform, replace the semi-colons with colons. The classpath must include the UDDITaxonomy Tools.jar filepath.

Each property name should be self-explanatory. The last one, string.delimiter, allows you to include values that contain the column delimiter. For example, if you had a category value of "01" with a description of "service standard #1", you would need to surround the description value in the data file with double quotes:

```
mytax#01#"service standard
  #1"#00
```

See the information center for WebSphere Application Server Network Deployment for more detail about how to configure and use user-defined taxonomies in WebSphere UDDI Registry. (Search for "Custom Taxonomy Support".)

Now that the configuration file is set up, load the taxonomy data file:
1. At the command line, change the directory to the location of the JAR file (c:\tools).
2. Enter the following (all on the same line):

```
java -jar UDDITaxonomyTools.jar
  -load c:/taxonomies/pro
  cess1.tax -properties c:/tax
  onomies/userTaxonomy.prop
  erties
```

3. The tool reports:

```
Loaded 51 lines of data file
  for taxonomy name procclas.
```

The -properties argument allows you to have several configurations that you can switch between easily, say one for Cloudscape and one for a DB2 database. You can choose a default configuration by renaming or copying the properties file to "customTaxonomy.properties" and storing it in the same directory as the UDDITaxonomyTools.jar file. You then don't need to specify the properties argument, so the earlier example would become simply:

```
java -jar UDDITaxonomyTools.jar
  -load c:/taxonomies/pro
  cess1.tax
```

If you were to try entering the above line after you've already loaded the taxonomy data, the tool will report that a taxonomy with the name "procclas" already exists.

The taxonomy name in the example is "procclas" and that's what's been inserted into the UDDI registry database. What if we don't like that name or realize it's a mistake? Use the tool's rename function to rename the taxonomy as "process1":
1. Enter the following (assuming you created a default properties file called "custom Taxonomy.properties" in the c:\tools directory as described earlier):

```
java -jar UDDITaxonomyTools.jar
  -rename procclas process1
```

2. The tool reports the number of rows updated:

```
Renamed 51 taxonomy name val
  ues from procclas to
  process1.
```

An important point to make is that naics, geo, unspsc, unspsc7, other, and udditype are reserved taxonomy names for the default internal taxonomies used by the WebSphere UDDI Registry. If you attempt to perform load, rename, or unload functions using these taxonomy names, the taxonomy tool issues a warning to say that the internal taxonomy exists, and offers an additional "override" argument to allow you to proceed with the operation. While it is possible to force the change in the database, we strongly recommend you leave the reserved taxonomy data as it is, to prevent possible data corruption with invalid values and to avoid breaking standards.

The taxonomy data is now in the registry database. The next step is to publish a categorization tModel that corresponds to the set of valid values in your new taxonomy. There are several ways to publish entities in WebSphere UDDI Registry and we'll show you two: using a SOAP message containing a UDDI publish request, and using the browser-based UDDI user console.

A categorization tModel must have a name (strictly, this should be a URI), ideally a description, and a categoryBag containing at least the keyedReference to the uddi-org:types categorization tModel (tModel key UUID:C1ACF26D-9672-4404-9D70-39B756E62AB4) with a key value of "categorization". Typically, category values used to describe a UDDI entity are validated against the set of valid values in the taxonomy. To indicate this in a categorization tModel, another keyedReference to the uddi-org:types tModel is present in the categoryBag, with a value of "checked".

With this information, you can publish the categorization tModel shown in Listing 1. However, there is one crucial bit of information missing – how does the UDDI registry associate the categorization tModel with the set of valid values that we loaded earlier? In addition, if registry users are using the UDDI user console to choose category values for their inquiry and publish operations, it is helpful to have a succinct, user-friendly name in the taxonomy tree display.

All UDDI registries provide a special taxonomy, uddi-org:general_keywords (with tModel key UUID:A035A07C-F362-44DD-8F95-E2B134BF43B4). It is special in that any keyedReference to it can take any key name/value pair (it's an unchecked taxonomy). See the Resources section for more information about the General Keywords taxonomy. Unlike other keyedReferences, the keyName is meaningful and should be represented as a URN (Uniform Resource Name). The WebSphere UDDI Registry uses this special taxonomy
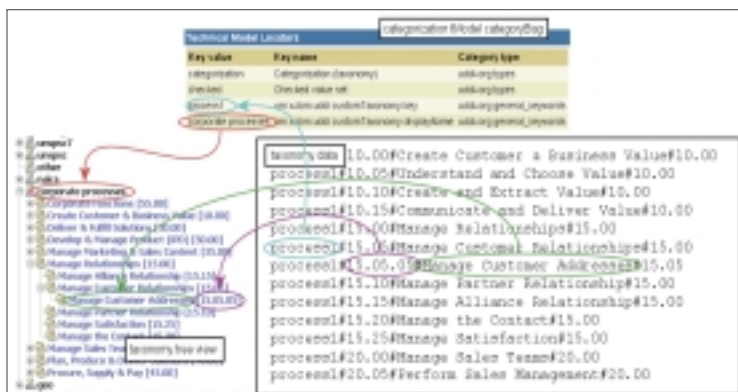
WEB **SERVICES**

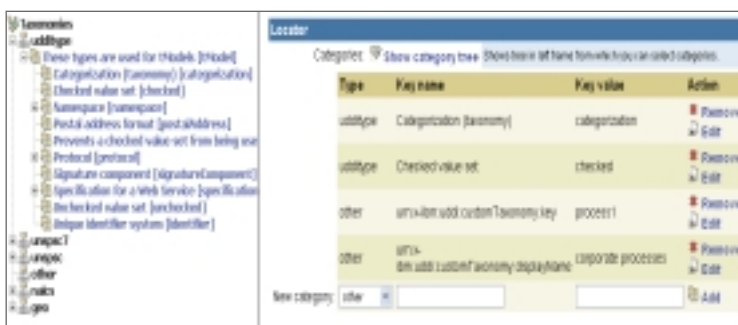**FIG. 1: HOW THE CATEGORIZATION TMODEL, TAXONOMY DATA FILE, AND TAXONOMY TREE ARE INTERRELATED**

FIG. 2: ADDING CATEGORY REFERENCES REQUIRED TO DEFINE A CATEGORIZATION TMODEL

FIG. 3: TWO BUSINESSES BEFORE ADDING SERVICES

FIG. 4: THE LOCATOR SECTION OF THE PUBLISH SERVICE PAGE SHOWING THE CATEGORY REFERENCES FROM TWO DIFFERENT TAXONOMIES FOR ADDRESS SERVICE 01

ABOUT THE AUTHOR

Yen Lu is an advisory software developer at IBM Canada Ltd., where he is currently responsible for the architecture and development of the Web Services Explorer and the Unit Test UDDI Wizard in the IBM WebSphere Studio suite. Yen holds a MASc in electrical engineering from the University of Toronto.

E-MAIL
yenlu@
ca.ibm.com

tModel to associate user-defined categorization tModels with taxonomy data loaded by the user, and optionally, to allow taxonomy publishers to specify a short name for use in user interfaces.

The following keyedReference is required to define the association between our new categorization tModel and the taxonomy data we loaded:

```
<keyedReference tModelKey=
  "UUID:A035A07C-F362-44DD-
  8F95-E2B134BF43B4"
  keyName="urn:x-ibm:ud
  di:customTaxonomy:key"
  keyValue="process1"/>
```

In this case the keyName value is a URN specific to the IBM WebSphere UDDI Registry, which specifies that the corresponding keyValue value is the user-defined taxonomy key in the taxonomy table of the registry's database, in this case "process1". The tModel key is that of the uddi-org:general_keywords taxonomy.

Similarly, if we want to specify a user-friendly name for user interfaces to exploit (including the user console of WebSphere UDDI Registry, and Web Services Explorer in WebSphere Studio), we also need a keyedReference to the special general keywords tModel, specifying a display name:

```
<keyedReference tModel
  Key="UUID:A035A07C-F362-
  44DD-8F95-E2B134BF43B4"
  keyName="urn:x-ibm:uddi
  :customTaxonomy:display
  Name" keyValue="corporate
  processes"/>
```

With these two additional keyedReference entities in our tModel, we can publish the tModel using SOAP, with the message shown in Listing 2.

Figure 1 shows the relationship between the taxonomy data file, the categorization tModel, and its keyedReferences, and how the taxonomy appears in the UDDI user console's taxonomy tree display. Note that the customTaxonomy:key key value has the same restriction as when loading user-defined taxonomy data files – you cannot specify reserved key names for naics, geo, unspsc, unspsc7, other, or udditype. Similarly, if a customTaxonomy:key key value for the taxonomy already exists, attempts to add a duplicate value are rejected.

If you prefer to use the UDDI user console to publish your categorization tModel, perform the

following steps (which assume WebSphere security is not active):

1. In a browser window, go to the UDDI user console home page. By default this is http://<host-name>:9080/uddigui.
2. Click the Publish tab, and then choose "Add a technical model" from the dropdown menu.
3. Enter "mycompany:process-es:2003.11" in the Name field.
4. In the Descriptions section, enter "This tModel specifies a taxonomy for describing corporate process-es." in the Description field, leave English as the selected language, and click the Add button.
5. In the Locator section, click on "Show category tree".
6. A tree of existing taxonomies displays in the left navigation frame. Expand the udditype taxonomy so that the Categorization (taxonomy) value [categorization] is visible (see Figure 2).
7. Click on "Categorization (taxonomy) [categorization]". The key name and key value appear in the Locator section. Click on Add to add this key name and key value to the list of categories for this tModel. Adding this value indicates this tModel is a taxonomy tModel.
8. In the taxonomy tree frame, click on "Checked value set [checked]". The key value and key name appear in the tModel's Locator section. Click on Add to add this key name and key value to the tModel's categoryBag. This value indicates this taxonomy is checked.
9. Still in the Locator section, on the New Category line, choose "other" in the pulldown list of taxonomies. "other" is the shorthand name used for the uddi-org:general_keywords taxonomy.
10. For the keyName value, enter "urn:x-ibm:uddi:customTaxonomy:key".
11. For the keyValue value, enter "process1" and click Add to add this keyedReference.
12. On the New Category line, "other"

should already be selected in the pulldown list; if it isn't, select it now.
13. For the keyName value, enter "urn:x-ibm:uddi:customTaxonomy:displayName".
14. For the keyValue value, enter "corporate processes" and click Add to add this keyedReference. At this point the Locator section should look like that in Figure 2.
15. Click on "Publish Technical Model".

Now that the categorization tModel is published, the user-defined taxonomy is immediately available for use by SOAP requests. To use the taxonomy in the user console you will need to stop and restart the UDDI application (restart the EAR). This makes the taxonomy appear in the taxonomy tree and pulldown list of taxonomies the next time you use the user console for publish and inquiry operations.

It is important to note that the user-defined taxonomy you've deployed is an internal taxonomy rather than an external taxonomy service. As such, it behaves exactly like any of the default taxonomies, such as NAICS and UNSPSC7, that are supplied with the WebSphere UDDI Registry.

Verify the Deployed Taxonomy

We published a user-defined taxonomy to allow us to better describe and precisely locate UDDI entities. First, we will publish two services with categories from different taxonomies. To publish services, you need to publish businesses to contain them. Using the UDDI user console:

1. If the taxonomy tree is still showing in the left frame, click on Publish in the top frame.
2. From the Publish tab, click on "Add a business".
3. In the Business Name section, enter "business 01" for the new name field, leave the English as the selected language, and click

on Add.
4. Click on "Publish Business".
5. Repeat Steps 1 through 3, this time using a name of "business 02".

Add a Service to the First Business

1. From the Publish tab, click on "Show owned entities". The Registered Businesses section will list your two businesses, as shown in Figure 3.
2. For business 01, click on Add service to display the Publish Service page.
3. In the Service Name section, enter "address service 01" for the new name field, leave English as the selected language, and click on Add.
4. In the Locator section (you might need to scroll down), click on "Show category tree". A tree of all internal taxonomies will display in the left frame, including the user-defined taxonomy labeled "corporate processes".
5. In the taxonomy tree frame, expand the corporate processes taxonomy so that the nodes for Manage Relationships [15.00], Manage Customer Relationships [15.05], and Manage Customer Addresses [15.05.05] are visible. Resize the left frame if necessary.
6. Click on Manage Relationships [15.00]. This copies the values to the key name and key value fields in the Locator section. Click on Add to add that keyedReference.
7. Repeat Step 6, this time selecting Manage Customer Relationships [15.05].
8. Repeat Step 6 again to add the Manage Customer Addresses [15.05.05] reference to the Locator section of the service.
9. Now expand the "geo" taxonomy in the tree so that Canada [CA] and Ontario [CA-ON] are visible.
10. Follow Step 6 again to add keyName and keyValue values for Canada [CA] and Ontario [CA-ON] to the Locator section. Your Locator section should now look

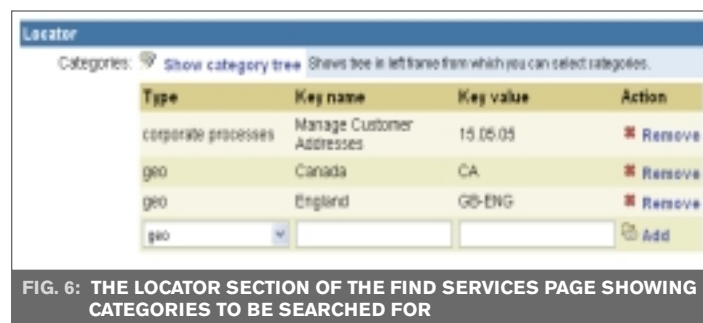**FIG. 5: THE LOCATOR SECTION OF THE PUBLISH SERVICE PAGE FOR ADDRESS SERVICE 02**

FIG. 6: THE LOCATOR SECTION OF THE FIND SERVICES PAGE SHOWING CATEGORIES TO BE SEARCHED FOR
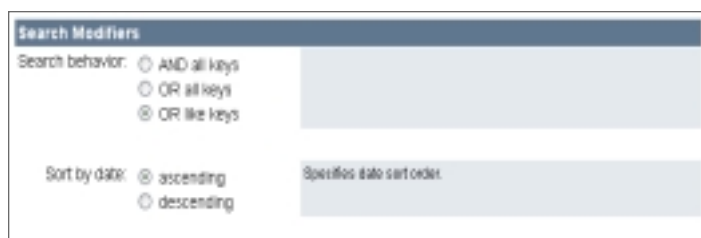

FIG. 7: SELECTING "OR LIKE KEYS" IN THE SEARCH MODIFIERS SECTION OF THE PUBLISH SERVICE PAGE CAUSES REFERENCES TO CATEGORIES IN THE SAME TAXONOMY TO BE OR'ED TOGETHER
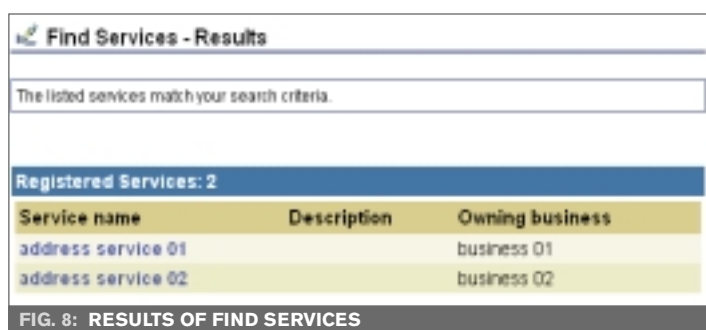

FIG. 8: RESULTS OF FIND SERVICES

like that shown in Figure 4.

11. Click on Publish Service.

To complete the example, create a second service under business 02, with the same references to the user-defined taxonomy, and the service located in England, UK:

1. Click on Publish in the top frame to display the Publish tab.
2. In the Publish tab, click on "Show owned entities". The Registered Businesses section will show your two businesses.
3. For business 02, click Add service to display the Publish Service page.
4. Now repeat Steps 3 through 8 of the previous instructions, this time substituting "address service 02" for "address service 01".
5. Now expand the geo taxonomy in the tree so that United Kingdom [GB] and England [GB-ENG] are visible.
6. Repeat Step 6 (from the previous sequence) to add keyName and keyValue values for United Kingdom [GB] and England [GB-ENG] to the Locator section. For the second service, the Locator section should look like Figure 5.
7. Click on Publish Service.

This example shows how the new taxonomy might be used to describe two separate services, both with the Manage Relationships category reference and subcategories. One service is applicable to the Ontario, Canada region, and the other to England, UK. Notice that for the location references, the categoryBag includes both the region and the country values. Similarly, the services specify values in the user-defined taxonomy of "15.00", "15.05", and "15.05.05". This technique helps service seekers who might begin their search looking only at services based in Canada, for example, before they narrow their criteria. Having the detailed value and one or more levels of parent values gives inquirers a better chance of locating a particular service.

An interesting aspect of user-defined taxonomies in UDDI comes into play when users start seeking entities of specific interest to them. To search for all businessService entities that have category references to "Manage Customer Addresses" (value 15.05.05) from the user-defined taxonomy *and* are located in Canada *or* England, UK, perform an inquiry as follows:

1. Click on Find in the top frame to display the Find tab.
2. From the Find tab, click on Find Services. The Find Services page displays.
3. In the Locator section, click on "Show category tree".
4. In the taxonomy tree frame, expand the corporate processes taxonomy so "Manage Customer

BREAK AWAY
FROM THE PACK

kenetiks
WORLDWIDE JAVA TRAINING & MENTORING

www.kenetiks.com
888.KENETIKS

- **WEBSPHERE TRAINING**

- **PROJECT MENTORING**

- **COURSE LEASING**

From development to deployment, Kenetiks shows you how to build J2EE applications for WebSphere Application Server and WebSphere Portal Server using WebSphere Studio Application Developer.

IBM Business Partner

"If registry users are using the UDDI user console to choose category values for their inquiry and publish operations, it is helpful to have a succinct, user-friendly name in the taxonomy tree display"

Addresses [15.05.05]" is visible.
5. Click on "Manage Customer Addresses [15.05.05]" and then on Add in the Locator section to add that reference.
6. Expand the geo taxonomy and use the same technique as in Step 5 to add two category references, one for "Canada [CA]" and one for "England [GB-ENG]". Your Locator section should look like Figure 6.
7. In the Search Modifiers section,

select the "OR like keys" radio button (see Figure 7).
8. Click on Find Services. The results should contain both address services, as shown in Figure 8.

The "OR like keys" find qualifier causes category references in the same namespace (same taxonomy) to be OR'ed together rather than AND'ed. If you wanted only services for "Manage Customer Addresses" based in Canada you would leave the

search modifier set to the default "AND all keys" and remove the reference to "England".

Conclusion

In this article, we created new category data, loaded it into the IBM WebSphere UDDI Registry, published a corresponding categorization tModel, and demonstrated how to use the taxonomy in publishing and finding UDDI entities. User-defined taxonomy support in the WebSphere UDDI Registry offers additional levels of detail with which to define services, as well as offering users a more effective way to pinpoint the services they want.

Resources

- *UDDI specifications:* www.uddi.org
- *UDDI tModels – Classification Schemes, Taxonomies, Identifier Systems, and Relationships, Version 2.04:* http://uddi.org/taxonomies/UDDI_Taxonomy_tModels.htm
- *WebSphere Application Server Version 5.1 information center:* http://publib.boulder.ibm.com/infocenter/ws51help/index.jsp

LISTING 1

```
<tModel tModelKey="">

  <name>mycompany:processes:2003.11</name>

  <description> This tModel specifies a taxonomy for

      describing corporate processes.</description>

  <categoryBag>

    <keyedReference tModelKey="UUID:C1ACF26D-9672-4404-

      9D70-39B756E62AB4" keyValue="categorization"/>

    <keyedReference tModelKey="UUID:C1ACF26D-9672-4404-

      9D70-39B756E62AB4" keyValue="checked"/>

  </categoryBag>

</tModel>
```

LISTING 2

```
<save_tModel generic="2.0" xmlns="urn:uddi-org:api_v2">

  <authInfo></authInfo>

  <tModel tModelKey="">

  <name>mycompany:processes:2003.11</name>

  <description>This tModel specifies a taxonomy for

    describing corporate processes.</description>

  <categoryBag>

    <keyedReference tModelKey="UUID:C1ACF26D-9672-4404-

        9D70-39B756E62AB4" keyValue="categorization"/>

    <keyedReference tModelKey="UUID:C1ACF26D-9672-4404-

        9D70-39B756E62AB4" keyValue="checked"/>

    <keyedReference tModelKey="UUID:A035A07C-F362-44DD-

        8F95-E2B134BF43B4" keyName="urn:x-

        ibm:uddi:customTaxonomy:key" keyValue="process1"/>

    <keyedReference tModelKey="UUID:A035A07C-F362-44DD-

        8F95-E2B134BF43B4" keyName="urn:x-

        ibm:uddi:customTaxonomy:displayName"

        keyValue="corporate processes"/>

  </categoryBag>

  </tModel>

</save_tModel>
```

CRAIG HAYMAN

VICE PRESIDENT OF WORLDWIDE DEVELOPMENT FOR IBM PERVASIVE COMPUTING

Pervasive Computing

IBM products address the need to be everyplace at once

WebSphere Journal recently interviewed Craig Hayman, vice president of Worldwide Development for IBM Pervasive Computing. In this exclusive interview, Hayman discusses pervasive computing and its role in IBM's on-demand computing initiative, IBM Pervasive Computing's key products, and some of IBM's pervasive computing success stories.

WJ: What is pervasive computing?

Craig Hayman: Pervasive computing extends e-business, enabling access to information and creating new interactions with new devices, part of IBM's on-demand approach to computing. Pervasive computing is about making it easy for customers to connect any type of device to any type of network to access any type of data in a flexible, secure, reliable, scalable way.

People typically think of wireless solutions in terms of mobile phones and PDAs, but it will be an integrated infrastructure – technology, hardware, software, and services – and enabling the communication between many devices, including chips, refrigerators, and cars – that will emerge as the key to the growth of wireless solutions. Handhelds are just the tip of the iceberg. Wireless technology is rapidly growing to support telematics, home gateways, consumer electronics, appliances, and Web-enabled self-service kiosks.

WJ: How are customers taking advantage of pervasive computing to gain a competitive advantage?

CH: Our customers use IBM's WebSphere mobile middleware to extend PIM [Personal Information Manager], e-mail, and Lotus Sametime Instant Messaging to devices. The applications that we enable help salespeople place orders from the field; retailers keep better inventory; insurance companies process claims more efficiently; and airline passengers use kiosks or benefit from "line-busting" PDAs.

A few other examples include warehouse workers who can receive incoming shipments from suppliers using their wireless PDA, allowing them to instantaneously update inventory systems and other key business systems.

Police departments can instantly download mug shots of suspected criminals from the beat with a handheld device. Doctors can make diagnoses and prescribe drugs more quickly to patients, who will be able to confirm prices and pick up prescriptions immediately. Mobile delivery personnel can keep shelves in their stores efficiently stocked using accurate routing and delivery information loaded into their wireless devices around the clock.

We approach three customer segments in pervasive computing: first, enterprise customers, for whom we build wireless capabilities upon their existing e-business infrastructure; second, network service providers like Nextel and AT&T Wireless, with whom we form partnerships to add data capabilities that they can pass on to customers; and third, device manufacturers like Nokia, Palm, Sony Ericsson, and Sharp, which we work with to include IBM middleware in their products.

WJ: What role does pervasive computing play in IBM's overall strategy for on-demand computing?

CH: On-demand is a major initiative for us, and pervasive computing helps customers access key business information, which plays a central role in IBM's on-demand strategy. Our goal is to help businesses be more flexible. Businesses are transforming their internal processes to be horizontal in nature, and these processes can connect on one side with the suppliers and on the other side with the distributors and customers. The end-to-end process must be flexible and dynamic enough that the organizations can respond to any change in market demand or to competitor threat.

Two areas of focus that are important to our operating environment and on-demand are open standards and autonomic computing. IBM's partnerships with the world's leading device makers represent the beginning of a movement to develop a framework to provide true interoperability across operating systems, devices, markets, and operators. Building an open standards–based wireless ecosystem allows customers to avoid a complex, time-consuming, and costly rip-and-replace process, and enables them to extend the existing IT system to mobile workers. We look at autonomic capabilities where the environment, whether it's the computer or the network infrastructure, has the ability to self-heal. In the operating environment,

ABOUT CRAIG HAYMAN

Craig Hayman is vice president of Worldwide Development for IBM Pervasive Computing, with responsibility for the delivery of IBM's mobile and voice products. He was previously vice president of Strategy for IBM Software Group. Craig joined IBM in 1997 as program director for Metadata Management, where he worked to create the XML standard needed to represent objects in transactional systems. In 1999 he was part of the core three-person team that initiated the major revamp of the IBM development tools product line. The resulting WebSphere Studio line of products was launched in 2001, together with the underlying open source development environment, Eclipse. In January 2000 Craig was named director of WebSphere Development, where he led the product development and support of the WebSphere Application Server.

businesses want to add flexibility so that any device will have access to key business information. Our focus is on driving the infrastructure to a broad range of devices and delivering that information in a very flexible and responsible way.

WJ: What key IBM products are involved in making pervasive computing work?

CH: The IBM Pervasive division's key products are WebSphere Everyplace Connection Manager [WECM] and WebSphere Everyplace Access [WEA] [see Table 1 for more information on IBM Pervasive Computing products.].

With the new version of WECM mobile employees can, using a device or a laptop, move from location to location enjoying secure, high-speed access to information – even when roaming across Wi-Fi, cellular, GPRS, CDMA, GSM, and PWLAN hot spots – without interrupting Web connections or losing an existing session.

WebSphere Everyplace Access middleware enables a wide range of pervasive devices that operate on open standards and over wireless networks to access Web applications like PIM and e-mail and Lotus Sametime Instant Messaging. With the most recent release of WEA, IBM now supports more handheld operating systems worldwide than any other vendor, including palmOne devices on Palm OS, Sharp Zaurus on Linux, Nokia and Sony Ericsson on Symbian, and Blackberry devices on RIM and Microsoft Pocket PC platforms. Together, these handheld operating systems make up at least 90% of the worldwide market.

WJ: What kinds of strategic partnerships are involved in pervasive computing?

CH: If you look at the partners we've made recent announcements with (Nokia, RIM, Sharp, palmOne, Sony, Openwave, Matsushita), you'll see the beginnings of a movement to develop a framework to provide true interoperability across operating systems, devices, markets, and operators. The ultimate goal is to create a multivendor ecosystem that mirrors the end-to-end Java-based programming environment of the Internet, providing customers with the ability to choose between a wide range of technology providers, and freeing them from reliance on one vendor.

WJ: How important are open standards to wireless? What work is your division doing in that space?

CH: We participate heavily with the W3C [World Wide Web Consortium] and the OMA [Open Mobile Alliance]. We also participate in the Internet Home Alliance in terms of consumer-based applications. We recently became a member of the Consumer Electronics Linux Forum. The focus is to roll out open industry standards using this end-to-end pervasive vision. Other areas include our work in XHTML and in VoiceXML.

We've also been working closely with our own Web services teams. In terms of various key standards and connections with key standards bodies, we're very active. It's certainly one of the more conscious portions of our strategy.

As we continue to build on open standards, it will not only allow open interoperability but will also help our customers deploy the solutions they want.

THE WEBSPHERE EVERYPLACE FAMILY OF PRODUCTS

WebSphere Everyplace Access	Extends productivity, data, and enterprise applications to mobile employees
WebSphere Everyplace Connection Manager	Delivers remote access to existing applications with a high level of security and seamless roaming
WebSphere Everyplace Embedded Software	Provides an end-to-end embedded software solution for a broad range of remote and wireless devices
WebSphere Everyplace Server, Service Provider Offering	Connects, adapts, manages, transforms, and scales Web applications and legacy data into the pervasive computing environment
WebSphere Everyplace Subscription Manager	Creates and manages service offerings while delivering a mature, industry-proven infrastructure for broadband and dial-up service providers.

THE WEBSPHERE VOICE FAMILY OF PRODUCTS

WebSphere Voice Application Access	Extends the WebSphere Portal infrastructure and programming model to voice
WebSphere Voice Server	Develops and deploys conversational e-business solutions
WebSphere Voice Response for Windows	Brings expanded functionality to Interactive Voice Recognition applications on the Windows OS
WebSphere Voice Response for AIX	Brings expanded functionality to Interactive Voice Recognition applications suited for large enterprises or telecommunications businesses
Unified Messaging for WebSphere Voice Response	Manages employee and customer voice mail, e-mail, and faxes for access virtually anywhere and any time over the telephone or Internet

TABLE 1: IBM PERVASIVE COMPUTING PRODUCTS

WJ: What trends do you see developing that will turn into interesting business models during the next two years?

CH: Wireless is the catalyst of what we describe as pervasive. All the other aspects are multimodal, or the ability to use other ways to interact with the environment. We see a significant number of enterprises over the next year enabling things like their general business applications environment. This is extending e-mail and PIM-type applications to different types of devices, such as a PDA or smart phone.

The next stage is to do customer-facing services, enabling sales and field force automation so that businesses can do different things like view customer profiles, look at product descriptions, or compose orders from remote locations.

A third phase would be employee-facing services. I can go through my company portal to get a number of things, such as access to different types of enterprise applications or to access news or educational services. Enterprises are also looking to extend those capabilities.

WJ: What pervasive computing technologies or capabilities are you personally excited about?

CH: Looking forward to the next wave of computing, I'm excited about the work IBM is doing with the world's major automakers to enable embedded speech technology in the navigation systems of automobiles like the 2003 Honda Accords and MDXs – powered by IBM's Embedded ViaVoice. The technology enables drivers to ask for navigation information like directions to the grocery store or the nearest ATM machine.

I am also excited about multimodal technology – the ability to interact with a device at times using traditional input methods like text or mouse-clicks, and at other times using voice for response or input. And I'm excited about location-based services as an application. Location-based and intelligence-based notification can offer some very interesting opportunities moving forward. I get excited about multimodality and the ability to provide intelligent input.

WJ: In which industries is IBM Pervasive Computing already being used? Can you give me a couple of success stories?

CH: I was hoping you'd ask about that. We have a number of customer success stories in pervasive computing to talk about.

One that comes to mind is Phoenix-based Amberwood Homes, Arizona's largest custom home builder. They faced a problem that's all too familiar to their competitors in the construction industry. They needed to eliminate delays that can result from coordinating the schedules of dozens of subcontractors – and which can reduce the number of homes they can build, even though the housing sector is booming sector in that region of the country.

Project managers used to drive from site to site troubleshooting problems before concluding their days filling out piles of paperwork on the delays caused by work that wasn't done on schedule. Wireless technology changed that.

Now, in addition to the usual tools you might see these project managers carrying, they are also armed with PDAs running a custom software application to help them schedule contractors in real time at multiple job sites – saving hours and days of delays. When they identify a problem, rather than writing a report or making a cell-phone call to a subcontractor, the technology lets them immediately send a memo to a subcontractor on the PDA requesting that they to return to fix it. This also fends off unwanted litigation that can result from disputes over mix-ups arising from scheduling. More than a convenience, the application has helped Amberwood build homes more quickly – shaving 30 days off the average 90–120 days it takes to complete a custom home. Additionally, the homebuilder cut in half the 40–50 calls a

"With the most recent release of WEA, IBM now supports more handheld operating systems worldwide than any other vendor"

supervisor makes in an average day.

IBM is also working with the Air Force Material Center at Hill Air Force Base in Ogden, Utah, to radically transform the way they manage worldwide logistical support for the repair, replacement, and maintenance of thousands of components for the nation's fleet of F-16, A-10, and C-130 military aircraft. Hill Air Force Base expects to save millions of dollars a year from a highly secure FIPS 140–certified IBM wireless solution using WebSphere Everyplace Connection Manager.

The Toronto Police Department is another success story. A wireless application called eCOPS, which uses software with secure seamless roaming capabilities, is changing the way its officers fight crime. Using WLAN hot spots and a 2.5G network, which connect officers to back-end systems to retrieve information, the officers can get reports from the field, uncover trends and patterns for where crimes occur, and download mug shots of suspected criminals – all in real time. While identifying problems earlier, addressing them more quickly, and minimizing the response time and cost to the communities they are paid to serve, the department is saving nearly $5 million in salaries annually – with an investment they expect will pay for itself in four years.

Closer to home, a partnership of public safety and transportation agencies in Virginia, Maryland, and the District of Columbia is building the Capital Wireless Integrated Network for the Washington, D.C., region. The first interoperable wireless system to span multistate government jurisdictions, the network will enable officials from more than 40 local, state, and federal agencies to communicate with each other in real time to better protect our homeland.

As I mentioned earlier, IBM worked with Honda to speech-enable the navigation systems in the 2003 Accords and MDXs. Powered by IBM's Embedded ViaVoice, the technology enables drivers to ask for navigation information like directions to the store or nearest ATM machine.

Finally, IBM and USA Technologies are wirelessly enabling U.S. college dorm laundromats to offer students the convenience of cashless payments and the ability to check washer/dryer status over the Web. The technology allows the laundry equipment to be monitored remotely for maximum performance, calls for service, and replenishment of supplies.

WJ: What new things can we expect to see coming from IBM Pervasive Computing in 2004?

CH: IBM will continue to help enterprises extend e-business through pervasive and wireless e-business products and solutions, enabling customers and mobile workers access to key business information. Many people typically think of wireless solutions in terms of mobile phones and PDAs, but it will be an integrated infrastructure – technology, hardware, software, and services – and the enabling of communication between many devices, including chips, refrigerators, and cars – that will emerge as the key to the growth of pervasive and wireless solutions in 2004. Handhelds are just the tip of the iceberg. Wireless technology will continue to rapidly grow to support telematics, home gateways, consumer electronics, appliances, and Web-enabled, self-service kiosks.

A solution to the need for unit testing

Jtest 5.0
from Parasoft

BY BHAVESH **PATEL**

Most developers would agree that software development is not as daunting a task as is *efficient* software development. We have seen teams that can design and develop software – and with automated development tools, IDEs as they are called, the software development process has become a lot easier. But while an architect can always come up with a design and a development team can write the code for a given design, there aren't many developers who can write the best possible code for a given design – especially in the case of distributed application development, in which multiple teams are working on modules of a given application.

ABOUT THE AUTHOR

Bhavesh Patel, an enterprise architect with Noospherics Technologies, has over 10 years of IT experience and holds certifications from IBM, Microsoft, Compaq, and Sun. Bhavesh, who speaks and writes on WebSphere topics, has worked as a project mentor, tech lead, and WebSphere administrator for several major telecommunications and financial companies, and for the U.S. government and military.

E-MAIL
patelby
@noospherics.com

How can you ensure that what comes out of these multiple teams is coherent and that it will work? This is where the need for software testing tools comes in. Testing tools not only help you detect errors in code, they can help you implement coding standards. This is one of the reasons behind the growing popularity of automated testing tools in the IT industry. These tools ensure that many basic software testing tasks are automated, resulting in more efficient and quicker development. Amazing, isn't it? But with more than 40 different testing tools, deciding which to use is enough to cause headaches for most software managers.

Parasoft Jtest was one of the first products to receive the Ready for WebSphere validation from IBM, which means it meets all the requirements for easy integration with IBM WebSphere Studio. With WebSphere Studio, developers can use Jtest for error prevention, perform black-box and white-box testing, and also do static analysis of Java code on WebSphere.

During the process of reviewing Jtest version 5.0 I looked at and evaluated some of its important features – including installation and integration with WebSphere Studio, coding-standard enforcement, automated unit testing, and use of the product in a team environment.

Ease of Installation and Integration with WebSphere Studio

The Jtest plug-in for WebSphere Studio 5.1 (and for all IDEs based on Eclipse 2.x) is available for download from the Parasoft Web site (www.parasoft.com), in the "Downloads" section. Note that you must register and provide an e-mail address linked to a valid company.

Parasoft doesn't accept e-mail addresses like xyx@yahoo.com or abc@hotmail.com. Once you are registered and logged in to the Web site, select the product and platform to download.

Installation

The installation of the Jtest plug-in for WebSphere Studio is very simple and intuitive. You are asked to select a directory for installing the Jtest extension files. The default is C:\Program Files\Parasoft\JtestExtension. You are required to point to your WebSphere Studio (or Eclipse) installation directory. Be sure that WebSphere Studio isn't running before you start the installation process.

User Interface

The Jtest user interface is straightforward and easily blends into the WebSphere Studio 5.1 (Eclipse) user interface. It is also accessible as one of the perspectives of WebSphere Studio 5.1. You no longer need to open a separate software/user interface to use Jtest. Everything required for using Jtest is accessible via your WebSphere Studio perspectives, which reduces the learning curve. To start using Jtest, open the Jtest Perspective in WebSphere Studio, browse to your class, select it, click the "Play" button on the toolbar, and zoom – you've already run a test. The Jtest summary panel displays all the required information about the test, such as the number of files checked, number of errors found, etc. The errors are shown in the Errors Found view.

One of the main features of Jtest that I looked at was the enforcement of coding standards. Implementing coding standards for your team can require overwhelming effort – the lack of which can result in delayed product schedules. Developing coding standards and then ensuring that each developer adheres to them can be a very cumbersome and practically impossible task. However, Jtest helps
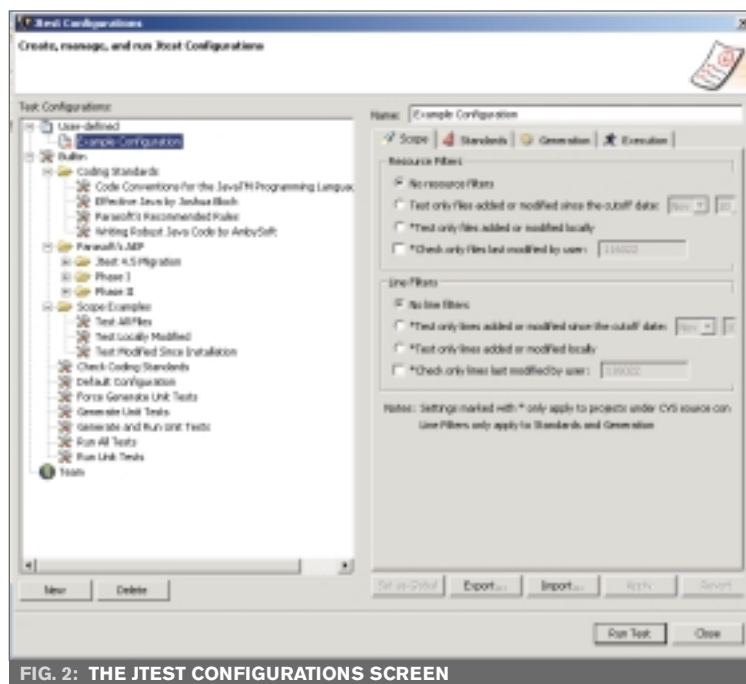
you automate the entire task of implementing coding standards. Jtest implements strict checks to test each class for as many as 380 coding standards/rules. Using such rules can help you eliminate possible errors and issues related to threading, garbage collection, memory leaks, etc. In addition to using built-in rules, Jtest allows you to write your own custom rules and include them to be checked by Jtest.

Jtest allows you to select from some of the most popular coding standard options, including "Code Conventions for the Java Programming Language," by Sun; *Effective Java,* by Joshua Bloch (one of my favorites); Parasoft's recommended rules, and "Writing Robust Java Code," by Ambysoft.

Using the Jtest GUI (see Figure 1) you can select a Parasoft Jtest configuration (see Figure 2) wherein you can choose the coding standards you want to follow. It could be any one of the above-mentioned industry-accepted coding standards, or it can be a mix. When I tried to create a custom mix from two coding standards I found that Jtest has no documentation about that, but the Jtest support team was quick in getting back to me with a well-documented solution about how to handle that!

Parasoft provides a description for each coding rule, or standard, that helps you understand the importance of the rule, how your code violates the given rule, and the effects doing so. It also gives you a choice to have Jtest automatically fix the coding standard/rules violation for you. Neat!
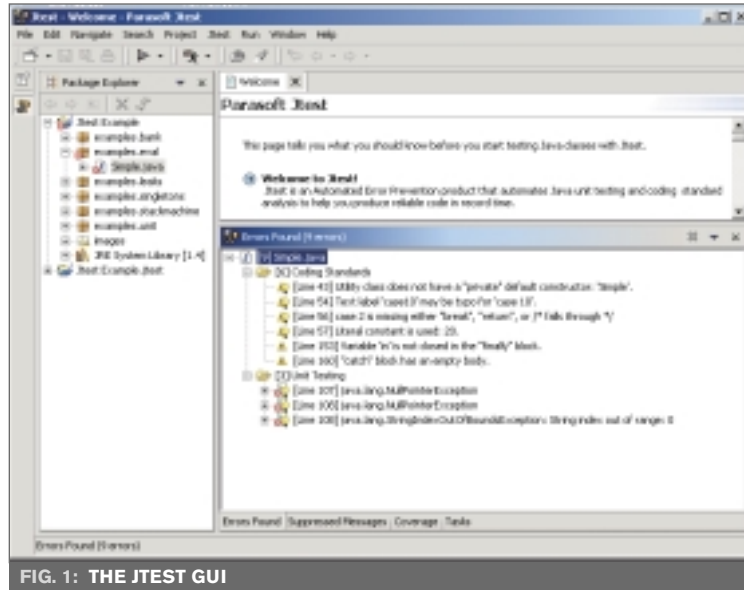
FIG. 1: THE JTEST GUI

Automated Unit Testing Using Jtest

Unit testing with the help of Jtest helps you confirm whether the code is working as it is supposed to. It enables seamless white-box testing for your projects, thereby helping you confirm the structural strength of your Java code. Jtest extends the power of JUnit and allows you to completely automate the unit testing of your projects. It does so by enabling automatic generation of JUnit test cases and test-case execution.

It also allows you to do functional testing (also known as black-box testing) for your projects. Functional testing ensures that your applications work as they are supposed to, or as the users would expect them to. This can be very useful in confirming whether the user requirements have been properly captured, as black-box testing tests the software from an end

user's perspective. This is done at the level of the smallest functional unit, and thereby the possible bugs or glitches in properly capturing user requirements are brought to notice much earlier in the process.

Jtest also helps you regression test your software and tracks all the changes to the code, maintaining a properly integrated and updated code base. This can be useful to identify whether the modified code has generated new errors. In large projects, regression testing is usually performed in nightly builds so that in the morning when the team comes back to their desks they have the error reports ready to work on, which saves valuable project time.

Detecting Memory Leaks Using Jtest

Jtest 5.0 monitors the object allocate events and object free events dur-

"Developing coding standards and then ensuring that each developer adheres to them can be a very cumbersome and practically impossible task"

FIG. 2: **THE JTEST CONFIGURATIONS SCREEN**

ing the execution of a unit test. By comparing the objects free versus the objects allocated, you can easily determine if there are any unreferenced objects living in the memory. In addition, because each test is executed three times, Jtest ensures that the leaked memory is not for initialization or cache. To detect memory leaks using Jtest you need to go to Execution > Options > and enable the "Detect memory leaks" check box. I won't say that Jtest will discover every possible memory leak, but it certainly allows those arising from poor programming to be easily caught and rectified.

> **"Implementing coding standards for your team can require overwhelming effort**
> **– the lack of which can result in delayed product schedules."**

Using Jtest in a Team Environment

A product that works well for one workstation but that isn't designed for team usage is of little help when it comes to team development. These days most IT projects involve a team of developers – and the product you use for automated unit testing must support a team environment. Jtest can be implemented across a team of developers/testers to help improve the quality of the code and, finally, the application, especially with the use of supportive products like the Group Reporting System and Team Server, both from Parasoft.

Parasoft has developed a recommended workflow that helps teams effectively use the automated error-prevention features Jtest offers. Builds occur on a nightly basis, and reports are generated and sent out to individual or group mailboxes every morning. Jtest offers batch testing facilities using the Jtest CLI (jtestcli.exe) that is in the Jtest installation directory.

However, to implement this, an ideal configuration would include one Jtest installation with a CLI license (the default Jtest license doesn't support the command-line interface or the batch mode), an installation of Jtest on each developer work-

station, and an installation of Parasoft's Group Reporting System and Team Server.

Developers will have to use their Jtest configurations to test the code they write, and then check the code in the team's source-control system. Each night the Jtest CLI will run on the build machine (pulling the source code from the source-control system), generate test cases, check the code strength, and store the results in the Group Reporting System. It can also mail out the reports generated to each developer (or a set of e-mail groups). The developer workstations will point to the Team Server, where they can access the Jtest configurations, coding rules, test cases, etc., to be used by the team. The Team Server is a single point of access for developers to get any updates that occur to the configurations, coding rules, etc.

Jtest is currently available with a price tag of $3,495 for one user (North American Users only). While this may not be an issue for a small team, for large teams you might want to have several testers with Jtest installations who can run tests for the rest of the team. I believe that Jtest should start offering some sort of team license, as it may difficult for an organization with a large team to implement Jtest on each developer's machine. During my last conversation with Parasoft, I was told that they plan to unveil a new licensing model to promote team adoption of Jtest in Q1, so you can check the Parasoft Web site for more information.
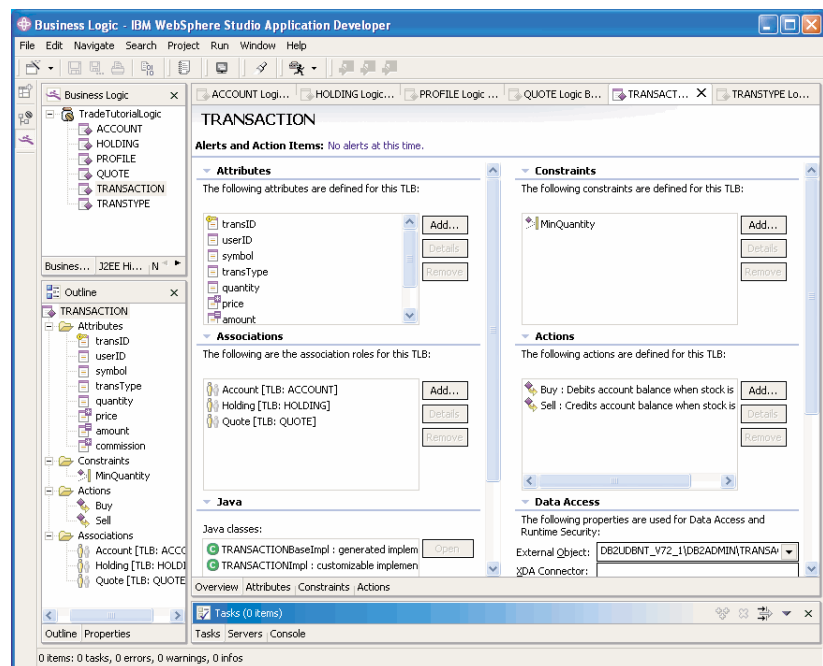
Conclusion

As an automated testing solution, Jtest passes the tests with flying colors. It helps you implement the industry-standard best practices during the development phase, allowing you to detect and eliminate coding errors and thereby reduce development time and cost, leveraging existing source-control systems. It installs quickly, integrates well with WebSphere Studio, and performs well! What's more, with a promise to improve upon the current features and add more, Jtest seems to be an ideal testing solution for development teams.

Excerpts from Chapter 28:
Transactions in WebSphere 5.0

Enterprise Java Programming with IBM WebSphere
2nd Edition

BY KYLE **BROWN**

The topic of transactions is one that most Java programmers would rather ignore than try to understand. And, in fact, in most cases you *can* ignore them – the default settings of WebSphere Application Server (WAS) and WebSphere Studio Application Developer work well enough in most situations that many programmers can build large and complex applications without having to know the details of how transactions work. Unfortunately, at some point this blissful ignorance must come to an end.

Then you have to hunker down and learn how transactions operate in order to solve problems that have ramifications all the way up and down your architecture. In this article, based on Chapter 28 of my new book, I'll examine how transactions operate in WAS.

Advice on Using Transactions

The best set of advice about EJB transactions that I've come across is a set of simple rules that Keys Botzum (from the IBM Software Services for WebSphere group) came up with that give you the 90 percent case for dealing with transactions. Keys' rules of thumb are:

- Always assume you're going to use container demarcated transactions when using EJBs. It's complicated and difficult to use the JTA API to do your own transaction demarcation, and not worth it in most circumstances.

- If you need transactions with a servlet (e.g., outside the EJB container), use the JTA API for demarcating a transaction. The beauty of this rule and the previous one is that by doing this you will not have to write your database code one way [using setAutoCommit()] to work within the EJB container and another way to work outside the EJB container. In fact, you should try not to mess with Connection .setAutoCommit()—just assume that the container will handle transaction commit/rollback for you.

- Assume that the container will handle the appropriate magic of managing local versus global transactions based on the number of participants. Also, assume that it will perform automatic PC optimizations if appropriate. Thus, there is no penalty to global transactions.

- Use XA enabled resources in the following situations:
  - If there may be more than one participant in a transaction (this could be two JDBC databases, or a database and a JMS connection or EIS connection, or any other combination). This allows the container to do any appropriate optimizations if there is only one participant, but to handle XA correctly if there are two or more.
  - If an EJB needs to access another EJB deployed in a different EJB container, then both containers should use XA resource managers. Sometimes one of your application's EJBs will need to use a utility EJB that provides some service to you. The only way to tie together the two EJBs into a single transaction is to use XA resources in both EJBs. This is an example of a distributed transaction; something relatively rare, but that also requires the use of XA resources.

These rules will work for most situations, but there are a few that you may find yourself in that will require you to go beyond the rules—in particular we need to look at some of the differences between the EJB 1.1 and EJB 2.0 specs with regard to local transactions. Let's look at the following sections from Section 6.5.7 of the EJB 1.1 spec:

"A session Bean's newInstance, setSessionContext, ejbCreate, ejbRemove, ejbPassivate, ejbActivate, and afterCompletion methods are called outside of the client's global transaction.

"For example, it would be wrong to perform database operations within a session Bean's ejbCreate or ejbRemove method and to assume that the operations are executed under the protection of a global transaction. The ejbCreate and ejbRemove methods are not controlled by a transaction attribute because handling rollbacks in these methods

would greatly complicate the session instance's state diagram (see next section)."

This statement was modified a bit in the EJB 2.0 specification to make things less confusing.

It said that the operations shouldn't be controlled by the transaction attribute of the bean, but it didn't specify what the behavior of these operations should be in relation to any ongoing global transaction. In particular, it didn't give the vendors much guidance as to how SQL statements in these methods should be handled. Should each statement be its own transaction (e.g., should it be as if the connection were in auto-commit mode) or should the method be a single transaction scope? So let's examine how this statement changed in EJB 2.0 (the following quote is from section 7.5.7 of the EJB 2.0 specification):

"A session bean's newInstance, setSessionContext, ejbCreate, ejbRemove, ejbPassivate, ejbActivate, and afterCompletion methods are called with an unspecifed transaction context. Refer to Subsection 17.6.5 for how the Container executes methods with an unspecified transaction context."

What the section discusses is that it is up to the Container vendor to determine how methods in the unspecified transaction context operate. Now, in addition, you should turn your attention back to the table referenced previously. When an EJB's transaction attribute is Never or NotSupported (or Supported without an outer transaction context) the business methods also run within an unspecified transaction context. It is important to understand exactly what that means in WebSphere, and how to know what the behavior of methods running in the unspecified transaction context will be.

In WebSphere 5.0, there are extended transactional attributes that apply to the unspecified transactional context. The three settings we have to understand are:
- Boundary (Bean_Method or Activity_Session)
- Resolver (Application or Container_At_Boundary)
- Unresolved action (Commit or Rollback)

Let's leave aside the issue of activity sessions for the moment. So, for the moment, just go along with this and we'll discuss what happens when you set Boundary to Bean_Method. Basically this means that all resource manager local transactions [RMLTs]— (we'll call them local transactions from now on) must be committed within the same enterprise bean method within which they are started.

What does Resolver mean? In short, Resolver specifies resolution control and determines who is responsible for handling the commitment of statements that are left hanging by being called within an unspecified transaction context. The two options for Resolver are Application, which means that your program is responsible for forcing commitment [either by using setAutocommit(true) or by using LocalTransaction.begin() and LocalTransaction.commit()], and Container_At_Boundary, which means the container is responsible for committing the local transaction.

If you set Resolver to Container_At_Boundary (and set the Unresolved Action to Commit), then the bean's method will act the same as it would if you had set the bean's transaction attribute to RequiresNew. That is, the container will begin a local transaction when a connection is first used, and the local transaction will commit automatically at the end of the method.

The difference is that this method will execute in a local transaction context, meaning it won't tie together two different data sources into a single 2-PC transaction within the same method. Likewise, you can't carry a connection over into a method that is being used in this way. You must obtain the connection, use it, and close it all within the same method. Any attempt to pass a connection carried over from another transaction context into a method set up in this way will throw an exception.

Things are a bit more complicated if you choose to set the Resolution Control to Application.

Now the behavior of the local transaction depends upon what your code does. If your code specifies the behavior of each local resource [if you use setAutoCommit(true) in JDBC], then each statement will run in its own local transaction. Another option would be that you could delineate the transaction yourself by using javax.resource.cci.LocalTransaction.begin() andjavax.resource.cci.Local Transaction.commit() [or rollback()]. The interesting bit occurs if you do neither of these things, and leave a transaction open or hanging. This could happen in one of two ways: either you could use LocalTransaction.begin() without a corresponding LocalTransaction.commit() at the end of the method, or you could use setAutocommit(false) after obtaining your JDBC connection and not add any code to control the transaction. As you can see, the state of the transaction at the end of the method is now ambiguous.

To resolve that ambiguity, there is the Unresolved Action option. If Unresolved Action is set to Commit, then open local transactions will commit at the end of the bean method; if it is set to Rollback, they will roll back.

28.12 Dealing with Concurrency

Every application that uses a database in any form will at some point have to face an age-old question: How do I keep two different users from stepping on each other while they update their data? That is, if Tammy in the graphics department is updating our catalog to reflect the new look of our summer items, while Bob in accounting is also updating the catalog to reflect the new price list, how do we keep Bob's updates from overwriting Tammy's and vice versa? This comes down to the issue of managing concurrency and there are two general approaches: optimistic and pessimistic concurrency management.

Pessimistic concurrency management, probably the easiest to understand, is the idea of using a lock on a database record to keep more than one application from updating the database at the same time. So, at the beginning of Tammy's transaction,

ABOUT THE AUTHOR
Kyle Brown is a senior technical staff member with IBM Software Services for WebSphere. Kyle provides consulting services, education, and mentoring on object-oriented topics and Java 2 Enterprise Edition (J2EE) technologies to Fortune 500 clients. He is coauthor of Enterprise Java Programming with IBM WebSphere, 2nd Edition; Enterprise Integration Patterns: Designing Building and Deploying Messaging Solutions; WebSphere 4.0 AEs Workbook for Enterprise JavaBeans (3rd Edition); and The Design Patterns Smalltalk Companion as well as the author of over 40 articles and papers on J2EE, OO design, and design patterns.

E-MAIL
brownkyl@us.ibm.com

she obtains a lock on the catalog row. When Bob comes along, he may be restricted from reading the catalog row (if Tammy's lock was a lock on read) and forced to wait until Tammy is done. Another option would be a lock on write, meaning that Bob can read the original data, but he's restricted from writing new data to the row until Tammy's update completes (this would ensure that Bob's updates are additive to Tammy's).

changed requires one of two approaches. Bob could detect Tammy's update either by using a time stamp, which is applied at the end of each update, or by using an overqualified update, which is where you use the originally read value of every column in the table as part of the WHERE clause of your update statement. In this case, if there are any mismatches (due, for instance, to Tammy's update), the WHERE clause

a general hint.

There are seven different settings for Access Intent in WebSphere 5.0. To begin with, we have the two possible optimistic settings:

*   wsOptimisticUpdate—Use this when you want to allow a method (or group of methods) to perform updates but use optimistic concurrency. This will not perform any locking on select statements, and will perform an overqualified update at the end of the transaction if any set() methods are used during the transaction.
*   wsOptimisticRead—Use this when you want to allow one or more methods to only read from a database. If you attempt to perform any updates [e.g., if you send a set() method] during the execution of the transaction, a Persistence ManagerException will be thrown.

> ## "The cool thing about the new access intent approach in WebSphere 5.0 is that it also abstracts away the details of picking the right isolation level for each particular database"

The main problem with the pessimistic approach is the waiting. If several readers are kept from reading a row while another holds a lock that might result in an update, then this may lead to unacceptable runtime performance. In this case, the readers are needlessly waiting for a write that might never occur. To avoid this, another option is the idea of optimistic concurrency. This involves two things: (1) not obtaining locks, thus allowing for maximum concurrency in reading, and (2) performing a read immediately before a write to ensure that the data has not changed in the interim. If the data has changed, the writer will abort the writing process.

In our scenario, Bob would read his row at the beginning of his transaction, getting the original row without Tammy's updates. At the end of his transaction, under most circumstances, he would read the row again and discover it had not changed, and then complete the update. In some cases, he might read the row, discover Tammy's update, and abort his attempt to write the row since he would overwrite Tammy's intervening update in the process.

Detecting whether a row has

of the SQL UPDATE statement will not locate that row and fail. The major advantage of optimistic concurrency control is that since it doesn't require locking, it allows for much better throughput—at the cost of some number of aborted updates when collisions occur.

28.12.1 Concurrency and EJBs

How is all this managed in WebSphere and WSAD 5.0? WebSphere 5.0 simplified the process by combining all this into one setting now called access intent. You now define one or more access intent policies to apply to a set of entity EJB methods that will control both the concurrency scheme used (optimistic or pessimistic) and the locking strength used in a pessimistic scheme.

The cool thing about the new access intent approach in WebSphere 5.0 is that it also abstracts away the details of picking the right isolation level for each particular database; because different databases have different locking semantics, the access intent setting allows the container to pick the right isolation level based on

In addition, there are the five pessimistic settings: wsPessimisticRead, wsPessimisticUpdate-Exclusive, wsPessimisticUpdate-NoCollisions, wsPessimisticUpdateWeakestLockAtLoad and wsPessimisticUpdate. The setting wsPessimisticRead is nearly identical to wsOptimisticRead other than the fact that the underlying database isolation levels are set slightly differently (wsPessimisticRead sets the isolation level to RepeatableRead instead of ReadCommitted as it is in wsOptimisticRead). As is the case with wsOptimisticRead, if an update is attempted, the container will throw a PersistenceManagerException. Note that the isolation level setting for this is different in Oracle.

The four choices for pessimistic updates differ in both the isolation level used and in the approach taken to using the FOR UPDATE clause on select statements. Here are the different choices for pessimistic updates:

*   wsPessimisticUpdate-Exclusive— Exclusive in this case means that your application needs exclusive access to the database rows it is using. This setting indicates that it will use the FOR UPDATE clause, and will set the isolation level to serializable. This setting will mean that you will not encounter either phantom reads or nonrepeatable

reads, and that the deadlock that is possible with the TRANSAC-TION_SERIALIZABLE level alone (without the use of a FOR UPDATE clause) will not occur. This is terribly expensive—this will force every transaction to wait in line to acquire a write lock at the beginning of the transaction and will hold all other transactions until this transaction completes.

- wsPessimisticUpdate-NoCollision—No collision means that the application should be designed such that no concurrent transactions are expected to access the same database rows. This setting (as in the previous) uses the FOR UPDATE clause but sets the isolation level to ReadCommitted.
- wsPessimisticUpdate-Weakest LockAtLoad—The default setting for WebSphere. WeakestLockAt Load is applicable only to those databases that support both read locks and write locks. If the database supports them both, a read lock is acquired when a row is accessed and the lock is escalated (promoted) to a write lock if an update is performed on the bean. This setting uses an isolation level of RepeatableRead, but does not use a FOR UPDATE clause. It will work pretty well with nearly every database except Oracle. We will discuss that further later.
- wsPessimisticUpdate—This setting uses a for-update clause on finder methods and sets the isolation level to RepeatableRead (as in WeakestLockAtLoad) except in Oracle.

Why is Oracle special? Because of the way that it implements its locking mechanism, Oracle does not support the TRANSACTION_REPEATABLE _READ isolation level in the same way as other databases (for example, DB2 or SQL Server). So, everywhere that in other databases the server would have used TX_REPEATABLE_READ the server instead has to use TX_READ_COMMITTED.

In addition, Oracle doesn't use locks in the same way as in other databases. In databases, there is usually a difference between read locks

and write locks. A read lock is shared; multiple processes or threads can read an item simultaneously. A write lock is exclusive; only a single transaction holds the lock on the item. In Oracle the weakest lock is an update lock. This becomes interesting when you consider using wsPessimistic Update-WeakestLockAtLoad in Oracle. As mentioned earlier, for Oracle the server has to use the TX_READ_COMMITTED for this setting. What's more, in order to maintain the semantics of the setting, the server must also use a SELECT…FOR UPDATE as well. This is only true of Oracle; no other database requires the server to use a FOR UPDATE clause for this setting. The problem is that in many cases using this access intent with Oracle will result in a runtime exception if the back-end datastore does not support the SQL Statement needed by this Access Intent. This is because certain types of SQL statements (for instance multiple table joins) cannot use the FOR UPDATE clause.

28.12.2 Choosing the Right Access Intent

Given all of this, which access intent should you use for your applications? If the question were simple to answer, then WebSphere wouldn't have included so many choices. The fact is this is a complicated question, often motivated by differences in the behavior of the databases themselves. So, let's work through some best practices, some decision points, and some recommendations to guide you.

First, what's the easiest route for access intent? In many cases you would have expected that to be wsOptimisticUpdate. In that case (as you remember), if an application writes to a database, and there are few expected collisions, then readers and writers entirely stay out of each other's way, and writers will only cause each other problems once in a blue moon. What's more, it's even easy to write your domain logic so that if a transaction fails in this way you can restart the entire process if you captured the original data the user was operating on (perhaps using

the Command pattern).

However, optimistic locking is not appropriate in all cases, Sometimes you have to use one of the wsPessimisticUpdate variants. wsPessimistic-WeakestLockAtLoad will work in most cases with nearly every database; that's why it's the default. However, if you are using Oracle, it will fail in joins, and wherever the DISTINCT keyword is used, so you would have to move back to using wsPessimisticUpdate-No Collision. However, this policy doesn't ensure data integrity. Since it doesn't hold locks, concurrent transactions can step on each other and overwrite each other's data. So, either you can live with that option (maybe by ensuring that you don't get simultaneous transactions against a row through some other approach) or choose to live with wsPessimisticUpdate-Exclusive, which would serialize access to each row for both readers and writers. In some applications, this would be a significant performance problem. In others, it wouldn't. Your mileage may vary.

Finally, there is a difference in how you set up the optimistic predicate in WebSphere 5.0.0 and 5.0.1. In WebSphere 5.0, by default, all non-binary columns were added to the predicate (as we saw was true in the 4.0.2 version as well). The problem here was that this is slow since typically not all the columns will be indexed. Also, this can lead to problems because now the predicate is too constrained. In version 5.0.1, the default is instead not to add any columns to the predicate, meaning that for optimistic locking, no locking will take place. Instead, you need to manually set which columns are part of the predicate by selecting the mapping of a column in the overview of the WebSphere Studio map editor and then setting the OptimisticPredicate property to true.

Setting up access intents for an EJB in WebSphere Studio is actually very simple. We covered this (in another context) in Chapter 25. The process is the same. ⊕

EAI & EAM help you get the most from WebSphere

Maximizing WebSphere

BY DANIEL **MAGID**

Ah, the good old days of data processing – the days when Unix developers worked on Unix, Windows developers worked on Windows, and mainframe developers stuck to the big iron. In those days, each group could gather around its own water cooler and make fun of the developers working on the other platforms.

ABOUT THE AUTHOR

Daniel Magid, president of Aldon Computer Group, is a recognized authority on software configuration management and software development in general. Daniel has spent over a decade writing and speaking about, as well as designing solutions for, the software development process.

E-MAIL
Daniel.Magid
@Aldon.com

The Unix team would make fun of Windows' reliability; Windows developers would make fun of the mainframe's outmoded technology; and the mainframers didn't care because they controlled the company's mission-critical business applications. Developers simply never talked to each other. Then along came WebSphere and everything changed.

WebSphere provides a standards-based application environment in which data and functions can be shared across applications. Empowered users access necessary functions and data regardless of the data's location or the application's operating platform. Users can enable their CRM system to display current order status from the billing system, work-in-progress information from the production control system, or the order history for a particular customer across all divisions and subsidiaries. Today's IT buzz is about how this form of greater enterprise application integration (EAI) can dramatically improve the productivity of an organization by providing universal, secure access to critical business functions and data to end users through simple, familiar interfaces. WebSphere provides the infrastructure to make EAI happen.

WebSphere and the Challenges of Greater EAI

To successfully implement EAI using WebSphere, however, requires changes in how IT has traditionally operated. As usual, making things simpler for end users complicates the work of IT. IT staff members can no longer operate in their isolated, platform-based silos. Suddenly, everyone has to communicate and collaborate. Changes to the mainframe database might require changes to the Unix applications using the data or to the Windows applications displaying it. New application functions need to be announced broadly, allowing interested users to take advantage of them and to avoid needlessly duplicating work in the future. Enhancement requests and bug reports in shared systems must then be centrally organized and prioritized for implementation. Deployments of updates may need synchronization across platforms. The list goes on.

These complications might concern IT less were they not already struggling to provide a high level of service to the enterprise. Currently, according to Aberdeen Consulting (and known intuitively to most of us in IT), 90% of completed IT projects are late and 30% are cancelled prior to completion. According to Gartner, 50% of IT projects come in over budget (by an average of 44%) and 50% fail to meet their intended business objectives. Addressing those problems while taking advantage of the new integrated application environment requires a change in how application projects are managed. So now enterprise application integration requires enterprise application management (EAM).

The Case for Greater Enterprise Application Management

To deal with diverse development teams, a multitude of platforms, and the resulting integration issues WebSphere introduces, many IT organizations have started down the road to greater process management. According to META Group, 5% of Global 2000 companies currently have an integrated IT governance program under way. They expect that percentage to increase to 40% by the end of 2004 and to 70% by the end of 2006. Although most current methodologies and tools employed for managing the WebSphere development process are ill-suited to providing full-fledged EAM (see Table 1), a new generation of practices and solutions are now targeting this need.

The development shops now actively applying EAM to WebSphere are, in reality, adapting time-tested, documented practices of manufacturing process management for software development. The ideal EAM setup allows IT management to describe the rules they want applied to actions and interactions during the application development, integration, and maintenance process, and then enforces and automates those rules across all environments and teams. At its simplest level, EAM means freeing programmers from repetitive clerical tasks that might otherwise be required to avoid introducing defects. On a more sophisticated level, it means establishing and enforcing the

rules during the entire development life cycle, including software development, testing, and support. As the process evolves, automated documentation of the activities enables the organization to continuously evaluate and refine development practices (see Figure 1).

The objective is not to become bureaucratic, but to establish practices that have repeatedly proven to increase the output and quality of most development groups. Adopting recognized standards of process management (as reflected in EAM) and the myriad tools to support them, programmers have frequently found that instead of constantly working on damage control for problems caused by greater personnel and systems integration, they now have more freedom to engage in the creative aspects of their profession. The crucial challenge is to put in place the appropriate combination of personnel, procedures, automation, and structure that will allow IT to continue to rapidly deliver high-quality applications and leverage WebSphere's entire feature set.

A Basic Outline of Enterprise Application Management

There are several elements critical to successful enterprise application management. They are:
- Total application lifetime management
- Enterprise inventory management
- Life-cycle process automation
- Release management
- IDE integration

TOTAL APPLICATION LIFETIME MANAGEMENT

Most of the tools used for manag-

FACILITATE ON-TIME AND ON-BUDGET DELIVERY OF APPLICATION CHANGES ACROSS THE ENTERPRISE BY:
- Tracking original requirements and estimates
- Tracking all work in progress against milestones
- Notifying managers of potential problems
- Providing simple query mechanisms for real-time updates on the project backlog
- Tracking changes to specifications
- Providing managers with reporting mechanisms for use in evaluating past performance and correcting problem areas

IMPROVE SERVICE LEVELS BY:
- Improving response times
- Providing a central location for storing requirements that is accessible by end users and IT staff so everyone understands project aims
- Reporting regularly on project progress
- Improving forecast accuracy

INCREASE DEVELOPMENT PRODUCTIVITY BY:
- Automating administrative processes
- Eliminating errors from the code promotion process
- Allowing developers to work in whatever IDE they wish, while ensuring that the changes remain under code control

REDUCE PRODUCTION APPLICATION FAILURES BY:
- Enforcing user-defined testing processes
- Automating the packaging and movement of changes to ensure all parts are moved to the right place at the right time

**IMPLEMENT REGULATORY COMPLIANCE BY:**
- Implementing consistent move-to-production processes across the enterprise
- Controlling access to application parts
- Automating and enforcing those processes
- Maintaining detailed records of historical activity for auditors
- Providing a single point of observation and control for all applications

TABLE 1: OBJECTIVES OF ENTERPRISE APPLICATION MANAGEMENT

---

ing applications today focus mainly on the initial development of an application. However, studies show that the typical IT organization spends four to seven times as much on maintaining an application as they did on its initial development. In an enterprise's WebSphere-run application, there will be multiple components representing many different stages of an application's lifetime, not just the initial development stage. A typical example is a mainframe COBOL application that has been running for years now trading information via XML and/or Web services

---

"As usual, making things simpler for end users complicates the work of IT. IT staff members can no longer operate in their isolated, platform-based silos. Suddenly, everyone has to communicate and collaborate"

FIG. 1: **SAMPLE APPLICATION-SPECIFIC LIFE CYCLES**
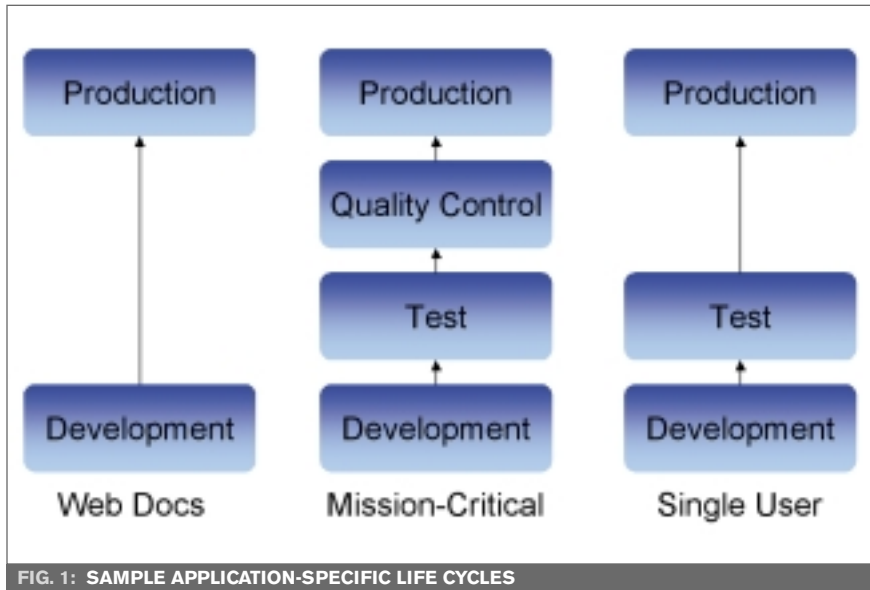
with newly created Java applications running on Unix or Windows servers. In order to manage these applications together, users must have a system that can assimilate work in progress across current applications as well as new development, and also tie those components together.

ENTERPRISE INVENTORY MANAGEMENT

The heart of any management system, whether it's a warehouse system, a payroll system, or an enterprise application management system, is a detailed inventory of all the parts it manages. Because the WebSphere environment can contain so many different parts from so many different platforms, building a central inventory poses a major challenge. The EAM system must keep track of the current location and status of all the application parts and control access to those parts regardless of file type or ultimate destination. Since these files can come from multiple platforms with varying storage architectures, the latest systems typically store these resources in a secure database repository such as BLOBs (binary large objects). Database functions like commitment control and constraints protect the database from corruption. Storing the files in a database significantly simplifies backup and recovery.

Once the files are stored in a database, an EAM system needs to provide

powerful functions for organizing and exploring the list of parts. Multi-platform WebSphere applications have extremely complex storage structures. In order to find the files of interest, a user might need to know the Unix directory structure, the Windows directory structure, the WebSphere directory structure, and the mainframe library structure.

Even developers can find it challenging to find files that way. But what about the managers, testers, operations people, and others who need to work with the application and who are likely to be unfamiliar with these storage structures? They will want to see all the files in an application regardless of the platform, directory, or library in which they reside. A QA tester might want to see all those parts currently in QA. A project manager might want to see where all the parts of a particular release or enhancement are in the development life cycle. Before a deployment, operations might need to review the list of parts and the target systems to ensure the installation is scheduled properly. An EAM system must maintain a metadata database with detailed information about the current location and status of every part under its control so that the user can find the list of parts they need easily.

LIFE CYCLE PROCESS AUTOMATION

The basic principles of total quali-

ty management teach us that in order to build anything of high quality – whether it is cars, houses, furniture, or software – you must first create a structured, repeatable, measurable process for building it. For application management, that means defining the entire process of what happens from the time a request for a change occurs until the time the solution is deployed into production. There will be workflow procedures like requirements gathering, design, review, and approvals, as well as code movement processes like checkout, promotion, and deployment.

The reach of an enterprise WebSphere application means that many people from different parts of the organization can feel the impact of a single change. To provide high-level customer service, the workflow rules must include notification to end users of the disposition of their requests and of system changes that will impact them. Keeping users up to date on the progress of IT projects (and delivering those projects on time) will go a long way toward reducing end-user anxiety and improving their perception of IT service levels. Putting in place approval procedures to ensure that ad hoc requests are not added to the project schedule without management acceptance will reduce project delays. Setting approval rules that allow end-user managers to sign off on moves to production that impact their departments will ensure that IT does not disrupt business operations by performing a system update at an inappropriate time. An EAM system can automate these processes so the appropriate people receive notification when they have a task to perform.

Once a change is approved, the project must move through the organization's move-to-production life cycle. Most of the first-generation tools for managing software development, like CVS, expect the management of that process to be either a manual function or a function automated by ad hoc user-written scripts. Both of those methods are potentially costly and error prone.

An EAM system allows users to define the rules that govern the move-

to-production process up front. From the beginning it automates and enforces those rules. The user need only identify what to move. The system knows where the parts are, where they need to go, how to deploy them, and how to install them. Since there is always a trade-off between the amount of testing and the speed to production of a change, users can define life-cycle processes that make sense for each application. IT organizations implementing this kind of structured, auditable process ensure they always know about every change and how it got into production.

RELEASE MANAGEMENT

In an enterprise application, managing releases can take many forms. For example:
- *Local customization:* If an organization has locations in different states or countries, applications will probably require some level of local customization.
- *Mixed short-term and long-term projects:* When maintaining

With an effective EAM system, all the user needs to know is which release requires work. The EAM system knows where the parts are, where to get the appropriate versions for checkout, where to move the code on check-ins and promotions, and where to deploy all the files. The system will automatically move the parts through the appropriate testing stages for the release to which they are associated. It will take care of notifying users when a change in one release must be reflected in other releases. And it will provide tools, as necessary, for merging releases together.

IDE INTEGRATION

There is an adage that says, "if the software process management system is not easy to learn and use, programmers will be very creative in not using it."

In the WebSphere environment, it is critical that developers can work from within the WebSphere development environment. Fortunately, IBM anticipated that

underlying management process. Developers get all the advantages of change control without sacrificing the ease of use and productivity of working in their IDE.

In fact, this is really what an EAM system provides for everyone. The idea is that each member of the team can access the system through a view that makes sense to him or her. Developers can see things through their IDE, project managers can look at a project-based view, operations can view things from a deployment package perspective, and executives can see a high-level summary of activity. All they need to know is what functions they want to perform or what information they require.

CONCLUSION

An enterprise application management system is the necessary infrastructure that will allow organizations to take advantage of the promise of enterprise application integration offered by WebSphere. An

"At its simplest level, EAM means freeing programmers from repetitive clerical tasks that might otherwise be required to avoid introducing defects"

mature applications there will be times when it will be necessary to manage large updates while still allowing for small maintenance changes to occur. These changes will need to move through different life cycles to ensure they don't interfere with each other.
- *Staged moves to production*: In an enterprise-level Web application, it may be desirable to gather sets of enhancements and fixes together into a package that will be deployed as a unit. There may be several of these packages in development simultaneously. These packages must move through the life cycle as units of work. They may not necessarily be moved serially, allowing users to deploy them as needed.

requirement and built the WebSphere tools on top of the Eclipse Framework. Developers or vendors that want to integrate with the WebSphere tools simply need to create plug-ins to the Eclipse environment. An EAM system should be implemented as an Eclipse "team repository," meaning users can access the parts or resources under management through any standard Eclipse perspective using standard Eclipse functions. Checking out, promoting, adding, refreshing, deleting, etc., are all likely accomplished by simply right-clicking on the desired files in the chosen perspective and selecting the appropriate option from the pop-up menu. There is no need for developers to learn or understand the whole

EAM system provides a single point of observation and control that allows users to implement consistent processes across the enterprise and provide management with the information they need to meet their quality and delivery objectives. By automating the day-to-day administrative functions, it frees up developers to concentrate on building applications. Its central repository of information about application development and maintenance efforts ensures that everyone is aware of activity that might impact them. In a complex WebSphere environment, an EAM system serves as the plumbing that will allow IT to consistently deliver high-quality applications that meet end-user requirements on time and within budget. ⊕

SmartPath MMS 5.0 Available for IBM WebSphere

(Morrisville, NC) – SmartPath Inc., a provider of marketing resource management (MRM) software applications, has announced that its comprehensive MRM solution, SmartPath MMS 5.0, is available for companies on the IBM WebSphere 5.0 Application Server platform. With capabilities specifically designed for WebSphere users, SmartPath MMS works with existing business processes and provides tighter integration between the planning, budgeting, and execution of marketing initiatives to meet the "on-demand" marketing needs of IBM customers worldwide.

With the widespread adoption of IBM WebSphere, major enterprises are demanding that the technology they rely on to handle marketing operations and management be integrated with the leading application server. SmartPath provides deep MRM functionality, which enables marketers to optimize core functions, including information management, operations, and team communication, from the initial strategic planning and creative development phases right through program execution and final evaluation, according to the company. SmartPath's marketing resource management capabilities delivered through WebSphere will

provide IBM customers with the power to better drive efficiency and marketing.

"As SmartPath becomes the most widely adopted MRM platform, supporting IBM WebSphere aligns SmartPath with global customer organizations that have large-scale marketing departments and that take technology and marketing standards quite seriously," said Michael Doernberg, president and CEO of SmartPath Inc.

"SmartPath provides an innovative and comprehensive set of MRM functionality with the flexibility to meet each customer's business process and marketing requirements. World-class organizations rely on SmartPath MMS to help them manage and execute marketing initiatives faster and with more effective use of resources."

SmartPath MMS 5.0 for WebSphere is available for immediate licensing.
www.smartpathinc.com

Context Media Delivers Expanded Support for IBM Platforms

(Providence, RI) – Context Media, Inc., has announced that its Interchange Suite content integration software provides additional support for multiple IBM platforms. As part of the expanded support, Context Media Interchange Suite is now powered by IBM WebSphere Application Server, IBM

Tivoli Directory Server, and IBM DB2 Information Management platforms.

Context Media has also announced that it has integrated Interchange Suite with IBM DB2 Content Manager and WebSphere Portal, allowing customers to directly import data from third-party content management systems onto their desktops through WebSphere Portal or Context Media user interfaces like Intershare, a Windows-based client that acts like a shared drive within any desktop application. As a result, users now have a single, unified view of their enterprise content assets, no matter where they reside.

In addition to announcing its expanded support for IBM platforms, Context Media has also expanded its relationship with IBM by joining the IBM Value Advantage Plus partner program to drive joint solutions in the small and medium business marketplace.

Context Media's Interchange Suite, powered by IBM WebSphere, DB2 Universal Database, and Tivoli Directory Server, is built on open Web services standards, enabling organizations to provide seamless and secure access to and management of both structured and unstructured digital

content, including rich media and text files, without disrupting workflows or existing technology investments, the company says.

The combination of Interchange Suite and IBM platforms allows businesses to aggregate digital content stored in multiple repositories and systems around an enterprise and present it to users through a single interface.

"We are working closely with IBM to deliver unfettered access to distributed enterprise content through the WebSphere and DB2 offerings," said Context Media Chairman and CEO Dan Harple. "Our work will help our mutual customers more easily leverage their valuable content assets, enabling them to be more efficient, compliant, and competitive."

Interchange Suite, powered by IBM WebSphere Application Server, DB2 Universal Database, and IBM Tivoli Directory Server, is currently available.
www.contextmedia.com

Bowstreet Announces Contribution to WS-I Working Group

Bowstreet, a provider of development tools for adaptive J2EE applications, has announced that their company was key to resolving incompatibility issues surrounding Web services standards such as SOAP, WSDL, and HTTP, as part of the Web Services Interoperability Organization's (WS-I) Scenarios and Sample Applications Working Group. Bowstreet is one of only 10 showcase vendors that built sample applications that interoperate with and conform to version 1.0 of the WS-I Basic Profile.

"WS-I's mission is to provide the real-world implementation leadership and support necessary for customers to deploy Web services," said Sinisa Zimek, chair of the Sample Applications Working Group at WS-I. "The progress made by the sample applications group is another significant step forward for WS-I and its 170-plus member organization. Bowstreet contributed advanced

capabilities to the applications group with its ability to rapidly develop applications for J2EE."

WS-I is an open industry effort chartered to promote Web services interoperability across platforms, applications, and programming languages. WS-I brings together a diverse community of Web services leaders to respond to customer needs by providing guidance, recommended practices, and supporting resources for developing interoperable Web services.

"As enterprises begin to take advantage of the true benefits of Web services, interoperability is essential," said Rose O'Donnell, vice president of engineering at Bowstreet. "Our involvement in WS-I has enabled us to directly impact the robustness of Bowstreet Portlet Factory in the production and con-

sumption of interoperable Web services, especially in the corner cases. We are very committed to continued collaboration with other companies in the Web applications space in an effort to establish the necessary parameters for Web services interoperability."

Bowstreet Portlet Factory supercharges WebSphere Portal with tools and technology for rapidly creating, customizing, maintaining, and deploying portlets, the company says. The ease of use and advanced development features of Bowstreet Portlet Factory dramatically streamline the entire portlet development process, enabling developers to deliver adaptive, robust portlets in a fraction of the time and at a fraction of the cost.
www.ws-i.org
www.bowstreet.com

Lay the Blame for Outsourcing Where It Belongs: With Management

BY DAVID **LANGE**

Reader David Lange felt compelled to respond to **WebSphere Journal** *Editor-in-Chief Jack Martin's editorial, "Outsourcing: Magic Bullet or Dirty Word? It All Depends on Your Perspective" (*WJ*, Vol. 2, issue 11).*

Let me offer the benefit of my experience with overseas Third-World programming skill sets, software design, and overall quality. My intent is to protect all of us from marginal to third-rate work unworthy of any of the firms that employ outsourced support.

The management types sell this overseas fodder as equivalent to or even better than homegrown talent. They've sold out their responsibility to the shareholders by creating subpar products that cannot be maintained, managed, or even improved upon. Usually the result is a throwaway piece of software that is pushed on customers for a short-term profit. Just look at the buggy, worthless products from respected vendors and the equivalent problems with outsourced support.

"The ability to develop and design software is more akin to an artistic talent"

The question comes to improvement and experience, which overseas coders will eventually gain. In time (a long time, in this case) despite their lack of vision, management will discover the damage they've wrought, and offshore coding will be frowned upon.

In the past two years I know of three Bay area venture capital projects by different startups that discovered – to their misfortune – the quality of outsourced projects. No, they're not around anymore.

Reality

1. Products of worth are not cheaply produced; there is no free lunch.
2. Software developers/designers are not churned out by diploma mills or trade schools. The ability to develop and design software is more akin to an artistic talent, no matter what management's myopic view of reality might be.
3. The cost of recoding a project is significantly greater than doing the task correctly from inception.
4. American expectations of profit and federal taxation have generated an environment in which investors and management have no provisions for long-term investment.

In the short term we're seeing the mass exodus of projects to faraway lands, but the following issues will arise:

1. *Quality:* It's not there – and never has been. Some 15 years of experience have shown me you can train people, but they either have the talent or they don't. You can't force-feed skill and talent to people. This only results in them banging away at the keyboard, producing a rather poor and confused imitation of a marketable product.
2. *Cost:* As these overseas IT people gain income, they'll want more and lose their attraction, which is based solely on price. The number of Indian outsourced projects that are being sub-outsourced to other nations proves this point. If greed isn't the motivating factor, then why are we risking the quality of the product for such a small amount of money?
3. *Control:* Management desires to be in control – and ego is the overwhelming motivation; in the long run there's no possibility that management will ever swallow the lack of control imposed by outsourcing. They'll want to be over there to micromanage product development – resulting in yet more costs – until it dawns upon them that they're no longer saving any money.
4. *Location:* Where do you think the talented people from these countries might actually be? Are they back home? Nope, anyone with any skill came here and is working at becoming a citizen. The United States has emptied out the talent pool, and left nothing over there but the dregs. But I'm forgetting the theory that you can create talent off an assembly line of diploma mills and trade schools. How shameful of me to forget something so obvious and undeniable.

It's time to face the music, people; we do live in interesting times. As an IT contractor, I'm not worried at this point, given the fact that people with skill and knowledge will eventually earn an income. I have not been without contracts the entire duration of the IT downturn, but I have noticed a rise in the number of shortcuts, the shoddy work, and the poor business sense of various management types. If you consider the sheer number of failed projects, the source of the problem is clearly management, not the intelligence or capability of American software developers.

ABOUT THE AUTHOR… David Lange is an independent consultant with some 15 years' experience in development, architecture, and administration with regard to Unix-based systems. David has survived projects dealing with scientific research, manufacturing, banking, and trading systems, but would rather spend his time pursuing operatic study, sports cars, and travel.

E-MAIL
dlange_scg@hotmail.com

oops.

N26MA

Forget something?

Post-launch is NOT the time to be verifying web applications.

The wild blue yonder of operational monitoring and management is extremely unforgiving. Which means that going live with the monitoring software you used in development is a great way to go dead—quickly! You simply can't support operations if your staff is drowning in details provided by development profiling tools and debuggers. **Let NetIQ cover your apps...with AppManager.**

AppManager—the industry's easiest-to-use Systems Management suite—is a proven management system for monitoring J2EE application servers, databases, operating systems and even end-user response time. NetIQ's AppManager monitors ALL application components—not just your server. **NetIQ. Nobody does UNIX better. Nobody.**

Visit us at www.netiq.com/solutions/web to learn how we can help you address the challenges of your operational monitoring and management.

net iQ
Work Smarter.