SYS·CON

NOW INCORPORATING XML JOURNAL

Web Services

.NET J2EE XML JOURNAL

Volume 4 Issue 5

Why are so many projects failing?

RETAILERS PLEASE DISPLAY
UNTIL JUNE 31, 2004

$9.99US $9.99CAN

05>

0 09281 03420 9

pg 22

Introducing OpenLink Virtuoso

A CROSS-PLATFORM Universal Server that transparently integrates disparate application databases and disparate applications across your enterprise. It provides a Virtual Database, XML Database, SQL Database, Web Application Server, and Web Services Platform as part of a **single-server solution.**

Empower your organization today, download a FREE evaluation copy from www.openlinksw.com/virtuoso or call us on 1 800 495 6322

Database Support
Oracle, SQL Server, DB2, Sybase, Informix, CA-Ingres, Progress, MySQL, PostgreSQL, and other ODBC or JDBC accessible databases.

Protocol Support
SQL, ODBC, JDBC, XML, XSL-T, XQuery, XPath, HTTP, WebDAV, NNTP and POP3

Platform Support
Windows, UNIX (all major flavors), Linux, Free BSD, and Mac OS X

Programming Languages
Any .NET bound language, Java, C/C++ etc.

Web Application Hosting
ASP.NET, Java Server Pages, PHP, Perl, and Python

Runtime Environment Hosting
Microsoft .NET, Mono, Java, Perl, Python, PHP, and others

www.openlinksw.com/virtuoso

OPENLINK SOFTWARE

Gartner

Application Integration & Web Services Summit

May 17-19, 2004
Los Angeles, California

FREE to every Summit attendee: Gartner 2004 Guide to Application Integration and Web Services Products and Vendors

Making Business Processes and Service-Oriented Architecture Pay Off

Topics covered:

- Application Integration
- Web Services
- Service Oriented Architecture & Applications
- Business Process Management
- Enterprise Information Integration
- Aligning IT and Business
- Portals
- Event Processing
- Measuring and Improving ROI
- Security Issues

and more!

Keynote Speakers

James Burke
Science Historian, Author, Producer

Peter Weill
Director of Center for Information Systems Research, MIT

Roy Schulte
Vice President and Distinguished Analyst; Conference Co-Chair

David Smith
Vice President and Research Fellow; Conference Co-Chair

Jeff Schulman
Group Vice President

- 40 in-depth sessions led by 30 leading experts
- Real world case studies
- The latest unbiased research and best practices
- Solutions from 50 leading vendors

Plus!

New Special Program
Enterprise Architecture:
New Challenges, New Solutions

and more!

Register Now: gartner.com/us/aiws
and use Promotion Code APNWSJ

Web Services Journal
.NET J2EE XML
XML Journal
MAY 2004 Volume 4 Issue 5

Inside WSJ

WSJ: Features

Managing the Bottom Line

Service-level management is essential to deploying business-critical Web services systems

By Fred Carter

22

Something Old & Something New

The key to Web services

By Jon Robert Pellant

14

ESB Technology & Innovation

Extending Web services with asynchronous message delivery and intelligent routing

By Robert Davies & Nigel Thomas

28

Information Integration Within the Enterprise Using XML

The XQuery language leverages XML for an effective real-time solution

By Julie Basu & Nirav Chanchani

48

WSJ: Product Reviews

Wisiba Web Service Management from itellix

A unique product that deserves consideration

By Paul T. Maurer 26

From the Editor

For Want of a Hammer

By Sean Rhody 7

Industry Commentary

BPM's Benefits

By Neal Novotny 11

WSJ Focus: Business Process

BPEL Unleashed

Putting a modern business process execution standard to work

By Doron Sherman 18

Closing the Deal with BPM

Move your client off the fence and into commitment

By David Cameron 20

WSJ: Management

Who Owns Web Services Management?

Where does it belong?

By Dan Foody 12

WSJ: Success Story

IVR for Web Services

Accommodating the mobile customer base

By Michael X. Zirngibl & Samuel Aparacio 32

WSJ: Standards

WSRP and the Enterprise Portal

A strong, and important, start

By Ed Anuff 38

Stateful Interactions in Web Services

A comparison of WS-Context and WS-Resource Framework

By Adam Blum 42

Heterogeneous XML-Based Information Integration

XML and XQuery for an effective real-time solution

BY JULIE BASU AND NIRAV CHANCHANI

48

Programming & Design: How to Reduce the Burden on Web Designers

Leveraging designer tag libraries with XSLT

BY PETER HORSFIELD

52

Semantic Mapping, Ontologies, and XML Standards

The key to managing complexity in application integration projects

BY DAVID S. LINTHICUM

56

Verifying Algorithms Through Unit Testing

Complicated algorithms are typically contained within one class or function so that developers can easily manage them. How then, can you test the functionality of your algorithms and verify that they correctly solve problems before you have a full application written? The answer lies in unit testing.

Unit testing is the perfect strategy for verifying and testing algorithms. Unit testing involves testing software code at its smallest functional point, which is typically a single class. Each individual class should be tested in isolation before it is tested with other units or as part of a module or application. By testing every unit individually, most of the errors that might be introduced into the code over the course of a project can be detected or prevented entirely.

I suggest that you investigate the benefits of performing unit testing to verify the functionality of your algorithms. Performing thorough unit testing reduces the amount of work you will need do at the application level, and drastically reduces the potential for errors.

By the way, unit testing is a fundamental part of Parasoft's Automated Error Prevention (AEP) Methodology. You can find more information about Automated Error Prevention at **www.parasoft.com**

*– **Adam Kolawa**, Ph.D.*
Chairman/CEO of Parasoft

DEPLOY SOA. NOW.

Service-Oriented Architecture. It's the buzz in the world of enterprise software development and integration: Web services and service-oriented architecture (SOA). And at BEA, we're all over it.

SOA—and the growing use of discrete, self-contained business services—has become widely recognized as the most relevant strategic direction for enterprise computing. And increasingly, the BEA WebLogic™ development environment is recognized as the premier platform on which to implement it. Right now.

As a ground shaking new architecture, SOA is making waves throughout many organizations. Come to BEA eWorld 2004 to see how easily you can deliver SOA in your organization NOW!

BEA eworld 2004

REGISTER NOW!

www.bea.com/eworld

MOSCONE WEST CONVENTION CENTER **SAN FRANCISCO, CA** **MAY 24-27, 2004**

Thanks to our generous sponsors

Diamond Sponsors

Platinum Sponsors

WebServices JOURNAL
.NET J2EE XML

INTERNATIONAL ADVISORY BOARD
Andrew Astor, David Chappell, Graham Glass, Tyson Hartman, Paul Lipton, Anne Thomas Manes, Norbert Mikula, George Paolini, James Phillips, Simon Phipps, Mark Potts, Martin Wolf

TECHNICAL ADVISORY BOARD
Bernhard Borges, JP Morgenthal, Andy Roberts, Michael A. Sick, Simeon Simeonov

EDITORIAL
EDITOR-IN-CHIEF
Sean Rhody sean@sys-con.com
XML EDITOR
Hitesh Seth
INDUSTRY EDITOR
Norbert Mikula norbert@sys-con.com
PRODUCT REVIEW EDITOR
Brian Barbash bbarbash@sys-con.com
.NET EDITOR
Dave Rader davidr@fusiontech.com
SECURITY EDITOR
Michael Mosher wsjsecurity@sys-con.com
RESEARCH EDITOR
Bahadir Karuv, Ph.D Bahadir@sys-con.com
TECHNICAL EDITORS
Andrew Astor aastor@webmethods.com
David Chappell chappell@sonicsoftware.com
Anne Thomas Manes anne@manes.net
Mike Sick msick@sys-con.com
EXECUTIVE EDITOR
Gail Schultz gail@sys-con.com
MANAGING EDITOR
Jennifer Van Winckel jennifer@sys-con.com
EDITOR
Nancy Valentine nancy@sys-con.com
ASSOCIATE EDITORS
Jamie Matusow jamie@sys-con.com
Jean Cassidy jean@sys-con.com

PRODUCTION
PRODUCTION CONSULTANT
Jim Morgan jim@sys-con.com
LEAD DESIGNER
Richard Silverberg richards@sys-con.com
ART DIRECTOR
Alex Botero alex@sys-con.com
ASSOCIATE ART DIRECTORS
Louis F. Cuffari louis@sys-con.com
Tami Beatty tami @sys-con.com

CONTRIBUTORS TO THIS ISSUE
Ed Anuff, Samuel Aparicio, David Cameron, Robert Davies, Nancy Esposito, Dan Foody, Mark Little, Paul Maurer, Neal Novotny, Savas Parastatidis, Jon Pellant, Sean Rhody, Doron Sherman, Nigel Thomas, Jim Webber, Michael X. Zirngibl

EDITORIAL OFFICES
SYS-CON MEDIA
135 CHESTNUT RIDGE ROAD, MONTVALE, NJ 07645
TELEPHONE: 201 802-3000 FAX: 201 782-9637
WEB SERVICES JOURNAL (ISSN# 1535-6906)
Is published monthly (12 times a year)
By SYS-CON Publications, Inc.
Periodicals postage pending
Montvale, NJ 07645 and additional mailing offices
POSTMASTER: Send address changes to:
WEB SERVICES JOURNAL, SYS-CON Publications, Inc.
135 Chestnut Ridge Road, Montvale, NJ 07645

For Want of a Hammer

WRITTEN BY
SEAN RHODY

The saying goes, "when all you have is a hammer, everything looks like a nail." It comes from carpentry, where it implies a certain amount of limited capability for a craftsman, but it has applicability in a wide range of situations. In particular for the IT industry, it denotes that someone doesn't have all the tools they need to do a job well. Not that the job will not get done, because we all know the jobs get done. But sometimes doing so requires heroic efforts.

At times, it feels like you only have a hammer when it comes to the use, and even awareness, of business process management, or BPM. Why that should be is not clear – perhaps the early tools didn't function well enough, or were hard to integrate.

Whatever the reason, BPM is one of the tools that's often put on the back burner while companies spend time and money doing it by hand. It may be an accounting problem – quantifying the cost of doing it the way it's always been done is sometimes difficult. But that cost can be enormous if you include the cost of lost opportunities caused by the inability to respond to business change in real time.

Think about mortgages for a second. Rates are at an all-time low, but they change quickly, and there's no guarantee that you can take advantage of the rate if you don't act in a timely fashion. Now what would happen if you had to make a code change, run it through QA, and deploy it in order to get that good rate you're hoping for? You know the answer – you'll be paying a lot more for that house.

It's the same with business opportunities. The market doesn't care that you have hard-coded business rules and a complex deployment scheme that takes you six months to move a change into production (don't laugh, this is real for many companies). The market changes when it changes.

That's why a BPM capability is so important. It's not just about the cost of making a programming change and deploying it – that's expensive enough. It's about the cost of all the opportunities missed because the software that you have is not agile enough to actually allow you to react to the market and move with it.

Admittedly, BPM requires a change in the way we think about software – but then again, so does Web services itself, and most of us wouldn't argue that the changes are for the better. It takes a new twist on thinking to start to abstract the business logic out of the code and into a descriptive process language, but it's not so hard to make that twist.

Part of it comes down to knowing what to make a service (something that's coded) and what to make a rule (something that's changeable). BPM experts will talk about managing a service or services using the term "process." Services may be individual steps in a process. You want to be able to define processes in a simple, business-like way that is intuitive to the folks that make business decisions.

Which means that services have to be developed in ways that make sense to business users – in other words, the API has to be understandable and usable to those who aren't programmers. Not that programming is necessarily going to go away – in fact the task of making services intelligible argues for increased programming savvy. What's needed are the super-coders; IT experts who also understand the business logic and can design intelligent APIs for the use of the business, not the IT department.

SOAs and Web services are paving that road for BPM today. Finally, we have a simplified API language (note that I didn't say simple, it isn't) that the right tools can use to allow business analysts to define processes. And we're starting to see tools that allow this definition in a fairly straightforward manner.

Not that it will be as simple as writing an English sentence such as "apply a 10% discount for orders over $100,000," or "include an overseas disclaimer for all orders placed outside the U.S." It's not going to be natural language processing. But it will be on the order of wiring services together instead of having to code them. And that's progress. That's capability, and the ability to see the task as more than just banging a nail into a board with a hammer. Finally, a screwdriver – and a power one at that.

About the Author

Sean Rhody is the editor-in-chief of *Web Services Journal*. He is a respected industry expert and a consultant with a leading consulting services company.
Sean@sys-con.com

Mindreef News
Volume 2 May 2004 www.mindreef.com

Retooling for a Service Oriented Application Lifecycle

Service Oriented Architecture

Service Oriented Architecture (SOA) promises to reduce costs, improve productivity, ease integration, and lead to platform and technology independent software. SOA-based applications are collections of services with well defined interfaces that are loosely-coupled and communicate with one another over network protocols.

Web services are the current technology of choice for developing and deploying applications following the principals of Service Oriented Architecture. Web services certainly changes how we develop applications, but what other impact does it have on the software development lifecycle?

Composite Applications

SOA introduces "composite applications" which deliver overall business functionality from a collection of cooperating services. The individual services making up the composite applications might be highly distributed across different physical machines and may even comprise services spanning different organizations.

The implications of an application spanning organizations are staggering. We no longer have control over deployment versions and timing. We don't have the luxury of attaching a source-level debugger to every component of our application. We also lose the opportunity for pre-deploy integration testing of the complete application. SOA and composite applications truly change the traditional software lifecycle. We now have to consider and accommodate continuous deployment and integration.

Continuous Integration

With loosely-coupled Web services we lose the traditional "integration" phase of application development. Services are discovered at runtime, integrating new components that we may have never encountered before.

For example, a manufacturer might require all parts providers to implement certain Web services to conform to their procurement standards. The manufacturer deploys their procurement system, but vendors come "online" one-by-one as they complete the implementation of their individual services.

The manufacturer may not have the luxury of integration testing with every individual vendor, much less integration testing of the entire composite application consisting of services from a variety of partners.

The challenges of continuous integration are compounded by versioning. Every partner delivering services for the overall composite application may update or upgrade their portions on different schedules. Service interfaces may need to be updated but not everyone will upgrade at the same pace.

The implications of an application spanning organizations are staggering

Lifecycle Changes

SOA suggests a dramatic impact on our traditional software development lifecycle. We will refer to this modified development lifecycle as the Service Oriented Application Lifecycle (SOAL). So, how does SOAL change the development lifecycle?

Individual services certainly may be developed with a traditional "design, develop, integrate, test" lifecycle. The lifecycle changes as we combine these services into composite applications, especially when those services are outside of our control. Now, how do we test this composite application? What is the deployment process for a service that has many dependent services and also itself depends on other services? How do we deal with versioning issues? How do we debug composite applications made of many distributed services?

We certainly don't have all the answers today, but the SOAL does suggest a need for a new breed of tools and approaches to testability of composite applications.

Service Oriented Lifecycle Tools

Traditionally we had great control of our application development lifecycle. Our tightly integrated stand-alone applications gave us complete control over development, testing, integration, deployment and versioning. The introduction of loosely-coupled applications with Service Oriented Architecture produces new lifecycle challenges. SOA has changed how we architect, design and develop software. SOAL also changes how we integrate, test, deploy and version our software systems. The Service Oriented Application Lifecycle will require some retooling to accommodate these changes. SOAPscope is the first in a line of products meeting the needs of the Service Oriented Application Lifecycle.

SOAPscope was designed to be used in environments that span organizations. We recognize that you can't attach a debugger at every node of your composite application. SOAPscope's approach is to debug at the message level, not the source level. Composite applications are non-invasively monitored for SOAP messaging on the wire.

SOAPscope also offers a suite of tools providing value to the developer and tester. These tools, applied to both SOAP and WSDL, allow detection of problems, recreating a bug, testing for boundary conditions, detecting errors due to versioning, and analysis for interoperability issues. SOAPscope has proven itself as a valuable tool for developer and testers, usually paying for itself in its first use.

But SOAPscope is not just a development tool, it is a Service Oriented Application Lifecycle tool. It goes beyond debugging and testing during development to provide value during the deployment and maintenance of a production system.

Imagine that your Web services management system detects a fault: what is the next step? Recreate the problem against a staging server to help to debug and fix the problem. SOAPscope allows you to capture the context of the problem, readily recreate it, and determine the root cause of the problem.

Another use for SOAPscope in production is to proactively detect potential problems.

For example, you might discover with SOAPscope that a particular service or client within the composite application is not truly WS-I Basic Profile compliant. That portion of the application can be dealt with before any real errors surface.

Think of SOAPscope as a tool for continuous quality throughout the new Service Oriented Application Lifecycle. Try SOAPscope for FREE today at www.mindreef.com.

"SOAPscope rocks!"
- Don Box, XML Messaging Architect, Microsoft and co-inventor of SOAP

LinuxWorld® Conference & Expo

CONFERENCE: August 2 – 5, 2004 EXPO: August 3 – 5, 2004

MOSCONE CENTER • SAN FRANCISCO, CA

Where OPEN MINDS Meet

LinuxWorld Conference & Expo is the #1 venue for decision-makers and influencers to discover real business solutions for real business problems, learn how Linux is accelerating as a total enterprise solution, and understand how it can be applied effectively to save their companies time and money.

Attend LinuxWorld and...

- Realize Linux as more than an operating system, but as **a world of applications**
- Evaluate the **latest applications and innovations** from leading open source companies
- Explore interoperability issues and opportunities in **open source and proprietary environments**
- Stay on the cusp of **emerging technologies** and the acceleration of open source adoption in enterprise computing
- Review the latest open source initiatives, their deployment and successes to help **make informed decisions** for your company

WWW.LINUXWORLDEXPO.COM

Register Online With Priority Code: D1401

CORNERSTONE SPONSOR

ORACLE®

PLATINUM SPONSORS

DELL™ IBM intel Novell®

IDG WORLD EXPO

WebServices JOURNAL
.NET J2EE XML

PRESIDENT AND CEO
Fuat Kircaali fuat@sys-con.com
VP, BUSINES S DEVELOPMENT
Grisha Davida grisha@sys-con.com
GROUP PUBLISHER
Jeremy Geelan jeremy@sys-con.com

ADVERTISING
SENIOR VP, SALES & MARKETING
Carmen Gonzalez carmen@sys-con.com
VP, SALES & MARKETING
Miles Silverman miles@sys-con.com
ADVERTISING DIRECTOR
Robyn Forma robyn@sys-con.com
DIRECTOR, SALES & MARKETING
Megan Mussa megan@sys-con.com
ADVERTISING SALES MANAGERS
Alisa Catalano alisa@sys-con.com
Carrie Gebert carrie@sys-con.com
ASSOCIATE SALES MANAGERS
Kristin Kuhnle kristin@sys-con.com
Beth Jones beth@sys-con.com

SYS-CON EVENTS
PRESIDENT, SYS-CON EVENTS
Grisha Davida grisha@sys-con.com
CONFERENCE MANAGER
Lin Goetz lin@sys-con.com

CUSTOMER RELATIONS/JDJ STORE
CIRCULATION SERVICE COORDINATORS
Shelia Dickerson shelia@sys-con.com
Edna Earle Russell edna@sys-con.com
Linda Lipton linda@sys-con.com

SYS-CON.COM
VP, INFORMATION SYSTEMS
Robert Diamond robert@sys-con.com
WEB DESIGNERS
Stephen Kilmurray stephen@sys-con.com
Christopher Croce chris@sys-con.com
ONLINE EDITOR
Lin Goetz lin@sys-con.com

ACCOUNTING
FINANCIAL ANALYST
Joan LaRose joan@sys-con.com
ACCOUNTS RECEIVABLE
Charlotte Lopez charlotte@sys-con.com
ACCOUNTS PAYABLE
Betty White betty@sys-con.com

SUBSCRIPTIONS
SUBSCRIBE@SYS-CON.COM
1-888-303-5282
For subscriptions and requests for bulk orders, please send your letters to Subscription Department
Cover Price: $6.99/issue
Domestic: $69.99/yr (12 issues)
Canada/Mexico: $89.99/yr
All other countries: $99.99/yr
(U.S. Banks or Money Orders)

Worldwide Newsstand Distribution
Curtis Circulation Company, New Milford, NJ
Newsstand Distribution Consultant:
Brian J. Gregory / Gregory Associates / W.R.D.S.
732 607-9941 - BJGAssociates@cs.com

For list rental information:
Kevin Collopy: 845 731-2684,
kevin.collopy@edithroman.com;
Frank Cipolla: 845 731-3832,
frank.cipolla@epostdirect.com

SYS-CON MEDIA

BPM's Benefits: Tactical As Well As Strategic

The benefits of BPM (business process management) reach from the mail room to the board room and beyond.

BPM, with enhancements enabled by Web services, provides tactical benefits that significantly enhance application creation and extension, while delivering strategic benefits to companies that leverage BPM methodology and reinvent themselves as process-managed enterprises.

WRITTEN BY
NEAL NOVOTNY

Tactical Benefits of BPM

BPM systems employ graphical tools to connect existing IT assets, speeding applications and sophisticated workflow creation. These tools bring business users closer to their own business processes, from which they had been precluded previously. Extended at the process level, processes created with BPM systems are much easier to customize and maintain than the embedded processes in their ERP and legacy systems counterparts. BPM implementations have been shown to reduce application development costs up to tenfold and significantly lower process ownership costs.

Components of a BPM System

BPM systems offer an extensive set of services that improves the entire process life cycle – from concept to improvement. Described below, the basic BPM system components offer a rich set of functionality.

- ***Design:*** Business users model the intent of their processes. BPM systems directly access or import these processes, and IT specialists, programmers, or, ultimately, business users convert processes into executable functions.
- ***Orchestrate:*** IT professionals enable business process execution by connecting (or binding) graphical icons to IT assets in one or more applications or systems.
- ***Rules engine:*** The rules engine maintains the business rules, enabling access by multiple programs as well as creation and modification by business users.
- ***Integrate:*** Connectivity via Web services and adapters links internal, customer, supplier, and partner systems in a cost-effective, reusable, pervasive manner.
- ***Deploy and execute:*** Processes are deployed to the BPM system and run in its execution engine, which supports high-volume transactions that span traditional systems.
- ***Monitor and analyze:*** The process logic flow is explicitly laid out in the BPM system, which feeds data to dashboards and other systems, enabling real-time monitoring, analysis, and process control.
- ***Simulate and improve:*** The BPM system enables scenarios wherein processes are validated, compared with forecasted levels, and when necessary adjusted to ensure resources are adequate.

Access, Standardization, and Reuse

The above feature set alone is compelling. Enterprises can use BPM systems' graphical modeling tools out of the box to create business processes that can span applications and be analyzed and monitored in real time.

Manipulating processes has previously been largely off limits because their dynamic nature and numerous dependencies made interaction with them more difficult than with data. However, just as data was decoupled from systems 20 years ago, processes are being extracted today via BPM systems. Placing program flow logic in the BPM system orchestration tool and business rules in the rules engine extracts them from individual programs and makes them more easily reusable across the enterprise. Their newfound abstraction increases their accessibility. The BPM system is becoming the application for processes just as a relational database has been for data.

Processes will increasingly be stored in BPM systems and accessed by other programs and user productivity tools, analogous to data access today. New process modeling languages that standardize the format of the end-to-end business processes are enabling portability across systems and compatibility between interwoven enterprises. These languages build on the reconfigurability and dynamism of Web services standards and further BPM systems' delivery of standardized, cost-effective integration that is both faster and easier to extend and reuse.

–Continued on page 17

Who Owns Web Services Management?

Where does it belong?

When I tell customers that my company does Web services management, the question I often hear is "so what do you mean by Web services management?" It's no wonder there's so much confusion on this issue, because the term "management" has been used to mean many different things. For example, there's business process management (actively coordinating the runtime execution of business processes) and systems management (passively monitoring performance and availability of IT infrastructure). These are two very different meanings for the word "management," and two very different markets.

WRITTEN BY
DAN FOODY

The question of management in the context of "Web services management" has gained a lot of visibility as the big vendors each try to claim ownership of the space. Hewlett-Packard's OpenView division acquired Talking Blocks. Computer Associates acquired Adjoin. BEA Systems announced that its future WebLogic 9.0 release would focus on management. Are application platforms like BEA WebLogic and systems management tools like HP OpenView in the same market? Hardly, but both camps are trying to stake a claim in the space. Further confusing the issue is that most pure-play Web services management solutions do a bit of both.

Who's right? Does Web services management belong in the domain of the application platforms, does it belong in the domain of the system management platforms, or is it truly a new market? Industry pundits state with absolute certainty that they know the answer. Unfortunately, they don't all come up with the same answer! As you would expect, it's not as clear-cut as any one industry pundit might have you believe.

In a world where the way people design, build, and deploy applications remains status quo, there is a clear separation between the roles of application servers and systems management tools: application platforms run the applications, and systems management tools make sure the applications are running well. Application platforms are used by developers, systems management tools are used by operators – and the two camps rarely cross paths. In a status quo world, what then happens to pure-play Web services management vendors? In this world, consolidation is inevitable.

The open question is whether the status quo will remain, or whether enterprise IT is on the cusp of a fundamental change. Frankly, Web services alone are not going to be the cause of a fundamental change. Web services can be deployed in traditional multitier or client/server applications instead of traditional proprietary technologies, and result in improvements in cost and time-to-market, but this is just evolutionary use of a new technology. It's not revolutionary.

The fundamental change that's coming in enterprise IT is the move to service-oriented architectures, which in theory allow IT organizations to avoid unnecessary duplication by sharing software services, like order processing, across projects and teams.

Web services are the vehicle to make service-oriented architectures available to the majority of IT organizations. Through reuse and consolidation of redundant services, a properly implemented service-oriented architecture results in significant improvements in both IT cost and flexibility. This reuse and consolidation changes the rules of the game in one important way: it means that a software project is no longer self-contained – one IT project depends on others, which are maintained on different schedules by different organizations.

These cross-project dependencies result in new requirements that previously did not exist. Maintaining performance and availability of the overall service network not only requires central visibility into the operation of all of the services, regardless of what platform they are built on, but also active in-network manipulation of the routing, quality of service, and in some cases, content of messages sent across projects.

Heterogeneous central visibility is something foreign to application platforms but typical of systems management products. On the other hand, active in-network control is something foreign to systems management products but typical of application platforms. To coordinate in-network control across heterogeneous projects requires a careful blend of functionality drawn from both systems management and application platforms – this is the sweet spot for Web services management.

This blending of functionality is what is

–Continued on page 36

untangle your integration woes with

Assande EAI Messaging Suite

Assande EAI Messaging Suite features

- XML Schema Repository
- Data Abstraction, Transformation, and Mapping
- Message Assembly and Generation from Components
- Messaging Integration Service Deployment
- Automated Naming Standards, Version Control, and DIFF Functions
- Infrastructure Utilization and Pipeline Reporting
- Conversation Deployment and Message Release Management
- State Management and Integrated Workflow
- Advanced Search Capabilities
- Integration Documentation (MS Word, Excel) Repository

ASSANDÉ™
INTEGRATION MANAGEMENT COMPANY

http://www.assande.com

Something Old & Something New

The key to Web services

As soon as new business applications roll out, someone is waiting to change them. Most application developers are resigned to this reality because they understand that business conditions change, and the applications that support key business processes have to change with them.

If constantly rewriting application code kept pace with business needs, it would be worth all the accompanying hassles. However, rewriting application code never keeps pace with business change. Rewriting application code is inherently too slow and inefficient to keep up with the pace of business change. Markets and customer whims shift faster than the best engineer can work.

WRITTEN BY
JON ROBERT PELLANT

Web services make change more manageable by replacing application-level, data-gathering functionality. That means applications require less coding to link them with every back-room system that contains relevant data. In this role, Web services save programmers from recoding every time a company needs to make a relatively minor change to a process – for example, a health care company increasing the reimbursement percentage on a prescription drug.

Even with Web services doing the data collection, however, applications are still too unwieldy to support change at the pace of business. The problem is the limitations of Web services and the traditional software development model. The traditional software development model is data-centric. As long as that mindset holds sway, applications will always be too hard to change because they devote too much code to chasing data all around the enterprise. The key to more manageable applications that change with the pace of business is to stop building them around data and start building them around rules, using a combination of Web services and a rules-based business process management layer as the enablers.

Rules-based business process management systems on the market today contain functionality that enables business-level users, not programmers, to change business rules, processes, or workflows in response to market conditions. Paired with Web services to perform data collection functions, rules-based systems can reduce the typical 100,000-line application to about 10,000 lines. Although that's a dramatic drop, thinner applications are not the real benefit. The business objective behind them is systems that process transactions with little or no human intervention. To do that, they must be able to change. This ability hinges on the proper mix of Web services and a rules-based BPM "change" layer in the IT infrastructure.

Combining the How and the What

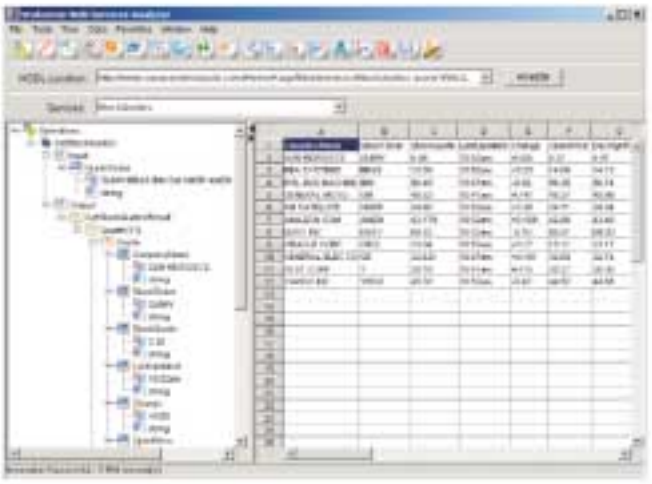
As a new technology with much-deserved credit for enormous potential, Web services appear at first blush to be the whole answer to rapid application change. However, their advantages go hand-in-hand with unavoidable limitations. Current Web services lack the intelligence needed to make agile processing decisions at the point of transaction. They can't make decisions based on the worth of the customer – such as whether to batch process an address change, which would delay a customer's order, or use a more expensive Web service to process a high-value customer's address change immediately.

Web services are an ideal mechanism for exposing processes to customers and partners. They aggregate information from various systems so applications don't have to. They do not, however, take the processing burden off the application, and as long as applications do most of the processing, they will resist change. Without intelligence and process automation ability, Web services are a limited solution. That's where "smart" business process management (BPM) comes in.

How to Eliminate Web Services Bottlenecks in Java Application Development

By Bob Brauer
bob.brauer@strikeiron.com

STRIKE IRON



The StrikeIron™ Web Services Analyzer for Java™ makes it easy to quickly discover, understand and utilize any Web service with a click of the mouse.

Reduce your costs and time to completion with a whole new level of flexibility!

The benefits of Web services are clear – lower integration costs, maximum reusability, faster deployment, more automation, easier to work with new partners, and so on. So, what about Java? …how can you leverage Web services to be more productive with your Java-based application development?

Identify development bottlenecks

First, identify the bottlenecks in your development cycle that slow you down. Is it finding or developing the right Web service for your needs? Is it understanding the behavior of a new Web service? Is it keeping up with Web service definition changes in your application? Is it inefficient testing procedures?

Identify solutions

The easiest and most productive way to solve these problems is with the right tools. Considering the bottlenecks above, the right tools:

- can be used with your current Java development environment,
- make it easy to visualize and understand the behavior of a Web service,
- require no additional software for faster evaluation and iteration,
- help create applications that require less support and are easier to maintain, and
- simplify your life!

Find the tools

The tools you need to break through your development bottlenecks are available today. StrikeIron is an innovator in developing new and easier ways to work with Web services. The StrikeIron Web Services Analyzer and the StrikeIron Web Services Developer Kit are the ideal tools for every developer working with Web services in a Java world.

Point, click and use!

The StrikeIron Web Services Analyzer allows anyone to easily and quickly discover, understand and utilize any Web service with a click of the mouse. Everything you need is easily understood in a graphical interface with familiar tree and table views.

The StrikeIron Web Services Analyzer

- Easy-to-use point-and-click interface.
- Graphically visualize any Web service.
- Faster and easier discovery with a Web services directory browser.
- Dynamically invoke any Web service to test behavior.
- No programming or compiling.
- Save your output to Microsoft® Excel.
- No need for additional software.

Build dynamic applications

The StrikeIron Web Services Developer Kit accelerates application development and lowers costs. It enables you to *dynamically* load, invoke, aggregate and govern the execution of multiple Web services at runtime, without recompiling and without additional software.

The StrikeIron Web Services Developer Kit

- A full copy of the StrikeIron Web Services Analyzer and all its benefits.
- Access to StrikeIron's WSDOT API for dynamic invocation and execution without recompiling.
- Complete, standardized, executable Java code generation, directly from an interactive environment!
- An Eclipse plug-in for graphical analysis of Web services and code generation within Eclipse.
- A rich set of additional functionality for building applications including Web service monitoring, batch capabilities, multiple workspaces and more.

Eliminate application development bottlenecks and become more productive!

- Dramatically reduce the time to understand a Web service's functionality.
- Quickly understand any Web service's behavior for faster utilization regardless of complexity.
- Significantly reduce your time and cost by eliminating the writing and compiling of additional code.
- Significantly reduce your support requirements through dynamic execution that is "self-adjusting". No need to re-write, re-compile, re-test and re-distribute.
- Spend more time creating applications and less time on maintenance and support!

Delivering Web Services to the World!

StrikeIron is a pioneer of innovative products and subscription-based online services that provide a new level of ease of use and productivity for the utilization of Web services.

Free Trial! Please visit our Web site at www.strikeiron.com to download our newest products and to sign-up for our free newsletter.

StrikeIron, Inc.
919.405.7010
sales@strikeiron.com
www.strikeiron.com

Smart BPM – BPM driven by business rules – can provide the missing intelligence needed to fully automate complex transactions. Smart BPM allows business people setting business priorities – not the programmers who implement them – to guide how rules influence the way work gets done. The ability to change business rules frequently without rocking the system is key to agility. They can add the intelligence and agility to Web services that companies need if they are to realize the full benefits of exposing key applications.

As Web services need BPM, so does BPM need Web services. Web services aggregate information – the "what" – and hand it up to the BPM layer. The BPM layer does the processing according to rules set by business users – the "how." When a business user needs to change a process, they make the change at the BPM layer in HTML. In the traditional software model, a programmer would have to make the change in the application code. That could take months, during which time transactions that should be routine have to be handled as exceptions.

A New Flavor of BPM

Traditional BPM alone is not enough to support mature and nuanced transactional automation. Smart BPM, built on sophisticated rules engines, joins process automation and best-practice rules to add fine-grained decision intelligence to Web services at the point of transaction.

Smart BPM automates work through business rules, workflow, and enterprise application integration (EAI) to automate the work that smart people used to have to do. On their own, workflow and EAI did little more than push around and assign work. Workflow applications ordered and streamlined processes, eliminating manual transfers and paperwork along the way. Rules-based BPM takes these concepts further by adding sophisticated rules engines to the mix to automate more complex business processes. With rules, smart BPM goes beyond workflow's limitation to documents or tasks on the one hand, or EAI's hard-coded, point-to-point rigidity on the other. Smart BPM automates both the decisions and the processes of how work gets done. Web services are in a good position to take advantage of the meeting of these technologies.

As Web services have matured, many organizations have exposed cost-effective business transactions and services to partners and employees. Some companies are using Web portals supported by XML and other Web service technologies to bypass application interfaces entirely, enabling users to submit expense reports without logging into an ERP system, or partners to change a shipping and receiving address.

The traditional Web service implementation model doesn't scale well in dynamic enterprises. The primary culprit is the aforementioned classical software development model. IT developers typically start with a problem to solve, then build their Web services application with primary consideration to their data model, instead of to the business processes they could be automating. In this approach, functionality that best exists at its own level ends up at the application layer, where it is most difficult to alter when changes inevitably arise. Soon, the development team is fighting a constant rear-guard action to keep up with changes the company has to make to stay competitive, but never quite catching up because the deck is stacked against them.

> "Web services make change more manageable by replacing application-level, data-gathering functionality"

In addition to the classic software development model's limitations, Web services have their own limitations. Typically, Web services lack the process intelligence that brings business-level discernment – the use of if/then logic to make sophisticated choices about how to process transaction input. One problem this causes is that the more Web services proliferate, the more challenging it becomes to ensure a given transaction applies the proper service (there may be several "change address" Web services available, for example, although one may be for partners and another for customers). The burden of choosing the right Web service in a given context rests with the application, which adds to the programmer's task.

Moreover, Web services are usually stateless. The management of policies, security, best practices, and so on rests with application developers – an open door for inconsistencies. And when business policies, practices, or rules change, portal developers must code the logic changes into the individual applications because their applications aren't smart enough to adapt on their own.

A better approach is to architect a smart BPM layer that abstracts the exposed Web services as a logical business process. Such a software layer applies global policies, business rules, and best practices to the entire portal platform. It manages and executes business processes centrally, tying exposed services into enterprise applications. It's smart because it takes advantage of advanced decision technology to resolve nuanced business rules in real time.

While the *how* of a process falls to Web service developers, smart BPM systems apply *who, what, when,* and *where* to the business context of a request. It executes the appropriate Web services via an integrated rules engine that models corporate policies, business rules, and best-practice knowledge.

Smart BPM Adds Agility

Smart BPM is crucial to Web services because business rules should be managed by the business, not programmers. Web services governed by smart BPM tailor their actions and responses to the applications calling on them, in effect creating custom services for each user. Intelligent systems leverage inference technologies to choose the correct process based on the context in which the process is used. This eliminates the need for IT personnel to aggregate a set of rules for a specific process, and makes the whole system more adaptable to change. Smart BPM systems scale well because IT staff can concentrate on adding

new functionality, exceptions, and Web services instead of hand-coding every rule change into each system it touches.

For example, an address change sounds like a simple Web service. But consider a large national bank. A "simple" address change may consume more than 70 process flows and involve close to 40 back-office systems. While many flows concern themselves with the "hows" of changing the address information in the back-office, many more will need to deal with elements like correspondence, fraud prevention, and auditing. The rest of the context involves delegated business rules for up-selling or cross-selling based on relative customer value, subscription status, demographics, marketing initiatives, change notifications and so on.

If a smart BPM layer manages all of these flows, the different portals involved don't have to know any of the address change details. If the bank changes a customer value policy, the rule is instantly enforced across the enterprise. There's no need for IT to aggregate the rule and distribute the change to multiple systems. This efficiency saves money and time, but its higher value lies in what it can do for Web services as a whole.

Smart BPM's rule-guided intelligence promises to simplify and coordinate Web services for real-world business agility, something that will be critical to all business in the 21st century. While exposing Web services is key to creating automated systems, true agility comes from the intelligent use of business rules, policies, and best practices. Smart BPM is one way Web services are adapting to their own momentum and success.

About the Author

Jon Pellant is the chief technologist for PegaRULES, Pegasystems' patented business rule development and deployment technology. In this role, he supports product technology, analysts, and key sales and partnering initiatives. Prior to joining Pegasystems, he managed the technical direction of Firepond's product suite. Jon holds a master's degree in science and electronic engineering from the University of Minnesota.

jon.robert.pellant@pega.com.

–Continued from page 11

Strategic Benefits of BPM

Beyond enjoying tactical benefits, companies that realign their organizational structure to exploit BPM's new cross-functional capabilities benefit as well. BPM bridges application and enterprise silos, driving companies to operate around a set of business processes as opposed to business functions or specific applications. BPM allows this to occur incrementally rather than starting from scratch with multimillion dollar, multiyear projects with ROI projected years into the future. ROI can be delivered and measured throughout the implementation, rather than waiting for a "big bang" that may never occur.

There are three major strategic components of BPM: cross-functional processes, process-driven enterprises, and the process feedback loop.

Cross-Functional Processes

BPM systems automate business processes that span applications, enterprises, and the extended enterprise. However, cross-departmental automation requires enterprise-wide managerial cooperation – especially that of the senior management – in order to institute and enforce cross-departmental projects.

Process improvements must also be reviewed across enterprises. For companies currently operating in close tandem with trading partners, the inefficiencies in business processes typically lay at the enterprise borders between the companies. In today's close-knit business world, when the major challenge of mapping the entire process between partners and creating optimized end-to-end business processes is solved, it generates tremendous competitive advantage.

Process-Driven Enterprise

Cross-functional business processes require a new management structure and mindset. Consider a typical order-to-cash process that touches sales, distribution, and accounting departments. The VP of Sales, CFO, and VP of Operations each has functional control of his or her individual business fiefdom. However, ensuring that end-to-end processes run smoothly requires one person – a process owner – to be accountable at the process level.

As the enterprise migrates toward BPM, one of the senior managers cited above may control the process, while all the functional managers remain intact. During this interim period, the organization would operate in a horizontal and vertical hybrid mode. Long term, it would probably replace its functional managerial structure with a process-centric model and operate horizontally, to better align itself with its business processes.

Process Feedback Loop

As business processes, automation increases, business ambiguity decreases. To automate processes, they are necessarily defined and documented. Business processes become better managed because senior management has aligned the processes with strategy.

These elements combine to create improved access to information. The end-to-end processes are contained in the BPM system and, thus, measurable. The forecasted results are already known, making comparisons straightforward and meaningful. Enterprises can benefit by implementing strategy rapidly and reacting to change faster, with increased accuracy.

Conclusion

Tactically, the BPM system enables superior application development and a set of full life-cycle support services. Companies that embrace BPM strategically create both clearly defined processes architected to leverage today's technology as well as the management structure to properly oversee the newly designed processes. The graphical tools used by both the business users and the IT professionals to model business processes achieve the seemingly impossible – unifying IT and business. All of this creates an enterprise that runs with a clear vision and the ability to predict and proactively act on business change, helping the process-managed enterprise stake out leadership in its markets.

About the Author

Neal Novotny is in charge of product marketing for Intalio. For over 15 years he has held product marketing and product management positions in the software industry, including stints at Vitria and BEA Systems.

nnovotny@intalio.com

BPEL Unleashed

Putting a modern business process execution standard to work

BPEL (Business Process Execution Language) makes business processes and composite Web services first-class citizens of the Java and .NET platforms, while preventing vendor lock-in. The result is a drastic reduction in the complexity, delivery time, and cost associated with implementing workflow, BPM (business process management), and related business integration projects.

WRITTEN BY
DORON SHERMAN

BPEL is a new standard for implementing business processes in an emerging service-oriented architecture world. As such, applying BPEL introduces new considerations, challenges, and pitfalls for delivering process-aware applications based on a service-oriented architecture (SOA).

The Rise of BPEL

There has been a continuous need in the enterprise to integrate systems and applications into end-to-end business processes. Traditional integration solutions are arcane, proprietary, and expensive, and have failed to be widely adopted across the enterprise. This has driven the industry to a new business process execution standard called BPEL. A widespread group of technology companies have committed to supporting BPEL, including all the Java and .NET platform vendors, and even at this early stage, portability across vendor implementations is being demonstrated.

In less than two years since being unveiled, BPEL has become the de-facto orchestration language standard, bypassing a number of alternative specifications such as BPML and WSCI. As Web services march towards mainstream adoption, focus is now shifting to delivery of process-centric applications based on a service-oriented architecture. To a large extent, the creation of BPEL was driven by the need to facilitate flexibility, visibility, and ease of management at the process layer. Applying BPEL effectively separates process logic from the rest of the application.

Separating process logic from the rest of the application and making it explicit is similar in pattern to the earlier separation of data representation from application logic. BPEL is a standardized, semantically rich, process language based on the Web services stack. The XML-based nature of BPEL processes enables the editing of process logic using visual design and modeling tools. It also allows for progress tracking, aggregate reporting, and complete end-to-end management of deployed processes. Access to runtime process execution information can in turn be used to analyze and further optimize BPEL processes.

With BPEL promoting interoperability at the tools level, over time a more collaborative development process is possible, allowing for developers, business analysts, and other skilled professionals to engage where they fit best at various stages of design, development, deployment, and management. The gap between business analysts and software developers has been a well-known impediment to efficient and successful implementations of process-aware applications. BPEL as a standardized process orchestration language carries the promise of narrowing that gap and allowing for effective collaboration between these otherwise disjointed groups of professionals.

The Scope of BPEL

Although relatively new, BPEL essentially derives its maturity from its predecessor orchestration languages, XLANG and WSFL, representing a vast implementation experience through research and commercial deployment. That background largely mitigates the risk generally associated with betting on a new standard and is augmented by the support of 100+ members backing the progress of the BPEL specification through the WSBPEL technical committee at OASIS.

With a design goal of maintaining a clean specification, BPEL focuses on supporting composition and coordination of services into business processes and new compound

–Continued on page 34

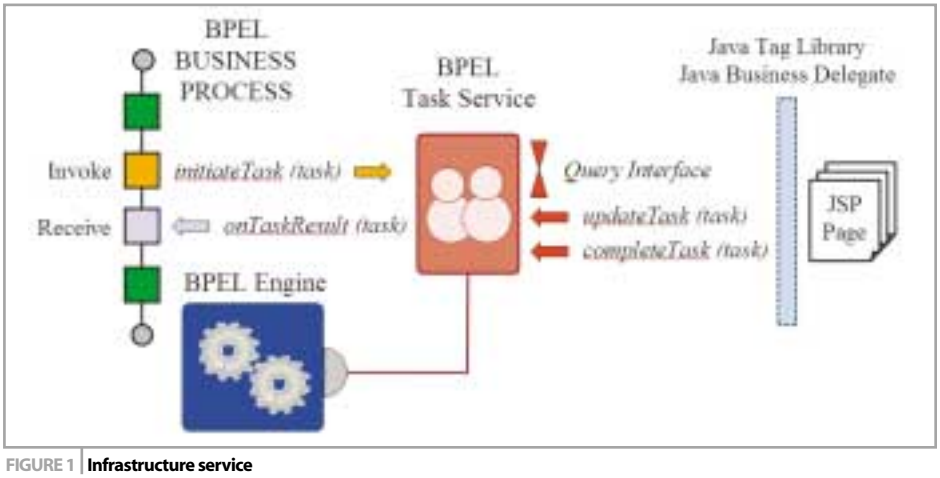


FIGURE 1 | **Infrastructure service**

Everything OLD Is NEW Again!

Imagine **XML-enabling** your existing **PowerBuilder** and **Visual Basic** applications

QUICK
CHEAP
SAFE

In hours, without changing a line of code

Inexpensively, using your current skills

Safely, preserving all logic and security

NOW YOU CAN

www.active-endpoints.com

Closing the Deal with BPM

Move your client off the fence and into commitment

Whether you're in a clothing store, a car dealership, or a bank's online catalog looking for low-interest loans, you're more liable to buy if you get some personal attention. You want someone to answer your questions in real time.

WRITTEN BY
DAVID CAMERON

The trick for companies doing business online is to provide customers with the same feeling of personal attention that they would get on the shop floor. But since there is no highly experienced sales person sitting next to you when you go online to buy a DVD player, companies need to create the illusion of a seasoned salesperson by setting up systems to respond immediately to your questions and concerns. Any delays or gaps in information likely result in an abandoned shopping cart.

Customer relationship management (CRM) and hard-wired enterprise application integration (EAI) tools were initial steps to solving the problem. With great effort, they connected some customer-facing systems in an attempt to create this virtual salesperson. The result was more like an inexperienced salesperson who might be able to tell you, in time, how much different DVD players cost, but not what you get when you pay for the more expensive model. Delivering the kind of "personalized" experience that customers expect from either a real or virtual salesperson requires seamlessly integrating internal systems.

Seamless integration requires XML in order to create a lingua franca between previously siloed business systems. Suddenly, common objects for "Trade" and "Address" mean the same thing in the CRM, enterprise resource planning (ERP), and other systems. But generating such a common language in XML is only the first step. Companies need to establish the framework for using that common language to create processes that deliver the information to the right person at the right time. A business process management (BPM) approach can help companies integrate core business systems at the process level, based on an XML framework for correlating data.

Such an integrated system allows product managers to change strategies on the fly to address shifting market tastes. It ensures that sales managers can constantly tweak sales strategies to handle new targeted marketing programs. It also gives service representatives instant access to customer information to deliver the best care possible. Moreover, it gives companies the tools to automatically generate business processes that increase an organization's chance of closing the sale.

Smart Customer Handoffs

Despite the desire to automate many processes, many call centers still field calls from customers who could not complete online processes. Unfortunately, the customer often has to start over with the call center representative, wasting valuable call center time and creating the perception of bad customer service. Managing the "handoff" from the online to the offline experience better can provide significant cost benefits by reducing call center time.

Making this handoff smarter requires integration. However, integration's well-known cost and complexity have scared many organizations into thinking that the difficulty of making the online and offline worlds function seamlessly is greater than the perceived benefits. New approaches to integration and technology, however, have changed the balance.

Traditionally, CIOs have had two ways to support the kind of requirements described above. First, they could write a lot of custom program code. However, code is extremely hard to change once written. That means that someone would have to query users to determine what they want the system to do prior to writing code. The cost for this type of approach is significant, which means that most CEOs will demand a detailed ROI with a 12-month payback schedule. Most of the cost of custom coding is due to two factors:

- Initial code must come very close to requirements, and requirements must be very specific, since it is expensive to go back and change after the fact.
- IT resources must be made available on an ongoing basis to troubleshoot, modify, and upgrade the code.

The second approach is that of data integration. By creating a common database, users and applications can share information and coordinate activity. This, too, is very expensive because it requires complex architecture efforts, as well as ongoing maintenance that involves refreshing data. Data integration's cost stems from multiple factors:

- Much of the cost is infrastructure to move and store data for potential (not known actual use) future use by application. That means most of the data is there for insurance purposes, and probably 20% of the data is accessed 80% of the time.
- Modifying business processes usually requires architectural rework, which drives up maintenance costs.
- The level of granularity of the data (i.e., do we need transaction-level data or just summarized data of monthly transactions?), and the latency of data (how often do I need to refresh it?) drive key cost decisions – decisions typically made data element by data element.
- The cost of having IT staff hide the complexity of the physical data's appearance from the end user who can't understand table layouts, etc., also plays a role.

It is not surprising, then, that when faced with requirements to be more agile (i.e., view change as necessary and good instead of costly and undesirable), the impracticality of these approaches rises to the surface.

Many end users, analysts, and vendors have reached the same conclusion – for the types of requirements emerging today in support of business imperatives, a new approach is necessary.

Integration and Automation with BPM

This new approach is business process management (BPM), and it relies heavily on new standards emerging in the form of XML and Web services to overcome the difficulties in prior generations of integration technology. The advent of these components enables a higher degree of automation and reusability than ever before.

BPM coordinates the actions of isolated IT systems – both online and offline. Using the built-in intelligence of existing applications and systems, BPM can help companies nimbly react to key business events and capture millions of dollars in potentially missed revenue opportunities while improving customer service.

BPM technology overcomes the weaknesses of the current generation of systems integration approaches.

It provides a *variable cost model* that allows for a small initial implementation footprint tied to a three- to six-month payback period, tightly coupling investment and return. Today, technology must prove utility before vendors demand license and support fees. Instead of the typical 12-plus month implementation and two-year payback model, BPM technology scales deployment efforts to initial requirements. As a company extends its BPM implementation over time, it builds upon work already performed.

BPM allows for rapid modification so end users can engage in "test-and-learn" process development and management. Most of the processes that support key business drivers are complex and must continuously evolve in response to internal and external changes. BPM technology enables users to implement, test, modify, and re-implement processes in rapid sequence based on what they've learned.

BPM gives end users more involvement, so they can change processes on the fly and improve agility and productivity without having to call IT. The current generation of technologies is brittle largely because IT shoulders the bulk of maintenance costs and responsibilities. BPM technology balances the workload between the end user and IT in such a way that users can maintain processes without resorting to IT support.

Companies benefit from BPM's ability to reuse common data objects such as "Customer" and "Trade" and use open standards to make them available to any application. The complex infrastructure many organizations operate has evolved from proprietary database structures and application logic syntaxes of years of systems development. Most valuable components of business processes, including data definitions, business rules, and transformation logic, are replicated in a variety of formats across the enterprise. BPM technology relies on an object model that exposes and abstracts these elements so that they may be reused across different systems via emerging XML standards.

> "Today, technology must prove utility before vendors demand license and support fees"

BPM provides context for business process to allow users to access information at the application level, before it has been saved to a database, dramatically reducing data integration. Most traditional application integration is based upon the movement of "state data,"or data that has been saved about a particular event. State data is stored in databases, and then "synchronized" with other databases linked to other applications. But companies don't save much information needed to support business rules because it is too expensive or too complex.

Companies can use BPM to cultivate existing business logic and integration capabilities to connect existing applications and databases without modifying them. Most of these systems already have connections built into them, either with middleware or published application programming interfaces (APIs). BPM technology uses these connections to link databases and applications while providing a layer of abstraction that hides variability between systems.

Finally, BPM automates many integration tasks that previously required additional code or manual configuration. For example, if a business process reaches a decision point and needs to know whether a given customer is "high" or "medium" value, programmers typically need to manually code the "fetching" of that data. Today's BPM technology can automatically provision information "just in time" based on instructions provided by the user at the metadata layer. This allows the BPM server to automatically get that information without a user having to specifically say "time to go get customer lifetime value." This feature is especially powerful in dynamic environments where processes may change, resulting in much manual rework of data fetching steps without it.

Gartner estimates that 75–80% of all integration projects are done with custom code. Despite integration technology's nearly decade-long evolution, it is still not a practical solution for most companies. As businesses evolve, brittleness must give way to agility in how systems support emerging business requirements. By taking a fresh look at integration from the process perspective, BPM technology has the potential to dramatically reduce the cost and effort involved with integration, generating tremendous benefits to both businesses and their customers.

Ensuring quality customer experiences online or in person means delivering the right data at the right time to the right person, whether it's a salesperson, a service representative, or the customer via self-service online. Using XML as its foundation, BPM provides the smooth integration of different systems to ensure that necessary data is available instantaneously to those who need it. It gives companies the tools to start converting undecided customers into those who are willing to commit, and thus bringing more revenue into the company.

About the Author

David Cameron is vice president of marketing and product integration at AptSoft Corporation (www.aptsoft.com).

david.cameron@aptsoft.com

If Every Company Is Adopting Web Services...

...Why are so many failing?

Like most companies, you want to streamline your business, automate manual processes, and expose key functionality to partners and customers. And like most companies, you probably have a mixture of applications and hardware – some new, some old.

WRITTEN BY
NANCY ESPOSITO

Your customer information may reside in a database, but you rely heavily on an AS400 green-screen system for critical business functionality. Other important business functions may be accessible via COM APIs or even as Web services. Without question, integrating these disparate applications and data sources is among your biggest challenges.

There certainly is a lot of hype saying that Web services will provide a cheap and easy solution to this type of integration challenge. Yet Gartner estimates that $1 billion will be wasted on misguided Web service projects by 2007. How much of this will be yours?

Striving for a Business Service-Oriented Architecture

IT departments are talking about SOAs (service-oriented architectures), CBD (component-based development), composite applications, and ESBs (enterprise service buses), but what do they really want? They want a standardized mechanism for exposing business processes so they can quickly and efficiently roll out functionality that supports key company goals. Using Web services is a logical choice.

To do this effectively the Web services must have a business focus and the ability to coordinate the integration of multiple applications and data sources. For example, insurance companies that want to expose a Web service for renewing an auto policy will need to orchestrate interaction between several different applications, including some that may lie outside the firewall. We'll call the environment that supports this *business service-oriented architecture* (BSOA; see Figure 1).

Creating a BSOA is more of a business-centric exercise than the SOA approach typified by object-oriented practices of encapsulating CICS transactions, J2EE components, SQL queries, and other functionality with Web services or another common interface (see Figure 2). Today's service-oriented architectures lack true business focus because few organizations have managed to expose Web services that can interact with multiple applications in real time. A typical Web service today tends to be data-centric and generally interacts with a single back end. For this reason you are far more likely to see a Web service that invokes an SQL query to retrieve customer information than a Web service that renews an auto policy.

What are the key characteristics of a business-centric service? First, it requires the ability to combine method calls, SQL queries, legacy system functions, and other individual components. To tie these pieces together the service needs business logic – for example, to govern the order of operations and specify the data elements needed from each system or component to fulfill the service at runtime.

A business service for renewing an auto policy might involve the steps and systems shown in Table 1.

Step	System	Business Functionality
1	Policy Administration	Fetch policy details
2	Claims	Check for outstanding claims against policy
3	Billing	Verify that premiums are up to date
4	Rating Engine	Generate renewal quote

TABLE 1 **Steps to a business service**

June 28–
July 1, 2004

Moscone Center
San Francisco, CA

Join James Gosling, the father of the Java Programming Language

Java™

Everywhere starts here

Java™ technology is everywhere, improving the digital experience for everyone. It all starts at the JavaOne℠ conference, your source for cutting-edge knowledge and proven solutions. Discover from the experts how to deploy Web services and connect the world securely; you'll learn to code simpler and faster, and bring higher efficiency and profitability to your business.

The JavaOne conference offers hundreds of in-depth technical sessions in:

- Topic 1—The Foundations: Core J2SE™ Technologies
- Topic 2—Core Enterprise Technologies
- Topic 3—Java™ Technology on the Desktop
- Topic 4—Java Technology for the Web
- Topic 5—Java Technology for Mobility
- Topic 6—Dissecting the Implementation: Solutions
- Topic 7—Intriguing and Unexpected: "New and Cool"

JavaOne℠

Sun's 2004 Worldwide Java Developer Conference℠

Save $200! Register by May 31, 2004, and receive the Early-Bird price for the full Conference package. Registration code: ADBEZKND

Register at
java.sun.com/javaone/sf

Sponsored by

Sun microsystems

Produced by

MediaLive™ INTERNATIONAL

At a lower level, the service would require additional logic to dictate what should happen when claims are outstanding, if the policy has lapsed, and so on. A business service that can traverse different paths based on these conditions provides functionality that, if exposed as a Web service, can offer significant value. Similarly useful Web services can be envisioned for providing access to a customer's portfolio, booking a vacation online, or checking inventory levels and placing an order based on the results.

If a company makes all back-end systems conform to the same interface, data-centric Web services may seem valuable because with a uniform interface it becomes considerably easier to write the integration code. But starting a project by wrapping all back-end systems as Web services causes a tendency to over-integrate; there's no intrinsic value in wrapping every component as a Web service. Companies that follow this route will end up with a giant Web services component library. (This is amazingly similar to the architectures that enterprises have spent millions building for years using standardized component models such as CORBA, COM, and EJB.) Before harvesting the functionality from their new library, organizations will need to spend significant time and money writing code in an application server to get the Web services to interact in a meaningful fashion.

This bottom-up approach to integration has never lent itself to rapid ROI and is a far cry from the cheap, easy integration promised by ardent Web services proponents. It is also the primary reason that so many Web service projects are failing – and a chief contributor to the alarming $1 billion figure cited by Gartner. Other trouble spots include lack of industry agreement on a standard implementation, performance issues, security concerns, fault tolerance, and difficulty guaranteeing service-level agreements. This combination of factors makes it easy to understand why the business community is left wondering why the IT people are unable to deliver on the promise of rapid ROI.

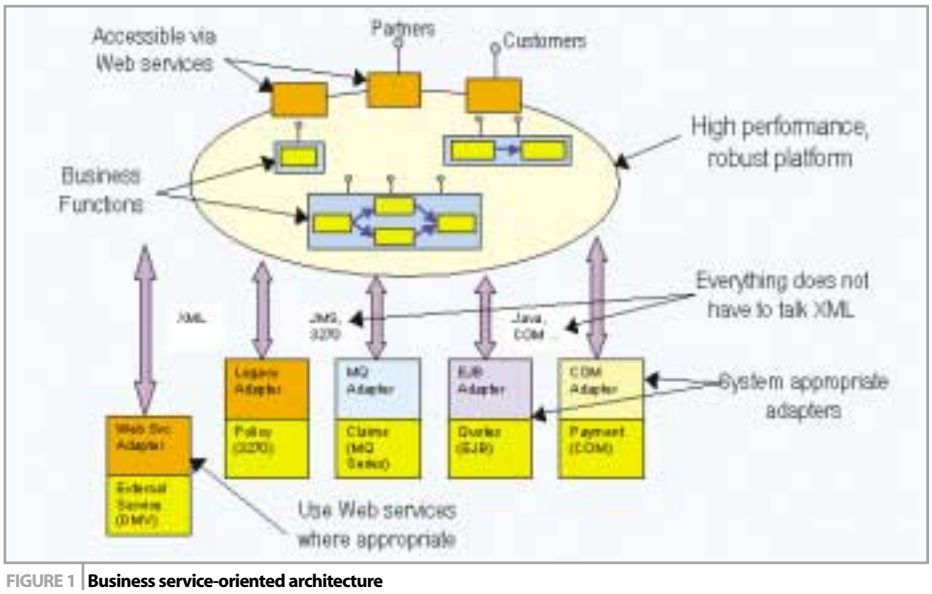


FIGURE 1 **Business service-oriented architecture**

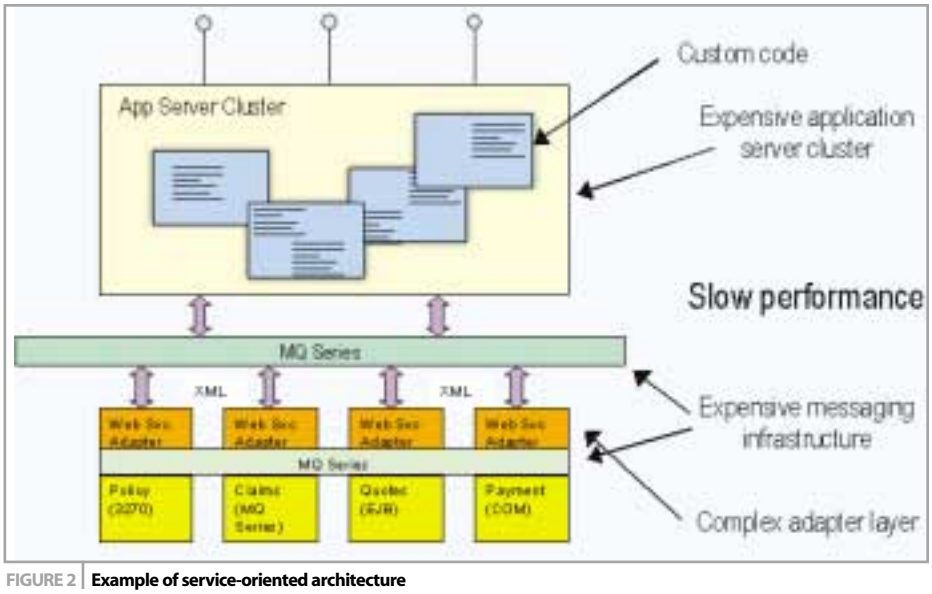


FIGURE 2 **Example of service-oriented architecture**

Sharpening Focus with BPI

The primary issue is not whether to expose key functionality as Web services, but how to do so selectively and cost effectively. Web services are an excellent paradigm for exposing business-centric services. They are gaining universal support and have the advantage of being vendor and platform neutral, and issues such as security and performance are gradually being worked out.

To take advantage of Web services without venturing down the componentization road of the past, a number of organizations are introducing business process integration (BPI) software into their technology mix. BPI's unique capabilities allow organizations to expose as Web services only the functionality that is actually needed by customers and business partners.

A relative newcomer to the industry, BPI offers a flexible and speedy approach to integration because it does not require sophisticated infrastructure, custom code, or a standard adapter layer. Its most distinguishable feature is a visual modeling environment that allows you to design business processes without programming expertise. This is made possible via point-and-click connectivity to back-end systems, data sources, SQL queries, CICS transactions, Web services, and so on. These connectors eliminate the need for a standard adapter layer and give you the wherewithal to tackle integration projects without wrapping every component with a standard interface.

After you design your graphical process models, the BPI software converts these models to executable programs that at runtime can access multiple applications and perform the business logic you have specified. With a robust execution engine that

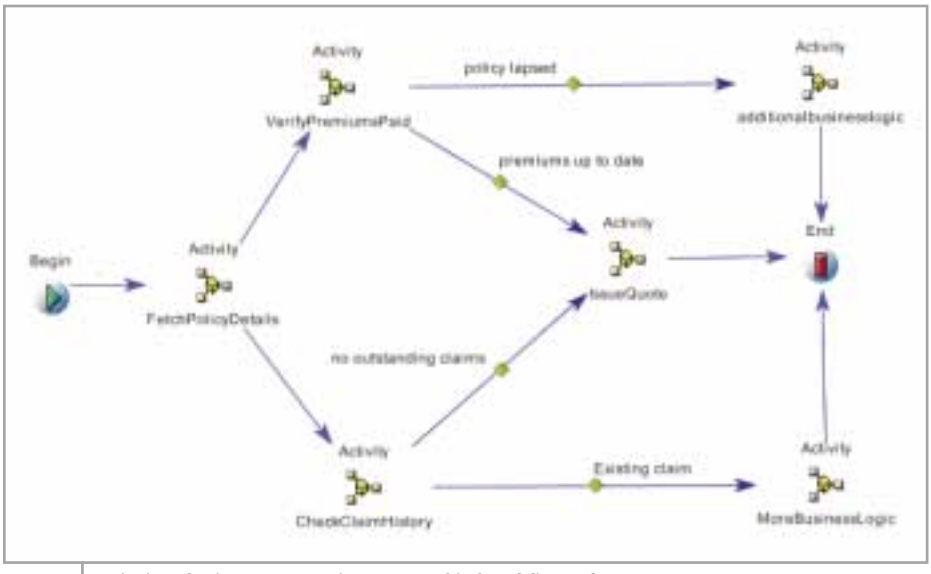


FIGURE 3 **Designing a business process using a BPI graphical modeling tool**

handles fault tolerance, load balancing, scalability, and security, you are free to expose functionality without additional messaging infrastructure or complicated code that must run under an app server.

BPI does not offer some highly specialized capabilities found in full-blown, multimillion-dollar EAI solutions (such as dealing with enormous volumes of data), but it is an affordable, all-purpose tool that greatly accelerates the pace of integration projects and the ability to deliver rapid ROI. You can swiftly assemble new business processes by piecing together fully functional components and using the graphical modeling environment (see Figure 3) to fill in the blanks with business rules. Not only do you avoid modifying perfectly good legacy applications, you can focus squarely on the functionality required to complete a business transaction and ignore whatever does not contribute directly to the process.

Finally, and perhaps most relevant to companies aggressively implementing Web services, BPI software enables you to expose the same business process using a variety of technologies. Even if your company is ahead of the Web services curve, some segment of your customer or partner base may not be ready to use your Web services. With BPI you can expose processes by using a JSP, ASP, or servlet; or even via a fat client such as a Java application or an Excel spreadsheet. What matters is providing Web access to your business functions; typically, customers and partners don't care about the underlying technology.

A Business Example

Let's examine the business-centric approach to building an SOA via our earlier example of renewing an auto insurance policy. Using the BPI approach you begin by analyzing the existing (often manual) business process for policy renewal and determining the systems involved. You then design the process using the graphical modeling tool; your model will include a sequence of activities outlining the steps each system will perform, the data that must flow between the various applications, and business rules that handle logic not performed by any one back-end system.

Note that this approach keeps the focus almost exclusively on business functionality. For example, the first step is retrieving the policy details, which means you are dealing with only a tiny slice of the sweeping functionality provided by your policy administration system. The same is true for your claim system: you are interested only in a single policy's claim history; all other functionality provided by this system is irrelevant and does not need to be componentized.

Next, using point-and-click wizards you connect the first step in your graphical model to the function in the policy administration system that fetches policy information and continue this exercise until each step in the model is bound to the corresponding system or data source. To ensure high performance, BPI uses the most appropriate technologies for connecting to back-end systems, so for example if the claims system already has an MQSeries interface, the adapter will use a message queuing technology. By contrast, the data-centric approach would first require you to wrap the claims system in a Web service. In addition to the extra work, this often has a negative effect on performance by making the connection to the claim system slower than using its native MQSeries interface.

When you are ready to test your automated processes, you deploy the visual models and connectors to the engine's repository. No additional middleware technology, app servers, or custom code are required; the processes are ready to execute immediately.

After this you are ready to expose your processes to customers and partners – as Web services for those who can embrace Web service technology; and for those who cannot, perhaps directly from an agency management system or as a JSP. You can also take advantage of BPI security features to allow access at the business process level to authorized users, and thus avoid exposing internal functionality to the outside world.

Finally, you can take advantage of BPI's monitoring capabilities to analyze the execution of your processes and make any adjustments necessary to fine-tune performance.

Delivering ROI

Web services continue to garner universal vendor support and are clearly here to stay, but this does not mean they can magically solve every integration challenge. The organizations that will experience the most success with Web services are those that use them strategically, where they make sense in the context of the business.

IT departments that resist the urge to build an expensive component library and take full advantage of complementary technology such as BPI will find that it is possible after all to improve service and deliver on their promises of rapid integration – and rapid ROI – to the business community.

About the Author

Nancy Esposito is director of software engineering at Metaserver, Inc., a leading provider of business process integration solutions for the insurance industry.

nesposito@metaserver.com

Wisiba Web Service Management from Itellix

A unique product that deserves consideration

There's a phenomenon I've witnessed again and again in my years building systems. I call it "Guerilla IT." Listen to my description and see if you've witnessed it inside your organization.

WRITTEN BY
PAUL T. MAURER

You are an IT professional serving your constituency. They are satisfied with your support of their business, but there are a few projects that are lower priority and small enough to fly under your radar. They implement a little "solution" themselves to help get their work done. They are proud of the quick, simple, and cost-effective solution and innocently wonder, "Why can't they get things done this fast?" It works well for them for a little while, but then as the business grows or changes or both, they find they're spending too much time maintaining and enhancing their "little solution". Now they want to hand it over to you to enhance and support.

There are probably a hundred variations of this story, the most relevant of which is grassroots growth of Web services implementations for use inside and outside the enterprise. It is so easy and tempting to expose capabilities as Web services these days that they can pop up almost anywhere. But then as demand grows the support requirements outstrip the original creator's ability to cope and they hit the "scramble threshold." That is, they scramble for someone else to support and maintain their Web services.

Enter the Adults

When enough guerilla projects hit the scramble threshold, the need for real enterprise Web services management becomes abundantly clear. That's when products like Wisiba from Itellix enter the picture.

Wisiba is a new entry in the Web services management marketplace. Wisiba comes in three configurations. The Base configuration provides monitoring, auditing, security, versioning, and provisioning. The High Availability configuration layers on top of the Base and adds SLA management, QoS assurance, load balancing and dynamic routing, failover, contracts, and revenue management. Finally the High Scalability Configuration includes the PowerPlus native proxy server.

The Basics

Wisiba's Base Configuration provides all the standard Web services management capability that you would expect. It acts as a virtual service endpoint that performs the basic services listed above and then routes the request to the physical service for execution. Wisiba can run coresident with the Web service container or run separately as a proxy.

Itellix provides two graphical user interfaces for Wisiba. The first is the Activity Monitor (see Figure 1), which allows the operator to track Web service operations. It is surprisingly functional for such a new product, and I especially like the ability to chart various Web service metrics. The second GUI is the Wisiba management console, which allows for setup and provisioning of Web services. The management console is also core to configuration of service-level agreements, which I discuss in more detail later.

High Availability

Wisiba becomes very interesting when you add the High Availability configuration. Yes, it adds the obligatory load balancing and dynamic routing but with an eye towards the economics of the situation.

jWisiba's logical management model (see Figure 2) starts with the "Service Provider." Service providers list available services and their versions. Service versions are associated with bindings that are a reference to the actual service implementations. A service offering associates a particular service with a payment model, any obligations, and usage. Payments supported are pay as you go (ad hoc), limited time, or limited amount of transactions. Each payment model is customizable. Finally, a provider contract associates a set of service offerings to a par-

itellix

Company Info

Itellix Software Solutions
101 Prestige Poseidon
139 Residency Road
Bangalore, INDIA-560025
Phone:+91-80-51125501
Fax:+91-80-51125504
Web: www.itellix.com
E-mail: madhukar.srivastava@itellix.com

Licensing Information (2 CPU and under)
Base Configuration: $25,000
High Availability Configuration: $15,000 (add)
High Scalability Configuration: $30,000 (add)

Testing Environment
OS: Windows XP Professional (Service Pack 1)
Hardware: Intel Pentium M Processor (1300MHz) – 1.29 GHz – 1 GB RAM

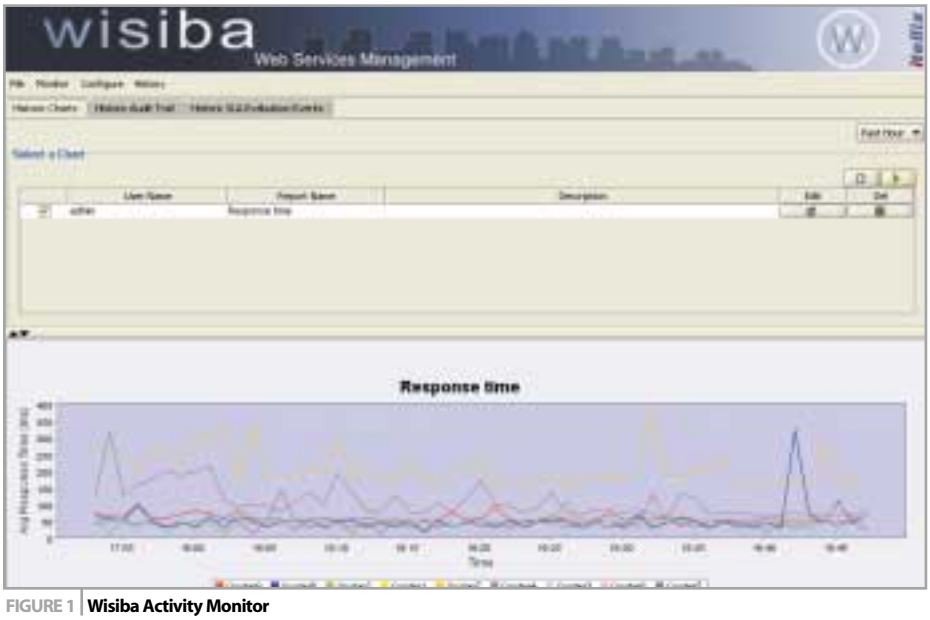


FIGURE 1 | **Wisiba Activity Monitor**

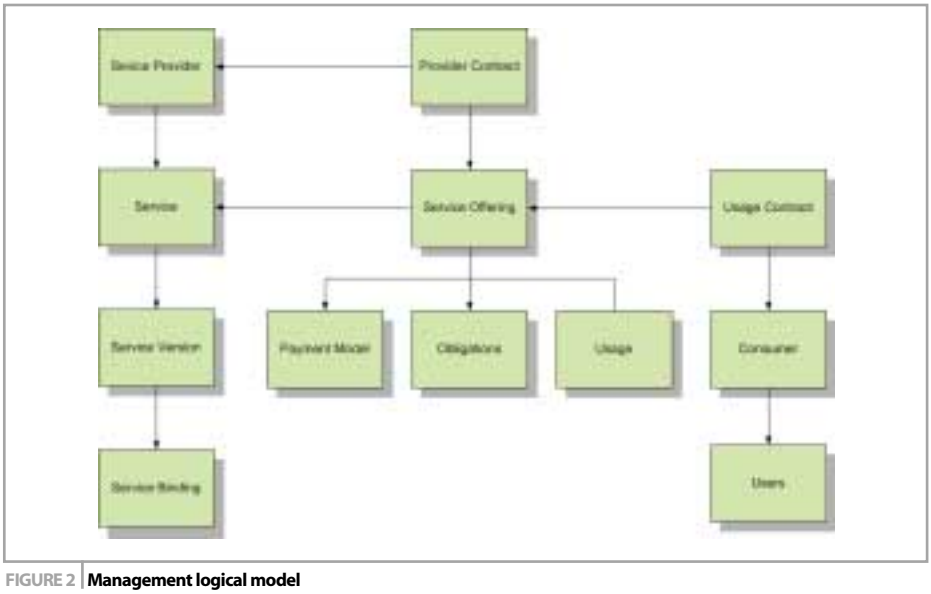


FIGURE 2 | **Management logical model**

ticular service provider.

From a service consumer standpoint, a consumer may be a business entity with multiple users. A usage contract contains details such as the service, version, and Quality of Service level contracted. The usage contract also contains the term of the contract and the payment model chosen.

Wisiba tracks all usage towards the contract and this data can be fed into the corporate billing systems. Additionally, Itellix has tied all the QoS and service-level agreement data back to its load-balancing and monitoring systems. For example, if a consumer has exceeded their transaction limit, the monitoring system can notify the consumer, producer, and operator. If the service falls below an established QoS agreement, notifications can also occur. Wisiba can easily spray Web service requests across a bank of machines, but a more realistic load-balancing mechanism would be based on SLAs; Wisiba's unique architecture allows this.

Scalability

Wisiba's High Scalability configuration simply layers the PowerPlus native proxy on top of the other configurations. My first thought when adding an additional box to the request path is "how much is this going to slow my response time?" The unique PowerPlus proxy architecture could alleviate this concern.

Itellix has built the proxy using a Staged Event Architecture first described in a Ph.D. thesis by Matthew David Welsh. This architecture requires a system to be built as a network of stages with requests processed using a combination of thread- and event-based programming models. The architecture also relies on feedback-driven control to allow it to adapt to overload conditions. The result is a system that scales well and is resilient to large variations in load. ***WSJ*** is not set up for load testing but Itellix claims they have achieved over 1 million transactions per hour on a commodity dual CPU system.

This architecture also allows Wisiba to profile requests and enqueue them based on predefined SLAs. Top-tier customers can be placed on the highest priority queue where they are served quickly.

Testing

I received the evaluation copy of Wisiba directly from Itellix. The install process was a little bumpy. Make sure to read through the install guide completely before starting as Wisiba will prompt for a number of things as the install proceeds. Don't assume the install defaults will just work. You'll need to provide your own database; the evaluation version requires MS SQL Server. Make sure you create a database and user ID specifically for Wisiba before you start the installation process.

There is a fair bit of setup configuration but it is important and definitely worth it when running in a production environment. Once I installed the product, testing went smoothly. After setup, I was able to provision and access several versions of my services quickly and monitor their progress.

Conclusion

Wisiba from Itellix sits well in the Web Services Management product space. The product's unique handing of Quality of Service and service-level agreements make it a product that should be considered. So, if your organization hits the "Scramble Threshold", contact Itellix and ask for more information about Wisiba.

About the Author

Paul T. Maurer is a principal in the financial services practice of a leading consulting services company.

paul@paulmaurer.net

ESB Technology & Innovation

Extending Web services with asynchronous message delivery and intelligent routing

Called to larger tasks, messaging technology is now in a phase of evolution. A mixed message model is needed to combine the best of Web services and traditional asynchronous message delivery to provide the flexibility required for today's real-time enterprise.

WRITTEN BY
ROBERT DAVIES &

NIGEL THOMAS

Traditional message-queuing middleware will soon be replaced by enterprise service bus (ESB) technology – taking messaging to the next level. The new ESB backbone, which will enable the next generation of integration and application platform products, will bring radical improvements to the software infrastructure of most enterprises. The industry transition to messaging and ESB as the core application platform infrastructure model will mark an inflection point – triggering a massive new wave of innovation around businesses' use of their information resources; capitalizing on the architecture of events. This will dispel any recently raised doubts about the critical role that IT can play in strategic business differentiation.

Introduction

Over the last decade, competitive pressures have combined with technology to fundamentally change the rhythm of business. In the past businesses could make decisions based on month-end batch reporting. Now, just-in-time processes mean that this morning's raw materials glitch or power cut becomes this afternoon's inability to deliver and invoice finished goods. The business has to react more and more quickly to unexpected events – otherwise, it goes to the wall. The "zero latency enterprise" has finally arrived.

Today's enterprise environments have grown piecemeal to meet this challenge. "Islands of computing" – isolated functional or departmental stovepipes of application and data – are supported by heterogeneous storage, networks, and hardware, making utilization and management overly complex and resource-intensive. This complexity is an obstruction to the most pivotal challenges that an enterprise must face:

- To meet continuously growing demands for rich information services delivered through a variety of channels
- To manage infrastructure to meet changing business requirements in real time.
- To diversify business offerings to flexibly grow the business and mitigate economic risks associated with stagnant product lines
- To guarantee customers, partners, and employees quick and high-quality responses to requests for information services

The rapid evolution of EAI, B2B, and application development segments over the last few years has led to the development of several key technologies and standards that have driven dramatic advances in the infrastructure space:

- XML has arrived as the universal, self-describing data exchange format for the majority of applications. Web-oriented information exchange and the infrastructure behind it, together with XML, have made the use of Web services inevitable.
- Java has been accepted as a major technology to use on the server side, and J2EE has been accepted as the standard for application servers.
- The use of an enterprise message bus for both transactional message exchange and real-time event notification has become standardized around the Java Message Service (JMS).
- A common management framework for

Lowering the cost and complexity of developing secure, manageable enterprise-class Web services with BEA WebLogic Workshop™ and Confluent for BEA

With Confluent for BEA, Web services developed in BEA WebLogic Workshop™ 8.1 are automatically monitored, managed and secured at development time. Confluent's Control is bundled with WebLogic Workshop 8.1.

Real-time monitoring
No additional coding required, turn on instrumentation for a Web service, view all conversations as they execute, and drill down to views of individual operations and controls

Policy specification and enforcement
Define operational policies including security, logging and monitoring for Web services within BEA WebLogic Workshop 8.1

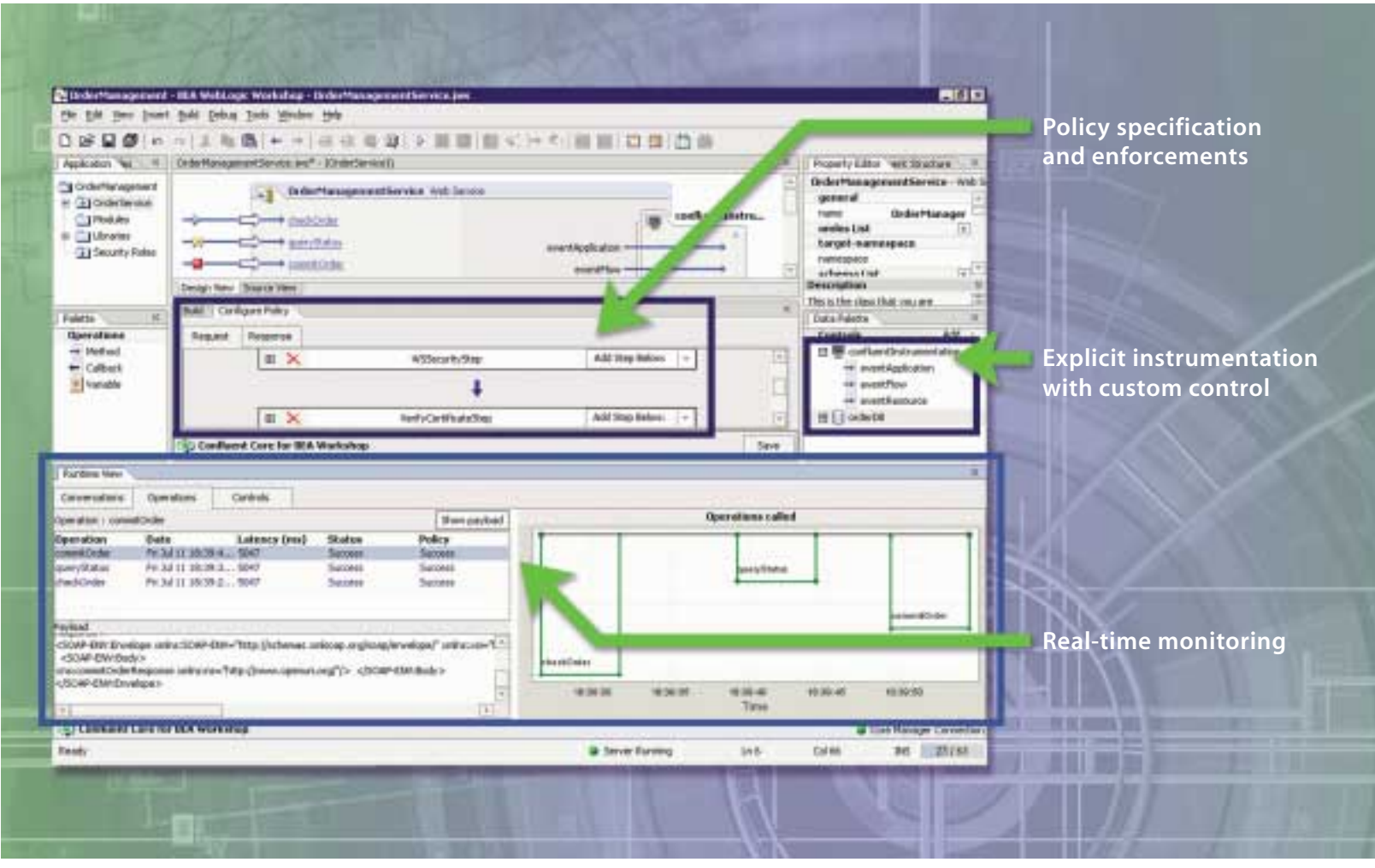
Explicit instrumentation with custom controls
Use Confluent Control to explicitly call operations such as monitoring and security from Web services

BENEFITS:

If you're a Developer—focus on application logic not infrastructure services

If you're an Architect—consistently implement security, change and other operational policies

If you're an Operations Manager—efficiently monitor, manage and evolve increasingly distributed applications



Confluent Software

Download Confluent for BEA at:
http://www.confluentsoftware.com/solutions/download/beacib.html

For more information on additional Confluent Web services management solutions, visit: **http://www.confluentsoftware.com**

server-side components has been realized through Java Management Extension (JMX) standards.

Infrastructures Must Behave the Way a Business Behaves

The ever-changing market requires the delivery of rich information services through diverse channels. The next-generation enterprise calls for loosely federated resources sharing a common communication and management infrastructure across multiple domains. Enterprise infrastructure has to behave the way a bricks-and-mortar business behaves, allowing the dynamic management of resources to meet fluctuating demand from customers and partners at the same time as dealing with changes in the supply and availability of system resources. Enterprise applications also require a standards-based collaboration model to maximize the utilization of this infrastructure. To achieve this, the real-time enterprise uses best practices from real-time infrastructure and server-side grid technologies.

The Components of the Real-Time Enterprise

The real-time enterprise is founded on some of the same concepts used to define server-side grid environments, and a structure similar to Gartner's five-layer grid technology model can be used to describe its core components (see Table 1).

An ESB builds on existing widely adopted technologies and open standards to provide an adaptable distributed architecture for services collaboration, management, and control. The ESB supports runtime deployment of business services anywhere in the enterprise and provides services for collaboration and notification as part of its core infrastructure. Let's see how ESB technology maps onto the five layers of the model.

Infrastructure Resources and Virtual Operating System

Layer 0 consists of the infrastructure resources, including networks, servers, storage, and operating system environments for each server. Layer 1 sits above the infrastructure layer and establishes a distributed operating system across multiple resources that support functions such as work scheduling, integrating resource names into a global structure, and ensuring consistent authentication across disparate systems.

Although Gartner places J2EE as a Layer 2 technology, we believe the combination of distributed JMX and a J2EE-based application server has the characteristics of a virtual operating system. Using a container or micro-kernel that provides hot deployment and full JMX management of all components and services allows remote activation and management of services.

JMX as a technology was originally designed to manage individual agents, such as an application server. By combining JMX with JMS, its scope can be extended to manage individual agents, clusters, or loosely coupled federations (super-clusters, if you like), allowing complete life-cycle and deployment management of a federated ESB infrastructure. Because JMX also integrates with more traditional management protocols such as SNMP, the ESB infrastructure can provide on-demand hot deployment and self-annealing infrastructure across Java, Web services, and legacy platforms.

Layer 4:	Administrative Support	Manual and automated management
Layer 3:	Applications	Distributed processes and services
Layer 2:	Distributed Programming Model	Service and distributed process model
Layer 1:	Virtual Operating System	Distributed operating system
Layer 0:	Infrastructure Resources	Network, operating systems, and storage

TABLE 1 | **Real-time enterprise model**

Distributed Programming Model

A distributed programming model forms Layer 1 of the real-time enterprise: core infrastructure that enables collaboration and notification between applications and services – whether internal or external. The ESB provides event notification, dynamic routing, and transactional guaranteed delivery; and a well-defined process language is used to allow applications to coordinate activities through a common API.

The real-time enterprise demands the ability to move the right data to the right place at the right time; JMS (the Java Message Service) provides the means for event distribution and transactional guaranteed delivery. An intelligent data fabric is also needed that can distribute information around the network for when it is required in order to increase throughput and reduce load on expensive back-end systems. The backbone of this fabric is formed by JCache – the Java universal caching framework.

A Linda-like tuple space combines the "one and only one" delivery semantics of a message queue with the broadcast capability of publish/subscribe and the loose coupling of a peer-to-peer system. A tuple space acts like an associative memory shared by an unlimited number of processes. Processes can add a tuple (essentially, a data object) into the space, or take a tuple out to work on exclusively – waiting until there is a matching object if necessary. Processes can also read tuples without removing them from the space. This paradigm – which combines the "one and only one" delivery semantics of a message queue with the broadcast capability of publish/subscribe and the loose coupling of a peer-to-peer system – is mapped on top of JCache, providing a high performance distributed implementation of the concept.

This can be combined with a business process model engine (for example, jBpm:

www.jbpm.org) to provide a rich set of distributed programming domains. Each process works independently of others – taking appropriate inputs from the tuple space and placing outputs back there for successor tasks. The order in which processes execute on a tuple can be less tightly constrained than in a traditional workflow system. This model provides distributed shared memory, generic clustering, parallel computing, and therefore the foundations for distributed workflow and BPM.

> "Over the last decade, competitive pressures have combined with technology to fundamentally change the rhythm of business"

Applications

The applications that form Layer 3 of the real-time enterprise depend on the resources of the enterprise infrastructure and communicate with each other using a collaborative programming model. Rather than developing monolithic or simple two-tier client/server applications, architects are realizing the benefits of a more loosely coupled and multi-layered component model. The adoption of standards for definition, discovery, and actual execution of this model (such as WSDL, UDDI, and SOAP for Web services) has helped to bring service-oriented architectures to fruition.

The J2EE application server that forms the basis of the virtual operating system provides a transactional secure service-based integration point for the infrastructure. As a distributed ESB is a grid-like enabling technology, a Web services interface based on the OGSI open source definitions is a natural choice. OGSI is currently the de facto standard for externalizing grid technologies and allows grid services written in one environment to be easily deployed in others.

In addition, the ESB can offer a scalable rules engine based on an optimized Rete algorithm. Externalizing business rules makes it possible to manage rapidly changing business processes, decision mechanisms and, at a lower level, message filtering and routing, without requiring code-level changes to underlying applications. That frees the business from its dependency on tardy code development cycles, allowing business-aware analysts to make the changes necessary to support the introduction of new products or regulatory requirements without interrupting system operations.

Administrative Support

The real-time enterprise requires services to manage and coordinate applications and their services at both the macro and micro levels. Layer 4 provides the administrative support necessary for implementing security polices, defining resource usage guidelines, and integrating operational processes. Essential capabilities include:

- ***Monitoring:*** Collating events and statistics in order to understand application performance, resource usage, and operational behavior. This allows for simulation, fault determination, and manual and automatic balancing of resource utilization throughout the infrastructure.
- ***Reactive coordination:*** Requiring intelligent management, control, self-annealing, and tuning of applications by heuristic analysis, dynamic rules, and flexible workflow. Through the use of efficient dynamic topologies (running the right number of applications in the right locations), the real-time enterprise manages utilization load and chooses the right hardware and location to run applications.

The ESB management fabric combines distributed JMX with statistical event collection and correlation, and the same standards-based Java rules framework that is used at the application level. This provides location transparency, discovery, remote control and collation of statistics for resource usage, performance monitoring and alert notification. These technologies allow predictive decisions to be made about the environment for intelligent resource visualization, orchestration, and provisioning throughout the real-time enterprise, delivering insight to both IT and business managers.

Conclusion

Using a distributed enterprise service bus, businesses can maximize the utilization of their existing investments in hardware and software by using standards to provide an agile real-time "service on demand" infrastructure. This agile infrastructure:

- Provides technologies that actively align IT resources so that business leaders can transform core information services to meet a changing market
- Creates a unified IT foundation built on open standards that enables flexible change for future requirements
- Lowers the cost of the IT infrastructure while maintaining high levels of performance.

These goals are achieved through a rich combination of enterprise messaging, real-time caching, and distributed active management technologies. The result is an IT foundation that comes with a lower overall cost to the business and a greater ability to meet business change as it happens. Through its reliance on standards, this real-time infrastructure pulls disparate technologies together into a continuous fabric that provides the means to rapidly align software and hardware infrastructure to serve the enterprise's real-time business needs.

About the Authors

Nigel Thomas offers independent product marketing consultancy in the application infrastructure software market place.

nigel.thomas@lyntonresearch.com

Robert Davies is the chief technology officer and founder of SpiritSoft. He is one of the leading experts in the industry for Java Message Service (JMS).

robert.davies@spiritsoft.com.

IVR for Web Services

Accommodating the mobile customer base

The use of a range of wireless, Internet-enabled portable computing devices has dramatically extended the reach of online services.

WRITTEN BY
MICHAEL X. ZIRNGIBL

& SAMUEL APARICIO

Nonetheless, complications such as wireless service gaps, time constraints, and hardware inconvenience can make contacting corporate enterprise systems while on the road difficult and inconvenient.

There is a broad spectrum of providers of online business and consumer services who furnish a range of offerings geared to "anytime, anywhere" access. These include many services that were designed and targeted to address the specific needs of mobile users, and users with travel-intensive businesses or lifestyles.

Convenience Is the Name of the Game

Online services provider Outtask furnishes corporate clients with Web-enabled, employee-facing travel, expense, and recruiting management software applications. Among its products, Outtask provides two successful Web-based travel and expense automation solutions – Cliqbook and Vinnet, respectively.

Because the two solutions are aimed at business people who are frequently on the move, Outtask realized that there was a significant Catch-22 to providing travel and expense services to traveling people: it is unreasonable to expect a client's employees to be wired to the Internet when they are riding in a cab or running through the airport.

Consequently, Outtask decided it could further extend those services – and provide a much-needed utility to customers – by finding a way to offer access to those business applications via a phone-based interactive voice response (IVR) system. Outtask set out to use the IVR platform to rapidly expand practical access to its outsourced applications to its clients' entire employee base.

Laying Out the Requirements

Outtask had a number of basic requirements in choosing a solution. First, to avoid an extended and disruptive deployment period, they would select a system without a steep learning curve. That meant the solution could not require Outtask to hire professional services or to engage in extensive IVR-specific training such as training in VoiceXML or speech recognition technology.

Second, because the move to IVR was intended to add functionality on top of existing services, the system must not require a heavy upfront investment. Outtask's natural inclination was to avoid the expense and risk of purchasing and maintaining the specialized hardware associated with many IVR solutions currently on the market.

And third, and perhaps most important, the new IVR system had to leverage Outtask's existing service-oriented architecture (SOA). With a significant investment in Web services, Outtask wanted an IVR solution that would fit in an elegant manner.

The Decision

Outtask's core expertise and skill set is around corporate travel and expense management, not in voice response systems. Outtask already had a substantial investment in its platform and was unwilling to bear the expense of building another core competence. Consequently, some platforms were eliminated from the possibilities because in order to interoperate with many of them, Outtask would have to entirely replicate its infrastructure and business logic – all just to add phone-based interfaces.

Outtask also ruled out solutions that created silo environments that failed to make efficient reuse of existing resources.

Finally, Outtask settled on Angel.com, a Web-based provider of turnkey speech recognition and interactive voice response (IVR) solutions.

Angel.com specializes in providing hosted IVR solutions, and has developed online offerings – such as IVR for Web services – that furnish companies with tools to provide on-demand phone access to Web-based applications. Angel.com has developed and maintains a broad range of pre-built components geared toward the low-cost creation of new IVR applications. Once companies create new IVR service applications on Angel.com's Voice Site platform, they can rebrand and resell those services under private labels, or integrate them with an existing Web offering.

Angel.com's online IVR product allows a motivated client to create, within hours of starting, a prototype that integrates with a company's existing system. Once an IVR application is up and running, clients can use Angel.com's online interface to manage and update it, as well as add services and capacity.

Because one of Angel.com's main service objectives is tied to adding IVR functionality to existing Web-based services, its service model fits well with Outtask's goal of making the most of its SOA – a common service layer that can be invoked by other applications.

The Solution

Once the decision was made, Outtask quickly set out to use the Angel.com system to validate the viability of an open standards hosted approach to building its IVR application. Outtask was able to deploy several same-day, proof-of-concept systems that served as vehicles to demonstrate how Angel.com technology would work with Outtask's back-end infrastructure.

The prototype applications successfully leveraged Outtask's existing Web services architecture, and the fact that Outtask could reuse its SOA for a new type of client convinced the firm's technical team that it could use the Angel.com system to voice-enable its existing enterprise data.

Plus, Outtask felt that by serving customer needs from the outset, the company would be able to clearly determine which business processes were the best candidates for a phone-based approach.

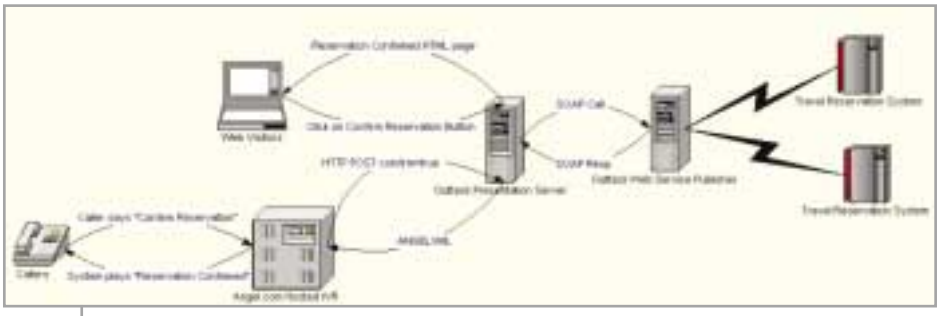


FIGURE 1 How they brought travel reservations to the phone

One Example

For its Cliqbook Travel Management service, Outtask uses proprietary technology to aggregate travel reservation content from multiple providers. These data feeds fuel a business layer that exposes a uniform Simple Object Access Protocol (SOAP) facade (see Figure 1).

Outtask's Web-based interfaces consume these Web services and use standard XSL transformations to render the graphical elements. In this case, the job of integrating the Angel-based IVR with existing Outtask systems was made easier because there was a strict separation between the presentation layer and the services it accessed. Consequently, the Web services were readily available and accessible for reuse for IVR consumption.

A significant strength of the Angel.com system was that it effectively presented itself simply as another Web client. Because of this, Outtask was able to use a different set of XSL transformations to render the responses from the SOAP calls into Angel.com's XML format for IVR interactions.

A New Service Is Born

Outtask rolled out its first phone-based applications within several days, and its first full solution in three weeks. Two new products – voice-enabled Cliqbook and Vinnet – were available in a little less than 10 weeks.

As part of the solutions, Outtask also incorporated a new breed of voice user interface (VUI) that was jointly created by Angel.com and VoicePartners, a design firm that devises solutions for a range of speech recognition and telephony platforms.

About the Authors

Since Angel's inception in 1999, Michael Zirngibl has played a leading role in establishing Angel's business, technology, and product strategies. Before his work at Angel, Michael served as senior product manager for MicroStrategy's Broadcaster and Telecaster products. In this role, he initiated, defined, and managed the industry's first intelligent outbound voice messaging platform.

Sam Aparicio oversees product management at Angel.com and is responsible for product development, partnerships, and technology evangelism. From its origins as a research project, Sam has spearheaded the effort to turn Angel.com into a world-class platform, full of innovations and breakthroughs.

–Continued from page 18

services. To keep complexity in check, BPEL does not directly support BPM- and workflow-specific constructs, such as tasks or roles. The explicit XML-based representation of BPEL process logic allows BPM functions to be provided independently on top of the existing language capabilities without adding to the complexity of the language. Workflow functions can be similarly provided through the notion of infrastructure services, for example, a task service for handling human interaction with a BPEL process (see Figure 1).

In a clean slate enterprise environment, all IT resources that need to be orchestrated into business processes would already be exposed as Web services. The prevailing reality is that this is seldom the case and existing legacy applications as well as newer applications are commonly exposed through a variety of proprietary interfaces and protocols. Supporting such environments in a pragmatic fashion becomes possible by using a WSDL binding framework (see Figure 2), such as Apache WSIF (Web Services Invocation Framework). This binding framework can support an arbitrary set of protocols in addition to SOAP, effectively decoupling resource connectivity from process design. The WSDL interface allows such resources to be viewed as Web services, ready for orchestration into a BPEL process, while native protocols are used to connect to them.

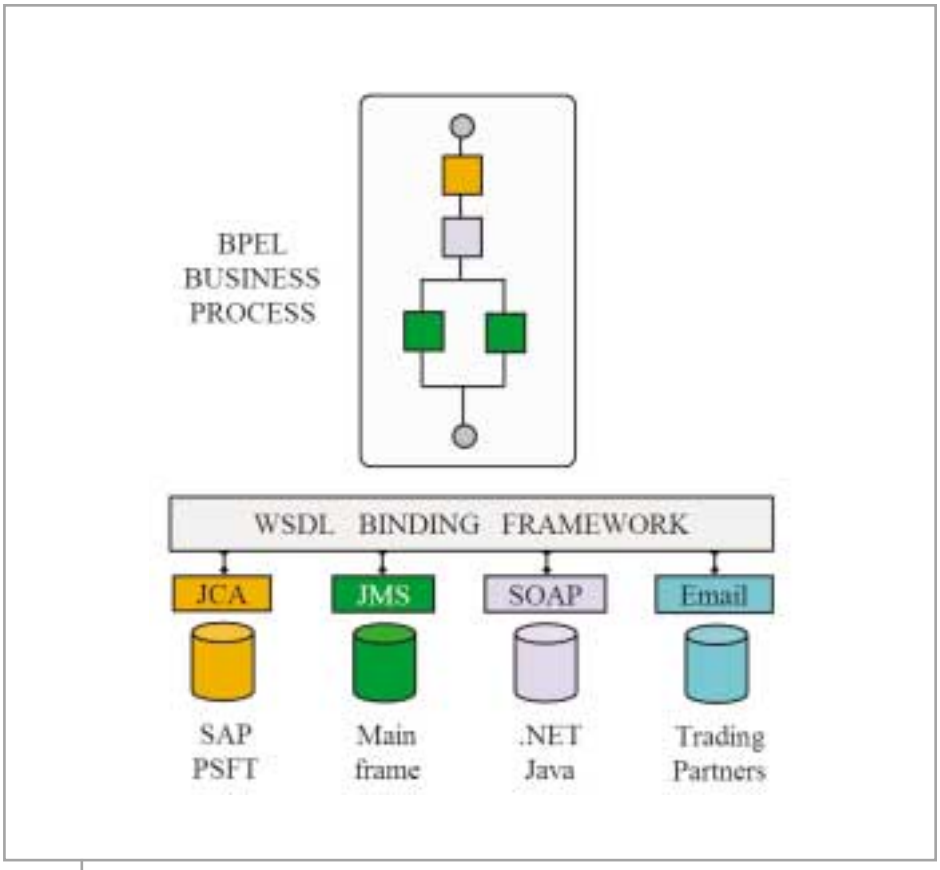


FIGURE 2 | **WSDL binding framework**

The BPEL Environment

Business integration scenarios that are appropriate to implement in BPEL commonly include several of the following technical requirements:

- Access heterogeneous systems
- Multiparty data transformations
- Asynchronous interactions (state management, correlation)
- Parallel processing (sophisticated join patterns)
- Compensation logic ("Undo" operations)
- Exception management
- Reliability and scalability (high performance)
- Operations management (auditing, monitoring)
- Change management (side-by-side versioning)

BPEL is quickly becoming a mandatory requirement for customers evaluating BPM and workflow solutions as it inherently addresses many of these technical requirements and others that are supported by any mature commercial BPEL server implementation. The benefits of BPEL far outweigh the risks associated with adopting an emerging standard, considering the typical shortcomings of existing BPM and workflow solutions: high cost, complexity, and unavoidable vendor lock-in. BPEL avoids these issues by drastically reducing the skill level required for implementing process logic and seamlessly fitting into the interoperable Web services stack. Most importantly, BPEL provides a portable process-logic representation with a growing number of vendors to choose from for BPEL process execution.

Unlike existing BPM and workflow solutions, BPEL frees customers from having to choose an all-in-one solution and naturally leads to component choice and the utilization of standards-based architectures. Sun's JSR 208 (aka Java Business Integration, www.jcp.org/en/jsr/detail?id=208) describes one such blueprint that includes a BPEL process engine in a best-of-breed integrated framework for business integration. JSR 208 demonstrates that BPEL can elegantly integrate with existing infrastructure, in particular with the Java/J2EE platform. Finally, BPEL relies on XML as a universal data model for message exchange between a process and its related services. This further enables BPEL to interoperate with a wide variety of value-add application components, most of which now support XML for communication, e.g., rules engines, transformation services, and more.

BPEL and SOA

The introduction of Web services is creating a material change in the integration space, mostly due to the success and ROI of numerous Web services projects. Experience shows that the technologies for implementing a proper service-oriented architecture exist and work. The important aspect of this architecture is that it allows an organization to maintain a focus on solving business problems and not just application problems.

From this perspective, solutions to business problems can now be addressed and implemented in a consistent and methodical fashion using this architectural framework.

It's important to note that there is a distinction between Web services and a service-oriented architecture. SOA is truly just an architecture and is independent of any particular set of technologies, including Web services. It is defined as an application architecture where functions are defined as loosely coupled (read: independent) services, with well-defined interfaces that can be called in specified sequences to form business processes. Web services, at the moment, act as the set of standards and technologies (such as SOAP, WSDL, and related XML standards) that collectively serve as the foundation for implementing SOAs.

With the above definition, it is apparent that SOAs will serve to address integration problems, leveraging services as building blocks. These building blocks will be composed and coordinated to form business processes, defining the role of BPEL in this architecture. The reusability and flexibility of services leads to integration problems from a top-down perspective rather than the traditional bottom up. Consequently, integration based on SOA can be addressed one problem at a time, rather than having to deploy upfront expensive and complex integration infrastructure throughout an enterprise prior to addressing any integration issue.

Services are then created to address specific integration problems and can later be reused to address new problems. Over time, collections of existing reusable services can be used to fully address an organization's integration problems. In that light, the value of SOAs grows incrementally as services are implemented and added as assets. Furthermore, with composition and coordination of services enabled by BPEL, functionality can be encapsulated at multiple levels to create more valuable services that can be leveraged across a wide range of new applications.

BPEL in Use

Commercial support for and implementations of the BPEL standard are rising at a fast pace, as evidenced by recent product announcements by leading platform vendors, EAI vendors, and startups. BPEL engine implementations span both the Java and .NET platforms while other vendors have added support for BPEL code generation from visual models created in their tool environments. Customer inquiries indicate that a growing number of end users are evaluating the use of BPEL for mission-critical projects. Such evaluations typically involve a technical hands-on product evaluation, a proof of concept, and sometimes an initial production deployment to prove the maturity and ROI of the approach. At this stage of adoption, many prospects are still discovering the proper criteria for evaluating BPEL tools and engines and treating their early implementations using BPEL as a stepping stone to more mission-critical applications.

However, as is commonly the case, certain industries are quicker than others to adopt new standards and technologies. Such industries include financial services, telecommunications, and various service providers to both business and government

entities to name a few. Early examples for the use of BPEL in commercial applications include:

- ***Resource management workflows:***
  - *Company*: ISV offering workplace resource management solution to Fortune 500 companies
  - *Operation:* BPEL server bundled with product offering, using BPEL processes to manage workflows
- ***New Policy Issuance Process:***
  - *Company:* Fortune 50 insurance and annuities issuer company
  - *Operation:* Selling policies to consumers through brokers and agents
- ***Service provisioning process:***
  - *Company:* Information services provider (to global space agencies)
  - *Operation:* Supply satellite data to commercial and government customers
- ***Enterprise-wide integration standards:***
  - *Company:* Multinational Consumer Product Manufacturing Company
  - *Operation:* Low-cost simplified integration solution for online markets

The common drivers for automation in the above use cases are reduction of cycle times and processing costs as well as minimizing or eliminating manual coordination tasks (which commonly contribute to less efficient, error-prone processing). The choice of BPEL in all of these use cases involved both business requirements as well as technical requirements.

Business requirements include:

- Lowest cost of acquisition and ownership
- Processes in a standard way to avoid vendor lock-in
- Support for both automated and human workflow

Technical requirements for the above include:

- Support for Web services standards and XML
- Scalability and reliability
- Portable
- Reusable
- Easily modifiable

The Road Ahead

BPEL is on its way to becoming the cornerstone of SOA implementations in an enterprise environment, responsible for coordinating disparate resources and services into end-to-end business processes. The explicit XML-based representation of the BPEL process allows for extensive management, reporting, and analysis functions to be added on top of the process execution layer. These functions can be provided by the BPEL engine vendor of choice, or by ISVs who provide value-add components that can enhance an overall solution structured around a BPEL engine. While some of these value-added functions are available in commercial BPEL implementations today, we believe that this is just the tip of the iceberg. The next 12 months should be very interesting and offer significant ROI opportunities for the companies that are ready to move to business process management with BPEL.

About the Author

Doron Sherman is the CTO of Collaxa, Inc., a BPEL server vendor located in Redwood Shores, California. He has been involved with Java since its early days and pioneered application server technology while the founder and chief scientist at NetDynamics.

doron@collaxa.com

–Continued from page 12

driving application platform vendors as well as systems management vendors to think that they "own" Web services management – even though, for the most part, the vendors are not building Web services management. As with anyone coming from a certain area of experience, their belief is that the problem entirely fits within their domain. To a hammer every problem is a nail. Systems management vendors are building in passive monitoring of Web services and application platform vendors are building in functionality such as routing of Web services – and both think that this gives them a complete solution for Web services management.

As the big vendors become more aware that Web services management is both a critical capability for maintaining a service-oriented architecture and that critical components of it are outside their experience base, they will scramble to bring the technology in-house, since only the largest vendors have significant expertise in both areas (though usually in product teams that rarely talk).

Regardless of whether a fundamental change is on the way for enterprise IT, the second- and third-tier Web services management players are likely to be shaken out of the market or acquired. The opportunity for the first-tier vendors, however, is to reach escape velocity and become successful independent vendors – demonstrating that Web services management isn't "owned" by any traditional market segment.

About the Author

As chief technology officer at Actional, Dan Foody leverages his extensive hands-on experience in enterprise systems integration software toward easing integration through Web services. He is an active participant in the Web services standards community, including WS-I and OASIS, where he spearheads Actional's contributions on the OASIS Management Protocol Technical Committee and its efforts to deliver XML-based Web services management standards.

dan.foody@actional.com

WSRP and the Enterprise Portal

A strong, and important, start

Web portal software has emerged as one of the most important components of software enterprises over the last few years. That success has carried with it the challenge of how to integrate disparate software services into the portal – services that can live across multiple platforms, operating systems, and networks. Solving that challenge is a key reason for the development of the new WSRP specification.

WRITTEN BY
ED ANUFF

WSRP, or Web Services for Remote Portlets, is an OASIS standard whose goal is to provide Web service-based access capability for portal servers. Up to this point, most efforts to add capabilities or features to portal servers have involved the use of proprietary APIs and protocols that differ from vendor to vendor. WSRP will standardize the way that portals communicate remotely with remote services that can extend the portal's core capabilities.

One unique facet of the specification is that it is a presentation-oriented Web service. Most Web services are expected to just carry raw data as the result of a request, and the caller is responsible for determining how it is used. While in an abstract sense this is also true of WSRP, it is different in the sense that WSRP generally carries fully rendered markup that is to be included within a portal page, with very few changes to be made by the consumer.

The recent release of the JSR-168 specification – called Portlets for short – is also noteworthy. The portlets specification is a Java-based API for creating an open standards–based portlet interface. This standard was developed at the same time as WSRP, and each expert group made it a priority to ensure the two would be able to coexist. This is important, since the two standards are a natural fit to work together.

The Basic Portal

At this stage in the evolution of WSRP, the specification addresses some very basic use cases for integration of portlet services through the use of Web services. Chief among these are the proprietary standards for integrating remote content, as described earlier. Standardizing the protocol enables developers and end users to more easily populate their portals with content from a variety of sources, with very little or no custom programming required.

A typical scenario can be seen in Figure 1, which shows how a user has selected a portlet from the list of available ones and placed the portlet within the portal page. In our example, the portal has two remote content sources via WSRP and one local source available through the local JSR-168 container. This is a fairly simple scenario showing that WSRP can be used to include remote portlets seamlessly into a portal that also pulls portal content from local sources.

Some of the advanced scenarios that WSRP can help solve will be discussed in detail later in this article. For now, let's start with some basic background on what comprises WSRP.

The Details

WSRP is made up of four distinct Web service interfaces: service description, registration, portlet management, and markup. To allow for varying levels of support by producers (generally the remote portlet server) and consumers (generally the end user–facing portal), not all of these interfaces are required to be implemented by the producer or to be used by the consumer.

Due to space constraints, the level of detail required to truly understand the protocol cannot be discussed in this article. Those interested in a more complete overview should visit the WSRP Web site (www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrp). Examining the WSRP primer and specification there will help yield a better understanding of what the protocol offers and requires of its users.

Service Description

The ability to determine what services a producer has available for use is the first step for a consumer to integrate these services into the portal. When requesting the service description, a consumer has the option to supply its registration credentials to the producer. While this information isn't required, a producer could use it to change the services for which it returns a description.

The purpose of the service description structure is to inform the consumer, or portal, of the capabilities of each portlet it offers. This is important because it affects whether or not a portal can display the markup contents of the portlet along with what modes and window states it advertises as being available. In addition to these functional details, the producer also returns a list of locales that each portlet supports.

The service description structure, like every other structure in WSRP, supports an extensions field consisting of an array of type Object that allows both the client and server to support custom features that weren't included in 1.0.

Registration

When a consumer registers with the producer, it submits two types of data as part of the process. The first set of data describes the capabilities of the consumer. An example of this is the window states and modes the consumer supports. The second set of data is the registration properties, described

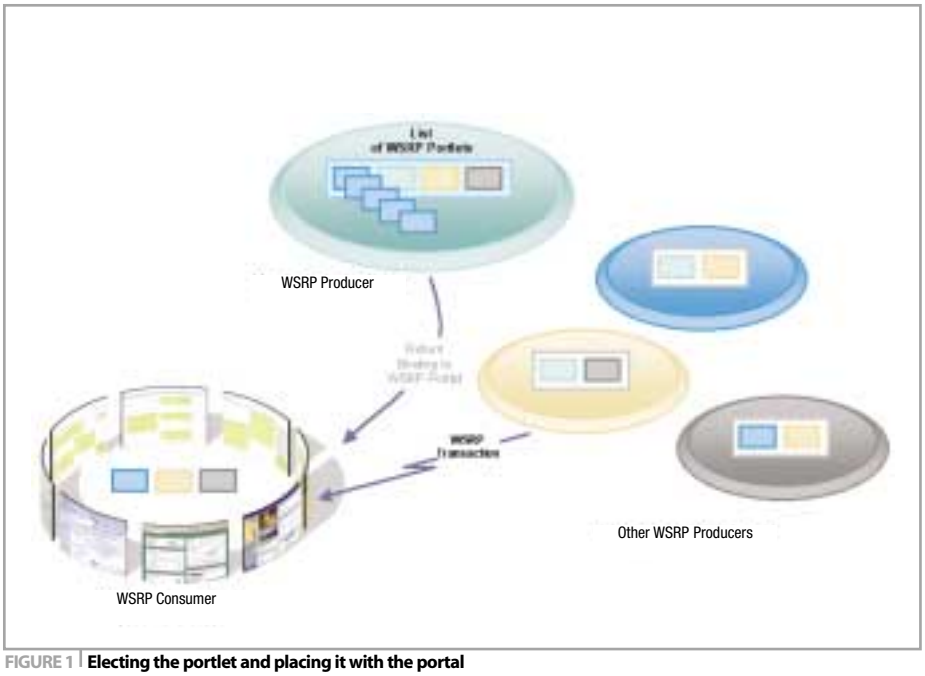


FIGURE 1 | **Electing the portlet and placing it with the portal**

in the producer's service description, that it requires for registration.

Registration is an optional interface that producers are not required to support. Producers that don't support registration at all, or those that only support registration in an out-of-band process, will not expose the interface to consumers.

The end result of registration, be it from an in-band or out-of-band process, is a registration handle that is unique to that consumer and from which all portlets created under this registration will be scoped.

Portlet Management

Portlet management is the second optional interface in WSRP. Its purpose is to allow the consumer to manage the portlets exposed in the producer's service description. When this interface is exposed, it allows the consumer to clone one of the producer's portlets for the purpose of customization. The original portlets exposed in a producer's service description are non-modifiable, which in effect means that if a producer does not expose this interface, its portlets can't be configured on the consumer portal. This limits the types of services that can be exposed but allows for a much simpler producer in cases where this level of functionality is not required.

Markup

Markup is a required interface as well as the one that will be invoked the most by the consumer. The interface has operations that support markup generation and interaction processing.

Both markup generation and interaction methods require very similar parameters for their operations to succeed; therefore, everything discussed in this section applies to both methods unless otherwise noted. Along with the normal identification and request-related context information (such as registration, request parameters, and portlet state), the consumer generally will send across end-user profile data such as first name, last name, username, and e-mail address so that the service can adequately personalize the consumer's offerings.

The key difference between a call for markup content and interaction is that the call to markup is just for markup, and is not allowed to affect state that the consumer keeps for the particular portlet for the particular end user. Interaction, however, allows the portlet to update its state along with returning markup when the call completes. To better illustrate this concept: clicking on a portlet or interacting with it will trigger a call to the interaction operation, while a refresh of the page will cause each WSRP portlet to just request markup, since it is not the current target of action on the page.

Another important capability is the format of the markup that is returned by these calls. The producer generally will have to encode all URLs that refer to content within the portlet so that the producer can rewrite them and put them in the context of the portlet for later proxy to the producer when needed.

Example

Declare a secure interaction back to the portlet:

```
wsrp-rewrite?wsrp-
urlType=blockingAction&wsrp-
secureURL=true&wsrp-
navigationalState=a8h4K5JD9&interactionS
tate=fg4h923mdk/wsrp-rewrite
```

Dynamic/Advanced Producers

What follows is a more interesting case about how WSRP can be used to create some very powerful portal services from formerly nonportal-centric services.

One of the most compelling reasons for adopting the use of WSRP is that it allows portals to integrate services and content into the portal. These services and content can be exposed on just about any platform or operating system and implemented in just about any language that supports XML-based Web services.

These producers can be like the simple case illustrated earlier, where the producer returns essentially static content with a static service description. That's all that is required in some cases. But for more compelling and useful applications, the need for a dynamic producer becomes greater. Dynamic producers offer a larger degree of flexibility and the ability to create new offerings on the fly. For example, imagine that an integration engine and application generator are exposed via WSRP, and that each newly configured integration component is then added to the service description. This provides a means for exposing any integration component directly into a portal, regardless of the integration platform, as long as it supports WSRP production.

The ability to quickly and easily add existing applications and services directly into the portal is a very compelling objective for companies trying to leverage existing applications inside their enterprise portal.

What's Next

As with all first versions of a specification, there is always a laundry list of items that couldn't be fit in; otherwise, 1.0 would still be in committee. The items below are planned for inclusion in the upcoming 1.1 and 2.0 versions of WSRP. The 1.1 version is scheduled to be completed sometime in 2004, while 2.0 is likely to arrive toward the end of 2005 or the first half of 2006.

> "WSRP generally carries fully rendered markup that is to be included within a portal page, with very few changes to be made by the consumer"

Security

Security was not addressed at all in the 1.0 specification, other than allowing a portlet to be marked as requiring the connection be secure. There is no provision for passing along, say, a user token from an SSO system or other means of authenticating a user to the producer. The current plan is to incorporate support for WS-Security along with guidelines on how it should be used within WSRP to ensure interoperability among vendors. In 1.0 it is left up to each individual vendor to determine how to attempt to secure WSRP usage. This will potentially lead to a number of issues when trying to interoperate between vendors and address application security. Support for this is planned in the 2.0 version.

UDDI Support

UDDI support will allow producers to post information about their services on UDDI servers to make it easier for consumers to search for and find their offerings when the location of the server that hosts these servers is not known. Version 1.1 will add simple support of UDDI so that a producer can describe its presence as well as each of the services it offers. This is essentially a subset of the data that is found in the service description.

Version 2.0 is expected to introduce more detailed structures to provide support for categorization, among other things. There are still some questions among the technical committee as to what role the specification should take in describing how UDDI should be used by producers.

Cross-Portlet Communication

This is the next large feature planned for introduction in WSRP 2.0. It will provide a mechanism that allows portlets to broadcast event information to other portlets spread across multiple producers if required. The key use case for this feature is so that portlets can post contextual information about their interaction, and other portlets can use that information to tailor the content that they generate.

Conclusion

Multiple vendors including Citrix, Oracle, Sun, Vignette, IBM, BEA, and Plumtree have expressed support for the first version of the WSRP specification and announced plans to provide implementations within their product offerings. This is a very strong vote of confidence among portal vendors. Developers and other IT organizations should investigate the use of WSRP when integrating their various service offerings into the enterprise portal.

About the Author

As vice president of product strategy, Edward Anuff is shaping Vignette's go-to-market strategy and maintaining a line of award-winning products and services. He joined Vignette following their acquisition of portal software company Epicentric, where he was chairman, cofounder, and chief strategy officer. Since the acquisition, he has taken an active role in the integration of the companies. He is the author of the *Java Sourcebook* (J. Wiley and Sons), one of the first books on the Java programming language.

anuff@vignette.com

FREE*CD! ($198.00 VALUE!)

Secrets of the Web services Masters

Every WSJArticle on One CD!

— The Complete Works —

CD is edited by WSJEditor-in-Chief Sean Rhody and organized into 40 chapters containing more than 400 exclusive WSJarticles!

All in an easy-to-navigate HTML format! **BONUS: Full source code included!**

ORDER AT WWW.SYS-CON.COM/FREECD

**PLUS $9.95 SHIPPING AND PROCESSING (U.S. ONLY)*

SYS-CON MEDIA

Only from the World's Leading i-Technology Publisher

Stateful Interactions in Web Services

A comparison of WS-Context and WS-Resource Framework

In July 2003 a consortium of Web services vendors released the Web services Composite Application Framework (WS-CAF) to the community. WS-CAF is comprised of three specifications that together provide a means of reliably composing individual Web services into larger aggregate applications. The cornerstone of this suite is the management of stateful interactions between Web services that is the domain of the WS-Context specification. WS-CAF was subsequently submitted to OASIS and an effort to standardize the framework is currently underway.

WRITTEN BY
JIM WEBBER

SAVAS PARASTATIDIS

& MARK LITTLE

In January 2004 a group of industry and academic practitioners from the Grid community released (the first parts of) the Web Services Resource Framework (WS-RF) specifications. WS-RF will support stateful interactions between consumers and resources hosted by Web services.

Clearly there is some overlap between the WS-Context and WS-RF approaches since both support stateful interactions on top of the stateless interaction model championed by Web Services Architecture (WSA). This article examines the different approaches taken by WS-Context and WS-RF, concentrating in particular on how each approach facilitates stateful interactions in composite Web services–based applications.

WS-Context

A composite application is a set of actions executing on a collection of Web services that executes in a specified sequence. The ability to scope arbitrary units of distributed work (known as activities) is a requirement in a variety of aspects of distributed applications, e.g., workflow, business-to-business interactions, automated business processes, and others. By scoping work, participants within an activity can determine unambiguously whether or not they are in the same activity.

In order to correlate the work of services participating within the same activity, it is necessary to propagate additional information, known as a *context*, to those participants. The context contains information like a unique ID and allows a series of actions to share a common outcome. In WS-Context, SOAP headers carry context information that is propagated with application-level messages. This context allows multiple participants to correlate SOAP message exchanges in order to create a larger abstraction such as a process flow, secure conversation, or other aggregation.

While context propagation is a fundamental requirement of many distributed systems, including Web services, the type of context information that is used may vary depending upon the circumstances. For example, in a transactional system it may be a URI for the coordinator, whereas for secure data interchange it may be the sender's public encryption key. Accordingly, WS-Context was developed as a standardized means of conveying context information to Web services.

The WS-Context specification also defines a context service that can be used by Web services to form composite applications. Since each requirement for context may require different information to be conveyed, WS-Context provides a minimalist (but extensible) context that allows services to register context affiliations and customize contexts on a per-activity basis.

WS-Resource Framework

The Web Services Resource Framework (WS-RF) was produced in response to input into the OGSI process by the Web services community (e.g., the WS-GAF proposal). Its authors, IBM and the Globus Alliance, proposed WS-RF as the "convergence point" between Web and Grid services, and WS-RF has been positioned as the natural evolution of the Open Grid Services Infrastructure specification (OGSI).

WS-RF follows the same conceptual model, which is based on resource sharing, that underpins OGSI, but without altering the underlying Web services specifications. WS-RF adopts many of those suggestions, especially in the areas of factorization, contextualization for modelling stateful interactions, clear separation between the concepts of a "service" and a "resource," and the unmodified use of existing Web services technologies.

The suite of specifications that makes up WS-RF has not yet been released in its entirety; only the specifications that describe resource state, resource lifetime, and notification have been made available, while those concerned with service groups, resource reference renewal, and faults will be released at a later date. Since the focus of this article is on stateful inter-

actions, we will focus on that aspect.

In the WS-RF conceptual model, a Web service is a stateless entity "that acts upon, provides access to, or manipulates a set of logical stateful resources (documents) based on messages it sends and receives." The model encourages the explicit exposure of resources (logical or physical) across the boundaries of a service. The representation of the state of these exposed resources and the way in which consumers may interact directly with them is the primary goal of WS-RF.

Supporting Stateful Interactions

While there are similarities between the goals of WS-RF and WS-Context, the specifications opt for different approaches (resource manipulation versus service composition). In particular, they differ on how to deal with supporting stateful interactions across Web services, with WS-RF advocating a resource-oriented approach, while a service-oriented approach is advocated by WS-Context.

To demonstrate the use of WS-Context in supporting stateful interactions, we will examine the simplest use case, where a stateful interaction is held between a single service and a single consumer.

We have discussed how WS-Context defines the notion of an activity to which the context is bound. Activities ensure that all interactions on a WS-Context–aware service will be uniquely and unambiguously tied to that activity through the context. In the simple case, the context is used by the consumer to identify a particular stateful interaction, and by the service to identify a specific conversational state.

In WS-Context the context life cycle is as follows:

1. To initiate an activity, a service requests a new context from the WS-Context service via a *begin* message. The initiator may specify a time limit for the session, or it can be set to live until explicitly terminated. Depending on the application requirements, the context may be also created by a service automatically on receipt of the first request from a consumer.
2. The *begin* action will return a *begun* message plus a context.
3. Whenever the consumer interacts with a WS-Context–aware service, the context is propagated in a SOAP header block. The receiving service should manage any context-specific state that it requires in order to correlate messages.
4. The stateful interaction can be terminated either by timing out, or by explicitly instructing the context service to end the activity.

WS-RF promotes implicit contextualization as a mechanism for stateful interactions between consumers and resources exposed outside the boundaries of a service (it does not model context as an external entity as does WS-Context). Such resources, logical or physical, are identified through WS-Addressing constructs. In addition to information about service endpoints, these constructs also contain information specific to resources, which are placed in the <wsa:ReferenceProperty/> element of a WS-Addressing construct.

When a consumer engages in a stateful interaction with an identified resource, it has to include the contents of the <ReferenceProperty/> element as a header in each SOAP message sent to the service identified by the <EndpointReference/> of the same WS-Addressing construct. The service receiving the message will use that information to route invocations to the resource (see Figure 1).

WS-RF mandates that a WS-Addressing construct is opaque to its consumers and so they should not try to utilize that resource-specific information. The information about the resource is considered private to the service and should be used only by that service. In effect, WS-Addressing constructs are used by WS-RF as network-wide pointers to resources (see Listing 1). Each consumer is required to include the <example:DataSetId/> element in the header of each SOAP message, which results in some explicitly identified action to be taken on the resource (e.g., a message requesting that the identified dataset be sorted or deleted). This element is used by the recipient service to identify and delegate invocations to the correct back-end resource.

As a network pointer, a WS-Addressing construct with resource-specific information fulfills the same purpose as a CORBA IOR, DCOM OBJREF, Java RMI URL, etc.; it identifies a resource across the network. In an approach that is similar to existing object-based, distributed-computing technologies, WS-RF pushes the issue of resource identification down from the application layer and makes it part of the Web services stack.

By requiring the identity of the resource to be passed as a header in each SOAP message, WS-RF models stateful interactions with specific resources rather than services. In combination with the additional specifications that offer lifetime management of exposed resources and a mechanism to renew the references to those resources, the WS-RF shares many concepts with distributed-object models.

> "There is some overlap between the WS-Context and WS-RF approaches since both support stateful interactions on top of the stateless interaction model"

The Filestore Example

To represent the issues raised in the previous sections, we will explore a hypothetical implementation of a simple Web services–based filestore. (*Note:* The filestore example was used because it is both simple and has been a canonical example for demonstrating the pros and cons of both WS-RF and WS-CAF within both communities.) In this example, the filestore implementation is simplistic; for clarity we ignore issues such as policy and security,

though in a real implementation both would be critical. An overview of the filestore implementation is shown in Figure 2.

Accessing the filestore using WS-RF is straightforward. A WS-Addressing endpoint reference to a specific resource is obtained (via some out-of-band mechanism like a registry or factory). This endpoint reference (extended with WS-RF–specific metadata) acts as a network-wide pointer to the resource hosted by the Web service. The endpoint reference obtained can be used as both an address to which messages can be sent, and as an implicit context for interacting with the back-end resource (using the contents of its <Reference Properties/> element).

In the case of the filestore, the file ID (or i-node or some other descriptor) can be used to provide the necessary metadata to enable the service to route invocations to the same resource for each message sent to the service (see Figure 3), where the consumer sends the message to the service endpoint identified by the WS-Address and the resource-related metadata is used to assist the service in routing to the correct back-end resource.

While the implementation of the WS-RF scheme is SOAP friendly (using the SOAP headers and WS-Addressing), developers should take care that they do not violate encapsulation by directly exposing private enterprise resources to the wider network. The main danger with WS-RF is that it encourages exactly this behavior. This in turn leads to applications that are brittle and difficult to maintain. The key weakness of this approach is that should the service wish to evolve (for example, if the filestore implementation migrates from a single file system to a database-driven configuration), the identity information captured in the endpoint reference metadata may become stale and thus the stateful session will fail. This is why additional mechanisms, like lifetime management and renewable references, are necessary parts of WS-RF. While it is possible to avoid such problems by using logical identifiers (which are resolved by the service into physical resources), it is not mandated by WS-RF. (*Note:* The WS-Context approach does not suffer this drawback since contexts are third-party entities entirely decoupled from the implementation of any service.)

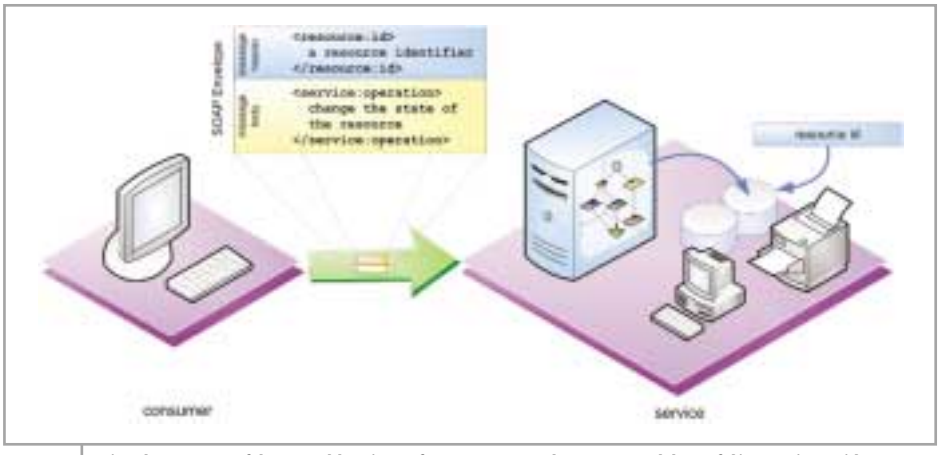


FIGURE 1 | **Using the contents of the WS-Addressing ReferenceProperty Element to model stateful interactions with resources**

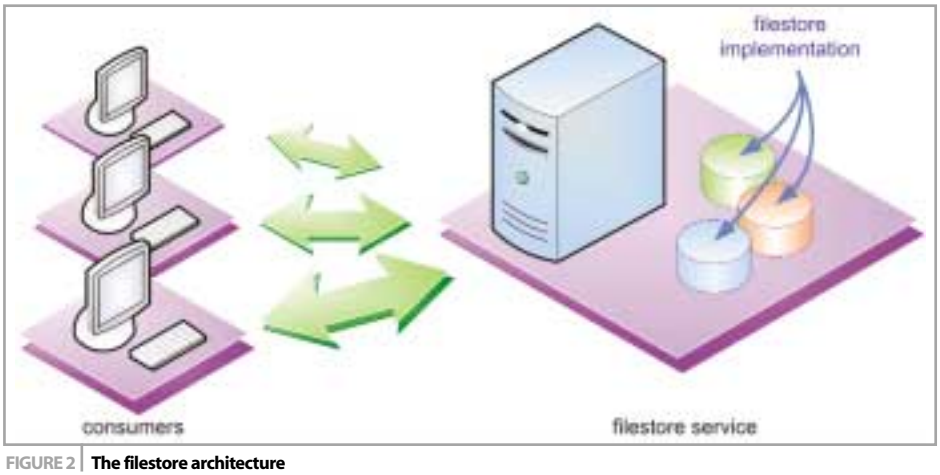


FIGURE 2 | **The filestore architecture**

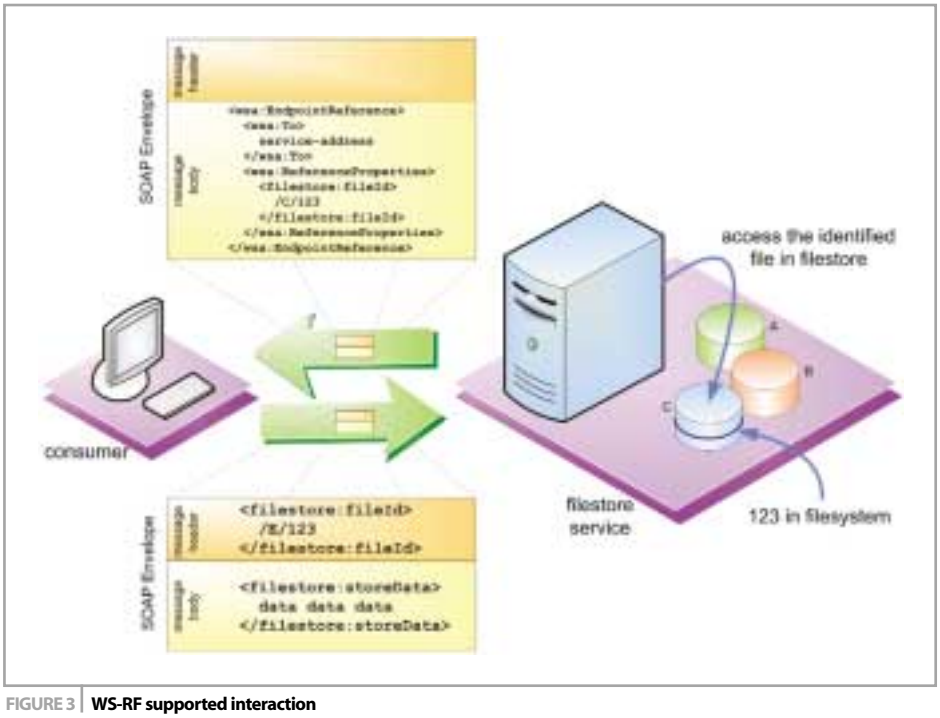


FIGURE 3 | **WS-RF supported interaction**

In this constrained scenario, the WS-Context approach is not entirely dissimilar from the WS-RF technique. A context is generated by some out-of-band means (such as a context service) and is embedded in a SOAP header block with every application-level message sent to the filestore. The filestore undertakes the action corresponding to the receipt of that message, using the context information to ensure the correct state and resources are used to serve the action.

Unlike the WS-RF approach, the context in Figure 4 is not an identifier for any back-end resources, but is an external entity that allows actions to be logically linked. WS-Context does not try to model service-side resources since this is considered out of scope, yet stateful interactions can still be supported. Because WS-Context takes the view that a service's implementation is private to that service, how context information is used to correlate messages into stateful interactions is left to the service architect. This means that while WS-Context–aware services are interoperable, no implementation choices are forced upon developers. As such, WS-Context respects the view of a Web service as independently evolvable and where no information (logical or otherwise) about the configuration of the service escapes from within its boundaries. Since context information exists independent of any context-aware service, those services can evolve as they see fit without jeopardizing the validity of future contextualized interactions.

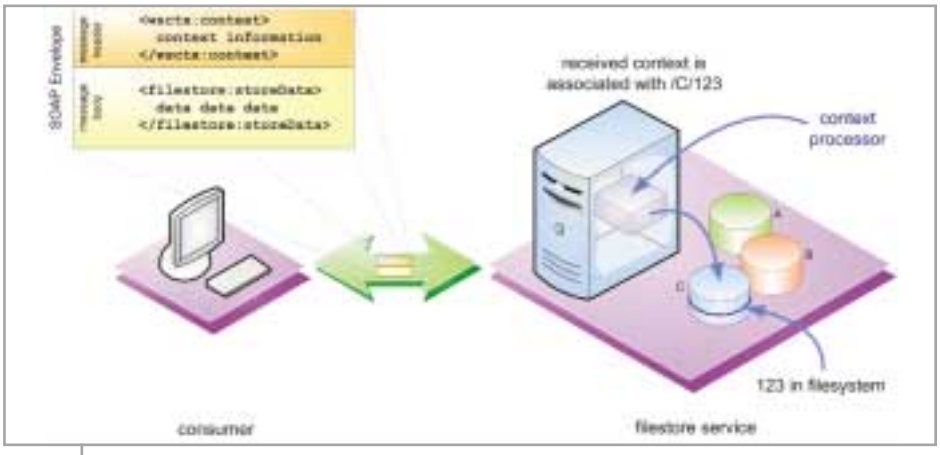


FIGURE 4 | **WS-Context supported interaction**

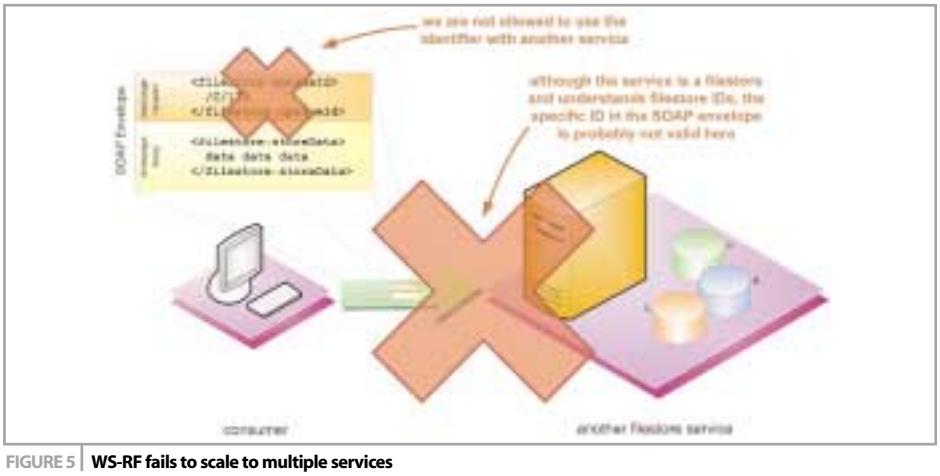


FIGURE 5 | **WS-RF fails to scale to multiple services**

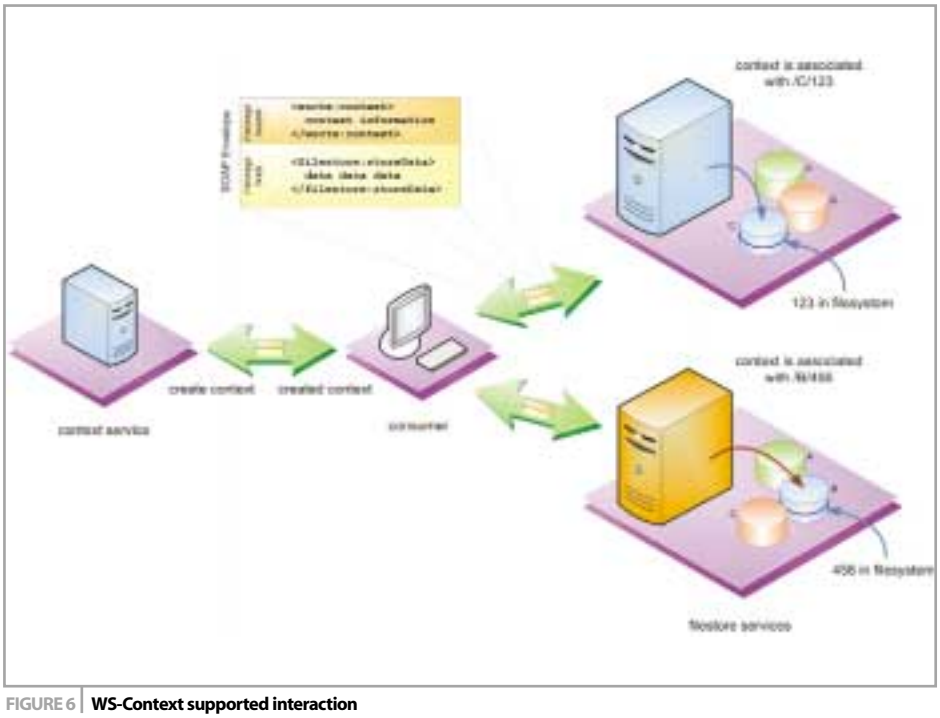


FIGURE 6 | **WS-Context supported interaction**

Scaling WS-RF and WS-Context

If WS-RF is used judiciously, then it can support single-party interactions in a similar way to WS-Context, where the contents of the WS-RF endpoint <ReferenceProperties/> element is used in place of the WS-Context context. In certain constrained circumstances WS-RF is an even simpler solution since it requires no additional protocol actors, whereas most WS-Context–enabled services are designed to take part in distributed activities and thus require a context management service, or internal means of generating a context.

However, unlike WS-Context, which treats context as an externally shared entity, the WS-RF model does not scale well past the simple consumer-service interactions since resources are identified with

service-specific information, which is used to contextualize interactions. To illustrate this, consider Figures 5 and 6.

In Figure 5 we are confident that the pointer-like mechanism that underpins WS-RF is valid when used to communicate with a resource hosted by a specific service. However, since that endpoint reference is service specific, the same resource-related metadata cannot be used when communicating with a second service.

Furthermore, since there is nothing to prevent the storage of WS-Addressing structures with resource-related metadata and service endpoint information, long-lived interdependencies between resources may be formed. Such interdependencies are difficult to maintain in large-scale systems and cause applications to be brittle.

In Figure 6, because context is modeled as a known, standardized, external entity, any WS-Context–aware service that receives it will be able to apply the context information to its own internal processes. Given that the context is explicitly managed and external to any individual service it is visible and valid to all services within the activity. On receipt of a context, the service can use the information to correlate messages to back-end resources, including the use of that information to discover the wider application context within which the action will be executed (i.e., the other services are participating), and thus stateful distributed activities are possible.

While distributed activities can be achieved after a fashion using WS-RF (by manually propagating all of the endpoint references in use to all services in use), scoping a distributed application by using collections of point-to-point addresses is inherently difficult, and leads very quickly to a combinatorial explosion of endpoint references that have to be managed, propagated, and kept up-to-date. Conversely, only a single entity, the context, is required in the WS-Context approach.

Conclusion

While both WS-Context and WS-RF can be used to enable stateful interactions between Web services and their consumers, the models they adopt to achieve this are very different, and as a consequence the scenarios in which they are best deployed are also different.

The WS-RF approach is based on an addressing scheme for back-end resources hosted by Web services. This addressing information can be used as a means of correlating and routing message exchanges with those back-end resources and thus as a means of achieving stateful communication.

WS-Context assumes that the back-end implementation details of a service are private. It is deliberately noninvasive and deals only with context management and propagation of contexts to services. What precisely is done to map a particular application level message plus context onto specific back-end resources is safely out of scope.

For single consumer-server interactions the WS-RF approach is certainly lightweight. However for interactions involving multiple services, the WS-Context approach scales readily to support distributed activities. Thus WS-RF might be suitable as a point-to-point solution for integrating two systems, but in the general case with systems composed from many Web services, WS-Context is the natural choice.

Acknowledgments

The authors would like to thank Professor Paul Watson (University of Newcastle upon Tyne) for his contributions to this article.

References

- *OASIS(WS-CAF), Web services Context (WS-CTX):* www.iona.com/devcenter/standards/WS-CAF/WSCTX.pdf
- *Web services Resource Framework (WS-RF). 2004:* www.globus.org/wsrf
- Parastatidis, S., et al. (2003). "A Grid Application Framework Based on Web Services Specifications and Practices. www.neresc.ac.uk/ws-gaf
- Tuecke, S., et al. (2003). "Open Grid Services Infrastructure (OGSI) - Version 1.0". https://forge.gridforum.org/projects/ogsi-wg
- Frey, J., et al. (2004) "Modeling Stateful Resources with Web Services."
- *Web Services Addressing (WS-Addressing):* http://msdn.microsoft.com/ws/2003/03/ws-addressing.

About the Authors

Dr. Jim Webber is a senior researcher from the University of Newcastle upon Tyne, currently working on the convergence of Web services and Grid technologies at the University of Sydney, Australia. He was previously a Web services architect with Arjuna Technologies where he worked on Web services transactioning technology, and was one of the original authors of the WS-CAF specification. Jim is an active speaker and writer in the Web services space, and coauthor of *Developing Enterprise Web services – An Architect's Guide.*
jim.webber@newcastle.ac.uk

Dr. Savas Parastatidis is the chief software architect at the North-East Regional e-Science Centre (NEReSC), Newcastle upon Tyne, UK, where he is NEReSC's expert in Web services and Grid computing technologies and standards. Previously, he co-led the research work at HP's middleware division that led to the development of the world's first XML-based transactioning system and represented HP during the early stages of the OASIS BTP standardization effort. Savas' corner of the Web is at http://savas.parastatidis.name.
savas@parastatidis.name

Dr. Mark Little is chief architect, transactions, for Arjuna Technologies Limited, a spin-off from HP that develops transaction technologies for J2EE and Web services. Previously, Mark was a distinguished engineer and architect at HP Middleware, where he led the transactions team. He is a member of the expert group for JSR 95 and JSR 149, the specification lead for JSR 156, and was one of the original authors of the WS-CAF specifications. Mark is the coauthor of *J2EE 1.4 Bible* and *Java Transactions for Architects.*
mark.little@arjuna.com

Listing 1

```
Example of a WS-Addressing with
resource-specific information
<wsa:EndpointReference
xmlns:wsa="..." xmlns:example="...">

<wsa:Address>http://www.example.com/
webservice</wsa:Address>

<wsa:PortType>example:ResourceOperat
ions</wsa:PortType>
  <wsa:ReferenceProperties>

<example:DataSetId>dataset15</exam-
ple:DataSetId>
  <wsa:ReferenceProperties>
</wsa:EndpointReference>
```

XML JOURNAL SUPPLEMENT

This Month

Heterogeneous XML-Based Information Integration

XML and XQuery for an effective real-time solution

BY JULIE BASU AND NIRAV CHANCHANI

Information integration within the enterprise doesn't have to be a problem. Emerging XML-based technologies can be used to effectively solve the puzzle. See how combining XQuery and a J2EE-based application server can give you a powerful, flexible architecture for synthesizing information in real time.

Semantic Mapping, Ontologies, and XML Standards

The key to managing complexity in application integration projects

BY DAVID S. LINTHICUM

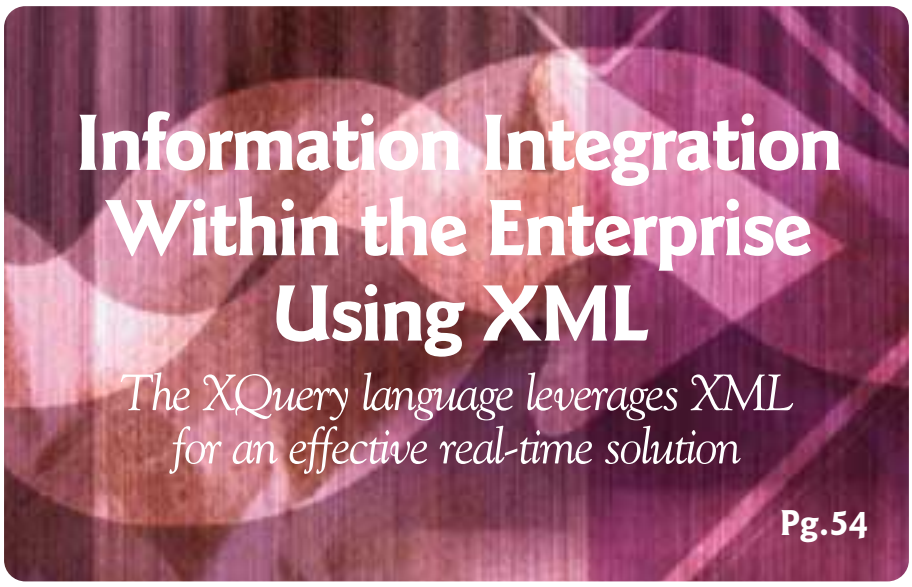
Dealing with application integration means dealing with complexity. The notion of ontologies can help you prepare generalizations that make the problem domain more understandable. This article discusses using ontologies to formalize the management and integration of information, services, and processes.

Programming & Design: How to Reduce the Burden on Web Designers

Leveraging designer tag libraries with XSLT

BY PETER HORSFIELD

XML is being applied on all business fronts for internal and external data exchange, transcoding between domains, archival, and powering dynamic and sometimes real-time Web sites. Yet all this technological weaving ties together more than just data; there's also a human aspect. As XML usage has grown ubiquitous, a web of dependency between Web designers and programmers has developed. This article shows how to eliminate the interdependency between HTML design skills and XML processing.



XML-Based Interop, Close Up

In addition to the strategy side of Web services, there is also the protocol-oriented side of things, the XML side. Embracing not only XML itself but also the full range of mainstream XML-based technologies like XPath, XSLT, XML Schema, and SOAP, *XML-Journal* has been delivering insightful articles to the world of developers and development managers since the year 2000.

It is our privilege to bring *XML-Journal* directly to readers of *Web Services Journal*, and vice versa. Anyone already familiar with the Web services world of SOAP, UDDI, and WSDL will find here articles and features each month that will interest them – about the cutting-edge technologies and latest products that are changing not only our industry, but the way the world exchanges information. To make it easy for you to find your way around, we have four distinct sections:

Content Management:
Organization, dissemination, and presentation of information

Data Management:
Storage, transformation, representation, and general use of structured and unstructured data

Enterprise Solutions:
Systems and applications that manage mission-critical functions in the enterprise

Labs:
Product reviews, book reviews, tutorials, and standards analysis

Information Integration Within the Enterprise Using XML

WRITTEN BY
JULIE BASU &
NIRAV CHANCHANI

The XQuery language leverages XML for an effective real-time solution

This article addresses the problem of heterogeneous XML-based information integration within the enterprise. First we examine the business problem and the technology options currently available to build a solution. Next, we discuss how emerging XML-based technologies can be used to effectively solve the problem. Specifically, we show how XQuery in conjunction with a J2EE-based application server can be the basis of a powerful and flexible architecture for synthesizing information in real time from heterogeneous sources across the enterprise.

Business Problem

A large organization generally has multiple applications to manage its business information. These applications are domain-specific: they're designed to address a certain set of business operations or optimized for a specific business function, such as customer order entry. Accordingly, each application is very good at handling information within its own application domain, but its view is generally restricted to that domain. A business or enterprise may need information that spans multiple applications to construct a full 360-degree picture of its customers or operations.

For instance, consider a common Web-based business application: a self-service customer order–tracking site. Such a site lets customers log in and get the status, payment, and package delivery information for their orders. To build a complete view of the orders, the application may need to access information from multiple sources, including:

- An order entry application for orders placed by the customer
- A catalog file with item details
- A financial system for information on payments received
- A customer management application to view customer profiles
- A shipping and distribution database for products shipped to the customer
- A package delivery tracking system to view the delivery status for the customer

The information sources in such an application have several distinctive characteristics:

- The sources are heterogeneous in nature. Each may have its own protocol and access method. For example, the financial system may be accessible via a Web service method call, while the shipping and distribution database may be accessible only via JDBC.
- Each source may be maintained and operated independently by different departments, across geographical jurisdictions, and even different businesses. Updates may happen asynchronously to the various sources. The systems must be accessed in real time in order to get the latest information.
- Each source holds domain-specific information that may be largely distinct from that of other sources. For example, the system for tracking the delivery status of a package may belong to a delivery service such as FedEx or UPS, which is a separate entity from the business accepting customer orders. The tracking system wouldn't be directly concerned with information in the order entry application. Hence, in this situation it wouldn't make sense to replicate all the information in a central database or warehouse.

The problem organizations face is how to build an information integration application that requires real-time access to information from multiple sources, each of which has its own data format, method of access, and communication protocol.

Given this problem, how do you build a flexible solution in the shortest amount of time? The architecture must be extensible so it can easily accommodate future growth, such as the addition or modification of information sources. Likewise, if the queries handled by the system need to change over time, you shouldn't extensively recode the application.

Technology Options

Most IT organizations looking to solve the problem have three options:

- Use enterprise application integration (EAI)
- Build a data warehouse
- Write a custom application

The EAI approach

EAI technologies don't offer a particularly good solution to our business problem, as they primarily allow one application to talk to another. This means that an event (such as an update) in one application triggers a workflow process that causes an action to take place in another application(s). The EAI model consists of publish-subscribe interactions, where one application can publish an event and one or more applications can subscribe to it (the request-reply model is also common). Furthermore, EAI necessarily requires knowledge of both endpoints. In many cases, enterprises just want to pro-

vide streamlined access to data to future applications, whose interfaces are not known in advance.

An EAI-based solution is not an ideal candidate since we need to aggregate information from various sources, not integrate the applications themselves. In some cases an EAI solution can be adapted to solve this problem; however, it may require a lot of custom coding and convoluted logic to achieve data aggregation.

The warehouse approach

The data warehousing solution involves a central repository where all the data is loaded into a predefined data model and then queried. The model is generally constructed a priori based on the queries that are expected to run against it. This approach has the following disadvantages:

- ***Replicated information:*** Replicating dynamic information from a large number of independent information sources into a central repository may not be feasible.
- ***Data freshness:*** The data in the repository gets stale unless it's constantly refreshed. In some cases, such as inventory information, synchronizing the data periodically would never be as good as accessing the information in real time in today's on-demand environment.
- ***Limited connectivity to data sources:*** Most data warehouse vendors provide gateways to other databases or data repositories, but not to packaged applications or legacy systems, which are often proprietary.

The custom code approach

Given the limitations of EAI and data warehouse solutions, the only viable approach left for IT organizations is to devise their own solution. They may implement it using any popular technology such as J2EE or Microsoft .NET. (This article assumes that J2EE is the technology of choice for application development.)

The J2EE platform provides a rich set of APIs for data access and manipulation in the middle tier. However, the custom-coding approach has several serious drawbacks:

- ***Time intensive:*** A purely programmatic solution requires programmers to use various Java APIs to extract data from the information sources and then join and sort the results. Although J2EE APIs are rich, they're also complex and prolific, so this is not a trivial programming task. Some of the APIs that are typically involved are Java Database Connectivity (JDBC) for database access, Java Connector Architecture (J2CA) for accessing packaged and legacy applications, and JAX-RPC for accessing Web services. In addition, application-specific code is needed to weave together the data fetched from the various sources in order to compute the desired result.
- ***Poor extensibility:*** Adding or modifying an information source typically requires changes to the programming logic. For example, if a source was changed from a database to a Web service, the application logic is likely to be affected.
- ***Lack of generality:*** Unless a general query-processing engine is embedded in the solution, the type of queries that can be posed to the application will be severely limited. The logic to join and merge the various pieces of data is typically quite proprietary and doesn't easily extend to arbitrary types of joining and merging.
- ***Lack of reusability:*** Unless the solution is carefully crafted with reusable components, most of the custom logic will be application-specific. For example, suppose that after developing the customer order–tracking, self-service application, we need to develop a customer support application that accesses the same sources but gets more detailed information. Since this requirement didn't exist when we developed the customer order–tracking application, the code isn't likely to be reusable – so the entire programming effort would have to be repeated from scratch. This duplication of code causes a lot of complexity, making it difficult to maintain and evolve multiple applications across the enterprise.

XML-Based Information Integration

The advent of J2EE standards–based technologies for the middle tier, like JDBC and J2EE Connector Architecture, along with powerful technologies such as XML and XQuery, enables a new solution to the problem of real-time heterogeneous information integration. As you'll see, these technologies let you quickly build a middle-tier solution that's flexible and extensible. The solution is based on using an information integration service, a software infrastructure module that supports the generic execution of XML queries and has pre-wired access to different types of data sources.

An XML-based real-time information integration solution using a J2EE-based application server and a middle-tier information integration service consists of the following steps:

1. The information integrator (someone familiar with the data sources of the enterprise) creates reusable aggregate XML views for accessing data across the enterprise.
   -The declarative XQuery language discussed in this article is ideal for this purpose.
   -Design tools can help you formulate complex queries.

2. An application invokes a selected XML view.
   -A view-naming mechanism is useful at design time and/or runtime. The design tool can help you list predefined views available across the enterprise.

3. The information integration service materializes the requested view by executing the associated query. This step requires:
   -Accessing information in diverse sources using standard J2EE APIs. This can be done in a generic way independent of the application logic.
   -Translating the information into XML, as necessary. It's best to provide a framework for plugging in custom or standard translation modules.
   -Feeding the XML sources and the selected query into an XQuery engine to compute the result. Again, this can be done in a generic way, independent of the application.

4. The XML results are returned to the application. The results can be made accessible through standard J2EE client interfaces such as Enterprise JavaBeans (EJBs) and Web services. The data retrieved from the client APIs may be consumed by the application for display, reporting, or other purposes, e.g., exposing the information on an enterprise portal.

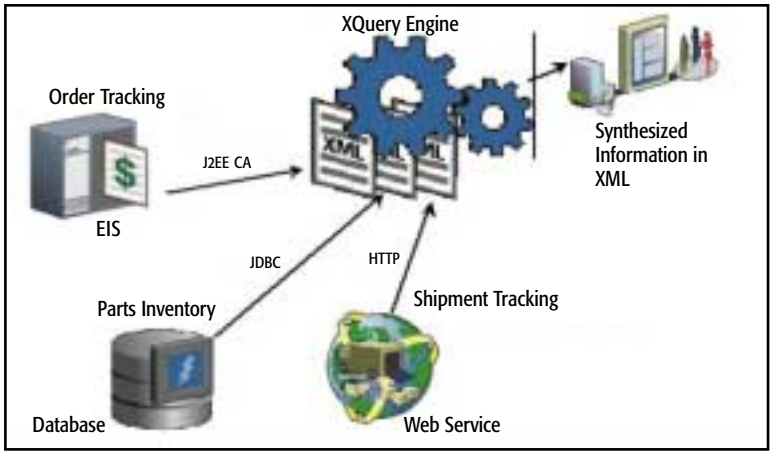


Figure 1 • XML-based information integration

Figure 1 illustrates the process. The set of XML views prescribed essentially defines a virtual data access layer across the enterprise. These XML views may be reused by different applications or to construct hierarchical views. The underlying information integration service is a critical component of the solution, helping to minimize duplicate logic in application-specific code and greatly increasing programmer productivity by reducing application development time. This service needs to be built as a first step in solving the information integration problem.

Why XML?

XML is an ideal choice for information integration because it provides a standard format for data representation and exchange. An XML document consists of markups with start tags and end tags around a piece of data or other nested tags, much like tags in an HTML document. The tags make it easy for a program to parse the various components.

Listing 1 shows an example of an XML file for a catalog that includes item names and prices. The well-formed tags make the document self-explanatory.

Due to its power and simplicity, XML is being adopted rapidly. Various supporting standard technologies are available to process XML data. For example, the XPath standard developed by the W3C is a notation to identify and extract individual sections or fields within an XML document. For further reading in this area, see the reference section.

The Power of XQuery

XQuery is a new query language standard for XML being developed by the W3C. It's designed to perform filtering, transformation, and join operations across one or more structured XML documents. The basic construct in the XQuery language is the FLWOR expression, which consists of:

- ***For clause:*** Similar to the SQL "from" clause
- ***Let clause:*** For declaring variables
- ***Where clause:*** Similar to the SQL "where" clause
- ***Order By clause:*** For sorting
- ***Return clause:*** For the result, similar to the SQL "select" clause

Listing 2 is an example of an XQuery that extracts the name and price from the XML catalog shown in Listing 1, and declaratively joins the suppnum field to get the supplier name and address from another XML document called suppliers.xml (not shown here). The Where clause in the query is used to select only those items whose price is less than 100, and the Order By clause sorts by the supplier name. Note that the XPath notation is used to identify individual fields within the XML documents.

The declarative power of XQuery should be obvious from this example. It's similar to the benefits that SQL provided in relational databases compared to custom coding with file systems. XQuery is designed to handle hierarchical data in XML format and defines a rich set of standard functions and operators for data transformation. For more information on XQuery, see the references section.

Solution Summary

XML-based information integration is an important middle-tier service that provides synchronous access to data synthesized from multiple sources. The service provides a set of XML views forming a "virtual data access" layer across the enterprise that can be used by different applications such as portal, integration, and other J2EE-based applications using a standards-based API. New applications can get secure access to consolidate data without having to program the consolidation manually each time. The information integration service also helps to minimize duplicate logic in application-specific code and greatly increases programmer productivity by reducing application development time and time to market. Furthermore, as part of the grid-computing model, the service can be used to cache data in the network and provide data as a networked service.

Conclusion

XML and XQuery can form the basis of an effective real-time information integration solution. There are many benefits to this approach, including a declarative and reusable query framework and predefined XML views. Applications written using such a methodology are flexible and extensible, and can be developed in a much shorter timeframe than allowed by a custom-coding approach. The J2EE platform provides a rich set of APIs that can be used in conjunction with XML and XQuery, and specifically to access a diverse set of sources including databases, packaged and legacy applications, and Web services.

References

- *XML:* www.w3.org/XML
- *XQuery:* www.w3.org/XML/Query
- *Java API for XQuery:* http://jcp.org/en/jsr/detail?id=225
- *J2EE Platform:* http://java.sun.com/j2ee/

Author Bios

Julie Basu is a senior development manager in the Java Platform Group at Oracle. She leads the JavaServer Pages and middle-tier XML efforts for the Oracle Internet Application Server (iAS) product. Julie has more than 12 years of design and development experience and has worked on various projects relating to database programming interfaces. Julie has MS and PhD degrees in computer science from Stanford University.

Nirav Chanchani is a principal product manager, Server Technologies, at Oracle Corporation. He has more than nine years of experience with transaction processing and middleware technologies. Chanchani has a bachelor's degree in computer engineering from Pune University, India, and an MBA from San Jose State University.

NIRAV.CHANCHANI@ORACLE.COM

JULIE.BASU@ORACLE.COM

LISTING 1

```
<catalog>
  <items>
        <item>
              <name> Item 1 </name>
              <price> 50.00  </price>
              <suppnum> S1 </suppnum>
        </item>
        <item>
              <name> Item 2 </name>
              <price> 75.00 </price>
              <suppnum> S2 </suppnum>
        </item>
        … … …
    </items>
</catalog>
```

LISTING 2

```
for $item in document("catalog.xml")//item,
    $supplier in document("suppliers.xml")
     //supplier [suppnum=
               $item/suppnum]
    where $item/price < 100
    order by $supplier/suppname
     return
        <item>
           { $item/name,
             $supplier/suppname,
             $supplier/address,
             $item/price
           }
        </item>
```

LOOK FOR YOUR FREE...

Coming this **SPRING!**

WRITTEN BY **PETER HORSFIELD**

Programming & Design: How to Reduce the Burden on Web Designers

Leveraging designer tag libraries

The whole point of teams is to allow different specialties to complement one another and achieve the extraordinary, so it can only be a good thing to reduce the barriers between them. This article shows how to eliminate the interdependency between HTML design skills and XML processing.

XML is being applied on all business fronts for internal and external data exchange, transcoding between domains, archival, and powering dynamic and sometimes real-time Web sites. Yet all this technological weaving ties together more than just data; there's also a human aspect. Engineers trade schemas and specifications between one another, and each engineer is expected to understand and be able to integrate those specifications. This, of course, is an accepted facet of our lives, but it belies a problem.

As XML usage has grown ubiquitous, so has this web of dependency. First between software engineering and data modeling specialties, and lately into design fields such as Web design. Designers are increasingly required to integrate Web page designs with presentation of XML data sources.

This has created a market opportunity for tools such as Altova's Stylevision product. Stylevision is a singular achievement that allows a Web designer to visually craft data-driven Web pages, with the end result being not HTML but instead an XSLT style sheet capable of transforming XML instance data into the desired page. Thus, software is provided to hide the increasing complexity of the design task (see Figure 1).

All server-side languages, from Perl CGI scripts to Python, provide a means to embed data in a page (see Figure 2). However, all these languages are deficient in their ability to separate the work of the designer from that of the Web programmer. The inventors of XSLT recognized that creating HTML from other forms of XML data would be a powerful tool and gave us the simplified model of XSLT.

Neither the simplified model nor the full-blown version alter the notion that there is a single file that contains both transformation and presentation elements. The remainder of this article will show that XSLT *can* be used to elegantly separate the concerns of the programmer while supporting those of the designer.

The Goal

Initially, server-side HTML consisted of procedural code placed in blocks that were somehow *escaped* out of a regular HTML document. With the advent of XML and XHTML, the trend has been toward using new tags similar in appearance to HTML. The HTML is still well formed.

We'll build a similar tag library using XSLT. A transformation file will be created that can be applied to merge an HTML presentation document with an XML data document. Any tags used in the HTML document will be replaced according to the design of the tag library. If no tags are used in the HTML document, the output will be similar to the input. Furthermore, the tag library will be domain specific and thus meaningful to the Web designer.

Reasons for Using XSLT

By limiting the data source to an XML document, we lose all the flexibility of the general-purpose tag libraries, but in return we also gain many benefits. Designers will find this XSLT-based solution easier to use than the other server-side solutions because:

- The designer does not have to work around arbitrarily complex server-side code.
- Web browsers can display local files without having to understand server-side directives.
- Transformations can be automated on the client side using live or test data.
- It can be integrated with graphical design tools such as Macromedia's Dreamweaver.

Furthermore,

- This solution can leverage XML/XSLT accelerator technologies.

Decoupling developer from designer

Our solution does not come with the baggage associated with embedding a general-purpose language. The simplicity of Listing 3 as compared with Listings 1 (PHP) and 2 (JSP/JSTL) is gained by moving all the logic into the XSLT style sheet. Listings 1–10 can be downloaded from www.sys-con.com/xml/source.cfm. (Listing 3 is well formed but not valid because XHTML documents contain only XHTML markup. After undergoing the XSLT transformation we are going to create, it will be valid, too.)

Any server-side solution that requires a single document to contain configuration or general-purpose code will suffer this bloating to some extent. The distinction is more obvious when using the verbose DOM extensions in PHP4 than when using others. In fact, the PHP page must also load the data XML document to process, but I omitted that for brevity.

The XSLT-driven form offers the designer a level of legibility by default that is far greater than that offered by a general-purpose server-side scripting language.

WYSIWYG IDE integration

Most XML-aware IDEs allow you to

AUTHOR BIO

Peter Horsfield has been developing software for the engineering and automation industry for the last six years. He recently spent a year and half building on the Apache Cocoon XML framework to deliver real-time radar processing data. In a break from tradition, he now works as a configuration management analyst for Constellation Power Source in Baltimore, Maryland.

Accessing Live Data

The document() function is implemented by the major XSLT processors (at least Saxon, Xalan, and Microsoft's). It allows an XSLT transformation to access XML outside of its primary input. The document function acts as an alternative root in XPath expressions and can even be assigned to variables. I highly recommend using the document function to load a top-level variable, and then referencing the variable elsewhere. In XSLT processors that load the entire target URI, this ensures that only one access is made – a major optimization.

view an XSLT+HTML page in a browser as it would be seen in multiple live situations, but not to edit the live view. Similarly, HTML WYSIWYG designers can usually handle HTML with embedded scripting, but require a test server on which to execute the script and hence to see the result. This solution is compatible with both types of tool.

Macromedia Dreamweaver allows you to add custom tag libraries through the Tag Library Editor (see Figure 4). Each tag can be manually added to Dreamweaver, or one of the many tag library file types can be imported. This includes support for loading XML Schemas.

Working locally

The difference between the two forms of custom tags in Listing 3 is in how expressive you want to be. Browsers ignore tags they do not understand, but continue into them when rendering. The upshot is that the designer can easily see what the live page would look like. For example, Listing 3 would be displayed locally as:

```
"Product Name {product}. Price is ."
```

After the transformation, "{product}" would be replaced with the actual product name.

Testing locally

Usually you will obtain your working data from a source other than a live feed. However, this is no more complex than passing in a different URL to the XSLT transformation. (The ability to select the source document was intentionally added to the design.) No more uploading to a test server just to test a simple data change, and you can batch up multiple mock data files to be run as unit tests or for the designer to see the page in use in various situations.

The same XSLT transformation that would execute on the server side can be performed with the command-line modes of tools such as Saxon or Xalan. For a more user-friendly approach, many XML-aware IDEs (e.g., Borland and Altova) allow you to configure transformation scenarios that can be applied with just a few clicks.

Leveraging XML/XSLT accelerators

XML accelerators are an emerging type of hardware device that allows XML documents to be routed based on introspection of the data. Recently, XSLT accelerators have turned up that can perform ultrafast XSLT transformations (e.g., the DataPower XA35 XML+XSLT Accelerator). There is no such hardware accelerator technology for server-side scripting.

Implementing an XSLT Tag Library

Figure 3 shows the designer segregated from coding and data-modeling tasks, and left dealing with only our design schema, an artifact of the coding effort. The data and design schemas are elevated to true contracts between specialties.

The design schema specifies the set of designer- and domain-oriented tags available. To add live data to a document, the designer would just drop in one of these tags. At some point we perform an XSLT transformation that removes those tags and replaces them with actual data.

Our task in this article is to create the merge.xslt file that will perform the replacements. To continue we're going to need some live data.

You'll find that Listing 4 shows a simple XML file containing some neat facts about the solar system. (An XML Schema for this file is available at www.sys-con.com/xml/sourcec.cfm.)

Simple XSLT Copying

When an XHTML page is fed into a style sheet with no templates, the input is stripped of all elements and only text content is output. So the first step is to create a style sheet that performs a copy without modifying the document, at least not functionally. (The page is completely rewritten by the transform, so some minor changes may be experienced as an artifact of this process.)

Listing 5 shows an XSLT style sheet for copying XHTML. The <xsl:output/> tag causes the XSLT processor to indent the output and to represent the output as a strict XHTML document (this only affects the document type processing instruction). The single template matches attributes and any node, and copies it to the output stream as we require. This copier is the foundation upon which we will build our data-merging system.

Replacing Designer Tags

The next step is to intercept those tags in the input that we're interested in. Our designer tags are distinguished from XHTML tags through the namespace to which they are assigned, so we add these namespaces to the style sheet. Now we can create a new template that will react to one of these designer tags. We'll pick a simple one to start with, an element that inserts a copyright string into the output.

This modification will cause any

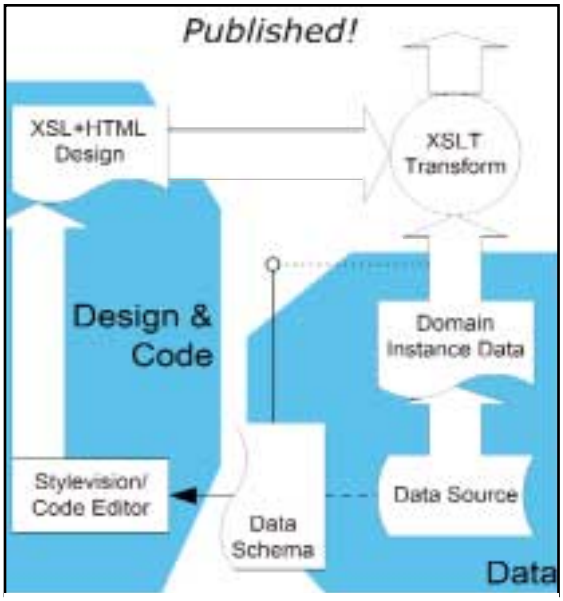


Figure 1 • Designing with XSL today

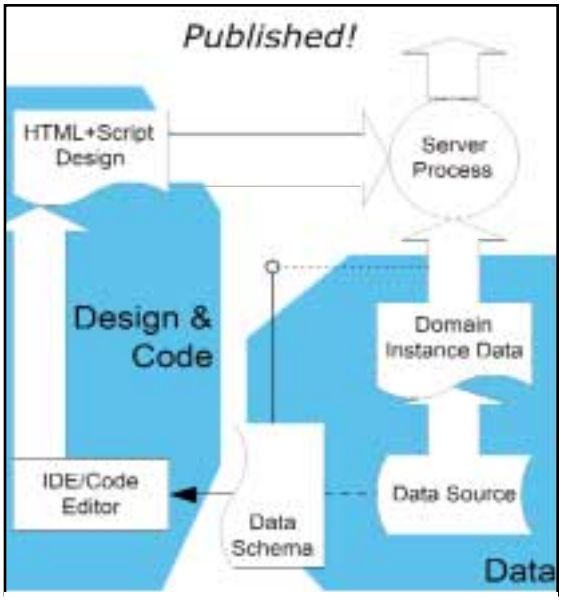


Figure 2 • Designing with scripting

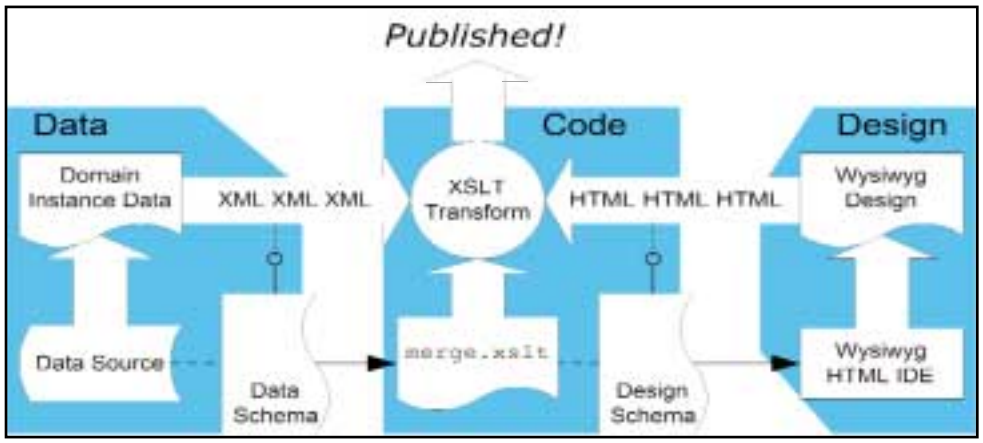


Figure 3 • Architecture and flow

copyright tags (which must be in the namespace http://xml.grumpykitty.biz/designer/stars/1.0) to be replaced with the copyright string *(C) 2004 Grumpy Kitty Productions*.

The XSLT document() function is used to access our data source. We must remember to add the data namespace to the style sheet.

Listing 7 shows a full style sheet that disregards its input and instead outputs the number of planets in the Sol system. The style sheet is decoupled from the source by passing in the URI as a style sheet parameter. This allows us to maintain Sol system data for multiple universes.

This style sheet will ignore any of its standard input and merely output the number of planets. Here is a sample command line for Xalan:

```
java org.apache.xalan.xslt.Process
-IN empty.xml
-XSLT mergedata.xsl
-PARAM dataSource solar.xml
```

Listings 5, 6, and 7 are easily combined to copy XHTML documents, replacing designer elements with live data.

Tags Within Tags

The complexity arrives when we try to represent relationships within the data set. We could specify separate tags for Star *n*, Planet *n*, Moon *n*, and so on. More realistically, we can provide tags for the designer that represent repeating elements in the source data. Whereas previously our designer tags were replaced with real data, these tags are not replaced with anything. They serve only to identify portions of the template XHTML page that should be duplicated for every data point. I call these tags grouping nodes.

Given this XHTML fragment:

```
<body><ul>
  <star:planetGroup>
    <li>Planet: <star:name /></li>
  </star:planetGroup>
</ul></body>
```

The intent is that all of the tags within the <star:planetGroup/> should be duplicated for each planet in the input data. This includes both XHTML and designer tags.

Grouping Implementation

Every time we reach such a grouping node in the template XHTML, we must duplicate the entire contents of the grouping node for each matching data point. However, the grouping node itself should not be copied.

Iteration over the source data is performed with the <xsl:for-each/> element as shown in Listing 8.

Note: The standard "dot" operator is overridden by the <xsl:for-each/> element. To work around this, I store the current node (i.e., the <star:planetGroup/> node in the XHTML document) in a local variable.

The duplication performed by <xsl:apply-templates/> is different because we must first lose the <star:planetGroup/> node and record the current data point from the source data. We can do the first by creating a separate template that just skips the <xsl:copy/> element and specifies mode="ignore".

Preserving the current source context is a little more complicated. The XHTML document being copied is arbitrarily deep, and we cannot use top-level variables for this purpose because parameters and variables in XSLT are immutable. Instead, we pass the current source context node(s) as one or more template parameters to every copying or grouping operation. Grouping nodes can then use variable hiding to specify new contexts. Listing 9 shows this technique in action.

Groups Within Groups

There are at least three semantic groups within our source data: stars, planets, and moons. Each of these groups must be maintained with <xsl:with-param /> and <xsl:param /> pairs. This must happen during the copy and during <xsl:for-each/> iterations.

The notion that moons are tied to planets, and that planets are nested within star systems, is embodied by the declaration of the <xsl:for-each/> element. Just as we can iterate over the contents of the document top-level variable, we can also iterate using the variables holding our context. Listing 10 shows such a template for the <stars:moonGroup /> grouping node.

The following code sample shows a leaf template that places the orbit of the current moon into the output by referencing the newly established moonGroup parameter.

```
<xsl:template match="designer:moon-
Group//designer:orbit">
   <xsl:param name="moonGroup"/>
   <xsl:value-of
select="$moonGroup/orbit" />
</xsl:template>
```

In this implementation it is very easy to add new leaf designer tags, and a little more complicated to add new groups. From the designer's standpoint, some of the XHTML is simply wrapped in a new, obviously custom, tag. When viewed after the transformation, the XHTML has been duplicated as many times as there are data points. This lends itself very well to building XHTML tables, lists, and so on.

Simplifications and Optimizations

Skipping XSLT template parameters

I've mentioned that all grouping and copying nodes must take the group parameters and pass them through. This is only the case if they are needed further down the XHTML tree. For the closest parent grouping node to a simple designer tag, this is almost certainly true. On the other hand, these simple designer tags may only need to access one of the several parent groups. Not every little piece of information about the moon needs to reference the earth!

Reusing designer tags

To simplify use of – and minimize the number of – designer tags, we can replace designer tags in the XHTML based upon their location. For example, <stars:name/> can be replaced with the name of a moon if it is under a <stars:moonGroup />

grouping node, and with the name of a planet if it is under <stars:planetGroup />.

This result can be obtained by:

1. Using <xsl:if /> or <xsl:choose /> to conditionally test for context
2. Using quite complex XPath statements to determine context
3. Using additional template parameters to track context

For the example given above, solution 3 can be used. The new template parameter (called latestGroup here) must be added to the duplication template. Every time a grouping node is encountered, the latestGroup variable is hidden with the current iterated values.

When processing reaches our designer tags, the latestGroup parameter always contains the correct node in the source data and is simple to use:

```
<xsl:template match="designer:name">
  <xsl:param name="latestGroup" />
  <xsl:value-of select="$latest
  Group/data:name" />
</xsl:template>
```

In the source distribution, you'll find examples of all three solutions.

Potential Issues

The primary issue with this solution is speed. XSLT transformations can easily create a bottleneck, and this solution uses XSLT extensively. If your data is rapidly changing, I recommend using a hardware accelerator or another solution. I highly recommend creating unit tests that enforce timings during development.

Also, the document function steps outside the bounds of the XSLT environment. This may expose you to nuances of the host environment. Ensure that your host environment and the XSLT processor play nicely together.

Integration

The main purpose of this article is to alleviate work on the designer's part. To this end, the designer's tag library we created can be used within IDEs such as Macromedia's Dreamweaver. To do this, I created an XML Schema that specifies the set of designer tags; however, the schema import functionality leaves a little to be desired. A workaround is to create a mock tag library descriptor file (.tld) that can be imported. It's also possible to create an XSL transform to perform this conversion.

Download the source distribution to find the dreamweaver.tld file and follow these steps inside Dreamweaver:

1. Select Edit>Tag Libraries from the menu.
2. Click the + icon and select JSP>Import from file.
3. Browse to and select the dreamweaver. tld file from the distribution.
4. Click OK twice, and you should see the new tag library in the list.

This XSLT method can also be integrated with the Apache Cocoon XML publishing engine, and this is the environment under which it was originally developed. Apache Cocoon treats publishing a document to the Web as a pipeline; at one end XML is sourced, and at the other XML is serialized into a byte stream. This article has detailed a method that allows us to have both XHTML template documents and native or custom Cocoon data sources.

Other XML-enabled IDEs and production environments should be similarly capable of integrating this solution.

Looking Forward

In this article I have described a method to offload work from the page designer onto the back-end developers. We used standard XSLT to simplify a designer's access to live data.

The additional layer of separation can be used to perform calculations outside of the page design and data model, to isolate the designer from data changes, and to allow for unit testing.

Although we focused on creating new designer-oriented tag libraries, the core data-merging techniques are applicable to processing any XML document, and even any tag. One particularly cool application I wrote using these techniques deals with manipulating an SVG drawing at publish time based on a dynamic XML data feed. I'm sure many of you will be able to blow that away with your own ideas!

Feel free to e-mail to me with your comments and observations. A full implementation of this technique and supporting files for ANT, JAXB, Eclipse, Dreamweaver, and Apache Cocoon is available for download at http://xml.grumpykitty.biz.

Reference

- *XSLT Quick Reference by Mulberry Technologies:* www.mulberrytech.com.

PETER@GRUMPYKITTY.BIZ

LISTING 1

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
      xml:lang="en" lang="en" >
<head><title>My PHP driven page</title></head>
   <?php   // load xml data from somewhere
$product = DomNode_first_child($xmldoc);
$productName = DomElement_getAttribute($product, "name");
$productPrice = DomElement_getAttribute($product, "price");
   ?>
   <body>
      <p>
      Product Name <?php print $productName ?>, price is
<?php print $productPrice ?>.
      </p>
   </body>
</html>
```

LISTING 2

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html   xmlns="http://www.w3.org/1999/xhtml"
        xml:lang="en"
        lang="en">
<%@ taglib uri="http://java.sun.com/jsp/jstl/xml" prefix="x"
%>
<head><title>My JSTL driven page</title></head>
   <body>
      <p>
      Product Name <x:out select="$productXml/name" >
       {product}</x:out>, price is <x:out select="
       $productXml/price" />.
      </p>
   </body>
</html>
```

LISTING 3

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<htmlxmlns="http://www.w3.org/1999/xhtml"
xmlns:product="http://.../d/product/1.0"
xml:lang="en" lang="en" >
   <head><title>My XSLT driven page</title></head>
   <body>
      <p>
      Product Name <product:name>{product}</ product:name>,
price is <product:price />.
      </p>
   </body>
</html>
```

Download the Code
www.sys-con.com/xml

WRITTEN BY **DAVID S. LINTHICUM**

Semantic Mapping, Ontologies, and XML Standards

The key to managing complexity in application integration projects

When dealing with application integration, as you know by now, we are dealing with much complexity. The notion of ontologies helps the application integration architect prepare generalizations that make the problem domain more understandable. In contrast to abstraction, generalization ignores many of the details and ends up with general ideas. Therefore, when generalizing, we start with a collection of types and analyze commonalities to generalize them.

Clearly, semantic heterogeneity and divergence hinders the notion of generalization, and as commonalities of two entities are represented in semantically different ways, the differences are more difficult to see. Thus, ontological analysis clears the ground for generalization, making the properties of the entities much more clear. Indeed, ontological analysis for application integration encourages generalization. Thus we can say, "Within an ontological framework, integration analysis naturally leads to generalization."

Considering that statement, it's also clear that application independence of ontological models makes these applications candidates for reference models. We do this by stripping the applications of the semantic divergences that were introduced to satisfy their requirements, thus creating a common application integration foundation for use as the basis for an application integration project.

Returning to the core problem we wish to solve within application integration domains, we are looking to achieve semantic interoperability between very different systems. The solution to this problem is based on our ability to leverage formal ontologies required to account for the different types of ontologies for any business reason. For example, we can have resource ontologies we leverage to define semantics used by our SAP systems, but we may also have personal ontologies defining the semantics of a user or a user group. In addition, we have the notion of shared ontologies, which are common semantics shared between any numbers of information systems.

Once we define the ontologies, we must account for the semantic mismatches that occur during translations between the various terminologies. Therefore, we have the need for mapping.

Creating maps is significant work that leverages a great deal of reuse. The use of mapping requires the "ontology engineer" to modify and reuse mapping. Such mapping necessitates a mediator system that can interpret the mappings in order to translate between the different ontologies that exist in the problem domain. It is also logical to include a library of mapping and conversion functions, as there are many standards transformations employable from mapping to mapping.

Finding the Information

One of the benefits of leveraging ontologies is the fact that no matter where the information resides, we can understand and map information relevant to the application integration scenarios. Ontologies allow you to differentiate between resources, which is especially useful when those resources have redundant data (e.g., customer information in almost all enterprises). Thus, in order to make better sense of the data and represent the data in a meaningful way, terms defined in ontologies allow the application integration architects to fully understand the meaning and context of the information. Again, this is ontology's value within application integration.

When considering schemas, local to remote source or target systems, the application of ontologies is leveraged in order to define the meaning of the terms used in some domain. Although there is often some communication between a data model and the attributes, both schema and ontologies play key roles in application integration because of the importance of both semantics and data structures.

Another important notion of ontologies is entity correspondence. Ontologies that are leveraged in more of a B2B environment must leverage data that is scattered across very different information systems, and information that resides in many separate domains. Ontologies in this scenario provide a great deal of value because we can join information together, such as product information mapped to on-time delivery history mapped to customer complaints and compliments. This establishes entity correspondence.

To gather information specific to an entity, we need to leverage different resources to identify individual entities, which vary widely from each physical information store. For example, when leveraging a relational database, entities are identified using keys (e.g., customer number). Within the various information systems, many different terms are used for attributes. The notion of ontologies, in this scenario, allows us to determine whether entities from different applications and databases are the same or noncrucial to fusing information.

Ontology and Mapping Servers

So, how do you implement ontologies in your application integration problem domain? In essence, some

AUTHOR BIO

David Linthicum is the CTO at Grand Central Communications (www.grandcentral.com), and a leading expert in the application integration and open standards areas. David is the author of 8 books, including his latest: Next Generation Application Integration.

technology – either an integration broker or applications server, for instance – needs to act as an ontology server and/or mapping server.

An ontology server houses the ontologies that are created to service the application integration problem domain. There are three types of ontologies stored: shared, resource, and application. Shared ontologies are made up of definitions of general terms that are common across and between enterprises. Resource ontologies are made up of definitions of terms used by a specific resource. Application ontologies are native to particular applications, such as an inventory application. Mapping servers store the mappings between ontologies (stored in the ontology server). The mapping server also stores conversion functions, which account for the differences between schemas native to remote source and target systems. Mappings are specified using a declarative syntax that provides reuse.

RDF and Ontologies

Resource Description Framework (RDF), a part of the XML story, provides interoperability between applications that exchange information. RDF is another Web standard that's finding use everywhere, including application integration. RDF was developed by the W3C to provide a foundation of metadata interoperability across different resource description communities and is the basis for the W3C movement to ontologies, for example, the use of Web Ontology Language (OWL).

RDF uses XML to define a foundation for processing metadata and to provide a standard metadata infrastructure for both the Web and the enterprise. The difference between the two is that XML is used to transport data using a common format, while RDF is layered on top of XML defining a broad category of data. When the XML data is declared to be of the RDF format, applications are then able to understand the data without understanding who sent it.

RDF extends the XML model and syntax to be specified for describing either resources or a collection of information. (XML points to a resource in order to scope and uniquely identify a set of properties known as the schema.)

RDF metadata can be applied to many areas, including application integration. One example would be searching for data and cataloging data and relationships. RDF is also able to support new technology (such as intelligent software agents and exchange of content rating).

RDF does not offer predefined vocabularies for authoring metadata; however, the W3C does expect standard vocabularies to emerge once the infrastructure for metadata interoperability is in place. Anyone, or any industry, can design and implement a new vocabulary. The only requirement is that all resources be included in the metadata instances using the new vocabulary.

RDF benefits application integration in that it supports the concept of a common metadata layer that is sharable throughout an enterprise or between enterprises. Thus, RDF can be used as a common mechanism for describing data within the application integration problem domain.

Web-Based Standards and Ontologies

The use of languages for ontology is beginning to appear, built on reasoning techniques that provide for the development of special-purpose reasoning services. In fact, the W3C is creating a Web standard for ontology language as part of its effort to define semantic standards for the Web. The Semantic Web is the abstract representation of data on the World Wide Web, based on the Resource Description Framework standards (see the "RDF and Ontologies" tidbit) and other standards still to be defined. It is being developed by the W3C, in collaboration with a large number of researchers and industrial partners.

In order for the Semantic Web to function, computers must have access to structured collections of information and sets of inference rules that they can use to conduct automated reasoning. This notion is known as knowledge representation. To this end, and in the domain of the World Wide Web, computers will find the meaning of semantic data by following hyperlinks to definitions of key terms and rules for logically reasoning about data. The resulting infrastructure will spur the development of automated Web services such as highly functional agents. What's important here is that the work now being driven by the W3C as a way to manage semantics on the Web is applicable, at least at the component level, to the world of application integration, much like XML and Web services.

An example of the W3C contribution to the use of ontologies is the Web Ontology Language. OWL is a semantic markup language for publishing and sharing ontologies on the World Wide Web. OWL is derived from the DAML+OIL Web Ontology Language and builds upon the RDF. OWL assigns a specific meaning to certain RDF triples. The future Formal Specification, now in development at the W3C, specifies exactly which triples are assigned a specific meaning and offers a definition of the meaning. OWL only provides a semantic interpretation for those parts of an RDF graph that instantiate the schema. Any additional RDF statements resulting in additional RDF triples are allowed, but OWL is silent on the semantic consequences of such additional triples. An OWL ontology is made up of several components, some of which are optional, and some of which may be repeated.

Using these Web-based standards as the jumping-off point for ontology and application integration, it's possible to define and automate the use of ontologies in both intra- and intercompany application integration domains. Domains made up of thousands of systems, all with their own semantic meanings, bound together in a common ontology that makes short work of application integration and defines a common semantic meaning of data – this, indeed, is the goal.

Extending from the languages, we have several libraries available for a variety of vertical domains, including financial services and e-business. We also have many knowledge editors that now exist to support the creation of ontologies, as well as the use of natural-language processing methodologies. We have seen these in commercially available knowledge mapping and visualization tools using standard notations such as UML.

Value of Ontologies and Semantic Mapping

The use of the ontologies concept within modern application integration techniques and technologies seems to be a good match. Indeed, today we are already leveraging certain aspects of ontologies within most application integration projects, whether we understand the concept or not. The value here is to recognize ontologies as a concept that formalizes the management and integration of information, services, and processes … formalizing something we are already doing informally.

The real significance of ontologies leveraging the reusable aspects is within vertical domains where the use of common metadata, services, and processes has the most value. Once we get semantics under control within vertical systems (more often, a collection of systems), application integration or linking a common set of semantics to back-end systems won't be as daunting as this process is today. What's more, the application of standards such as Semantic Web and OWL will make ontologies that much more attractive.

DLINTHICUM@GRANDCENTRAL.COM

Dralasoft Debuts Workflow 3.0 for Web Services

(Westminster, CO) – Dralasoft, Inc., a provider of Java technology for e-business and enterprise application infrastructure, has released Workflow 3.0, the newest version of its business process management (BPM) software. Version 3.0 includes an extensive new SOAP interface that can be used to invoke a workflow engine as a Web service, enabling business partners to seamlessly collaborate in mission-critical workflows via the Web.

Workflow 3.0 capitalizes on its SOA capability by enabling business units and/or trading partners to integrate common workflow technology into any major production situation. Document management, task management, claims processing, e-commerce, and supply chain management are some of the ways Workflow 3.0 saves time, manpower, and cost in distributed environments. www.dralasoft.com

Actional Introduces "Watchdogs" for End-to-End Availability and Performance

(Las Vegas, NV) – Actional Corporation, a provider of Web services management, has announced the availability of Web services "watchdogs," which provide advanced capabilities for monitoring end-to-end availability and performance of Web services. A new feature, unique to Actional Looking Glass, Actional watchdogs enable organizations to identify and prevent potential service failures by automatically and proactively checking service availability and performance. Actional watchdogs interoperate with Actional's policy and service stabilizer capabilities, helping IT operators to ensure the ready state of the Web services network, discover problems, and trigger workarounds to avoid business disruption. www.actional.com

Parasoft SOAPtest Supports WS-I Testing Tools 1.0

(Monrovia, CA) – Parasoft, a provider of Automated Error Prevention software solutions, has announced that Parasoft SOAPtest 2.5 has extended its testing capabilities to include Testing Tools 1.0, designed by the Web Services Interoperability Organization (WS-I), to check conformance to Basic Profile 1.0. SOAPtest 2.5 verifies the Web Services Description Language (WSDL) and Simple Object Access Protocol (SOAP) traffic for conformance to Basic Profile 1.0 using Testing Tools 1.0 developed by WS-I. Parasoft contributed to writing the Testing Tools as a member of the WS-I Testing Tools Working Group.

SOAPtest enhances the usability of the WS-I Testing Tools by eliminating the need to set up man-in-the-middle monitor and configuration files for the Analyzer. By functioning as both the monitor and the analyzer, it is able to verify both WSDLs and SOAP traffic for conformance to Basic Profile 1.0 without any configuration files. The only required input to SOAPtest is the WSDL URL. SOAPtest produces WS-I conformance reports that can be viewed directly in a Web browser.
www.parasoft.com

Sarvega Announces Guardian Family of XML Security Appliances

(Chicago) – Sarvega, Inc., a provider of high-performance XML networking solutions, has announced the availability of its Guardian Gateway and the Guardian Accelerator for XML security applications. Guardian Gateway is a flexible XML security gateway with full support for the OASIS WSS Core and WS-Security specifications, a comprehensive authentication, authorization, and auditing framework, with SAML, XML Digital Signatures, and XML Encryption support. It integrates seamlessly into the existing security and management infrastructure.

Sarvega's Guardian solution addresses critical requirements for private key security with hardware key storage and the FIPS 140-2 validated module for cryptographic operations. FIPS is a product standard for security recognized by both governments and private firms.
www.sarvega.com

Grand Central Communications and DataPower Team to Provide End-to-End Privacy for Web Service Deployments

(Cambridge, MA) – Grand Central Communications, a provider of integration on demand through its Business Services Network; and DataPower Technology, Inc., a provider of intelligent XML-Aware Network (XAN) infrastructure, have announced a partnership and marketing agreement that brings privacy and security to enterprise business transactions through Grand Central's Business Services Network utilizing the DataPower XS40 XML Security Gateway.

DataPower offers a family of XML-Aware Networking devices with its XS40 XML Security Gateway and XA35 XML Accelerator powered by DataPower's patent-pending XG3. These "drop-in" network devices enable comprehensive message-level security, centralized security, and hardware performance acceleration for any XML or Web service–based application. Grand Central Communications' Business Services Network plugs directly into an enterprise's IT infrastructure. The Network transparently mediates and brokers between disparate partner applications and network architectures to facilitate change and enable enterprises to quickly provide extensible and differentiated services to their customers and partners. By utilizing Grand Central in conjunction with DataPower's network devices, enterprises can trust in a shared and transparent network infrastructure for their Business Services Transactions that is secure and easy to implement, while benefiting from end-to-end transaction privacy.
www.grandcentral.com, www.datapower.com

Korean Consortium Selects Infravio for B2B/M2M Web Services Integration

(Cupertino, CA) – BuiltOne Ltd has chosen Infravio Ensemble to manage Web services for the business-to-business (B2B) and marketplace-to-marketplace (M2M) integration of a paint industry consortium project in Korea. The project is scheduled to go live in July of 2004.

While this project delivers a B2B infrastructure for the Korean paint industry, other marketplaces will be addressed later for a rapid evolution to a true M2M system. The Korean government has selected 11 industries to participate in the B2B Standardization Project sponsored by the Ministry of Commerce, Industry, and Energy. BuiltOne Ltd was chartered with the integration project to provide centralized control over all Web services.

Making every Web service available to all parties in a controlled manner was the key challenge. Although it is easy for any party to invoke a Web service, it is difficult to control and track all Web service invocations in a dynamic and constantly changing environment. Infravio Ensemble provides the control and management that BuiltOne needs to define, configure, track, and manage all Web services used throughout the consortium, as well as to access marketplaces beyond. www.infravio.com

WebAppCabaret™

J2EE Web Hosting

http://www.webappcabaret.com/jdj.jsp

1.866.256.7973

Quality Web Hosting at a reasonable price...

<JAVA WEB HOSTING AND OUTSOURCING>

You have developed the coolest mission-critical application. Now you need to deploy it.
Outsource your hosting and infrastructure requirements with WebAppCabaret so you can save time and money and concentrate on other important things. WebAppCabaret is the leading JAVA J2EE Web Hosting Service Provider. From shared hosting to complex multi-dedicated server hosting, WebAppCabaret has the right solution for you. 30 Day Money Back Guarantee and SLA.WebAppCabaret offers the latest Standards based Servlet containers, EJB servers, and JVMs. We provide options such as e-Commerce, EJB 2.x, Failover, and Clustering. Our Tier 1 Data Center ensures the best in availability and performance.

At WebAppCabaret you have the flexibility to choose the LATEST hosting technology best suited for your WEB application or your programming skill. If you are a programmer or consultant,WebAppCabaret has the right hosting solution for your project or client's dynamic web application/services requirements.

OUTSOURCING:
Do you really need an IT department for your web applications, mail systems, and data backups when we can perform the same functions more efficiently at a fraction of the cost - with competent technical expertise and redundant hosting facilities.

J2EE HOSTING:
Below is a partial price list of our standard hosting plans. (Reseller accounts also available). For more details please log on to http://www.webappcabaret.com/jdj.jsp

$39/mo Enterprise

Latest JBoss/Tomcat/Jetty
Latest JSP/Servlets/EJBs
Private JVM
Choice of latest JDKs
Dedicated IP Address
NGASI Control Panel
PHP and Perl
Web Stats
1GB Disk
200MB DB
MySql
PostgreSql
Dedicated Apache
Telnet . SSH . FTP
5 Domains
100 Emails
Web Mail . POP . IMAP
more...

$17/mo Professional

Latest Tomcat/Jetty
Latest JSP/Servlets
Private JVM
Choice of latest JDKs
NGASI Control Panel
PHP and Perl
Web Stats
200MB Disk
30MB DB
MySql
Telnet . SSH . FTP
2 Domains
20 Emails
Web Mail . POP . IMAP
more...

$191/mo Dedicated

Managed Dedicated
Server starts at $191
per month for:
256MB RAM
Pentium 4
40GB RAID
Firewall
Unlimited Domains
Unlimited Web Site Hosting
NGASI Control Panel
with your own Logo
Each dedicated
server configured
for standalone
web application
and web hosting (resellers)
at no extra charge.
more...

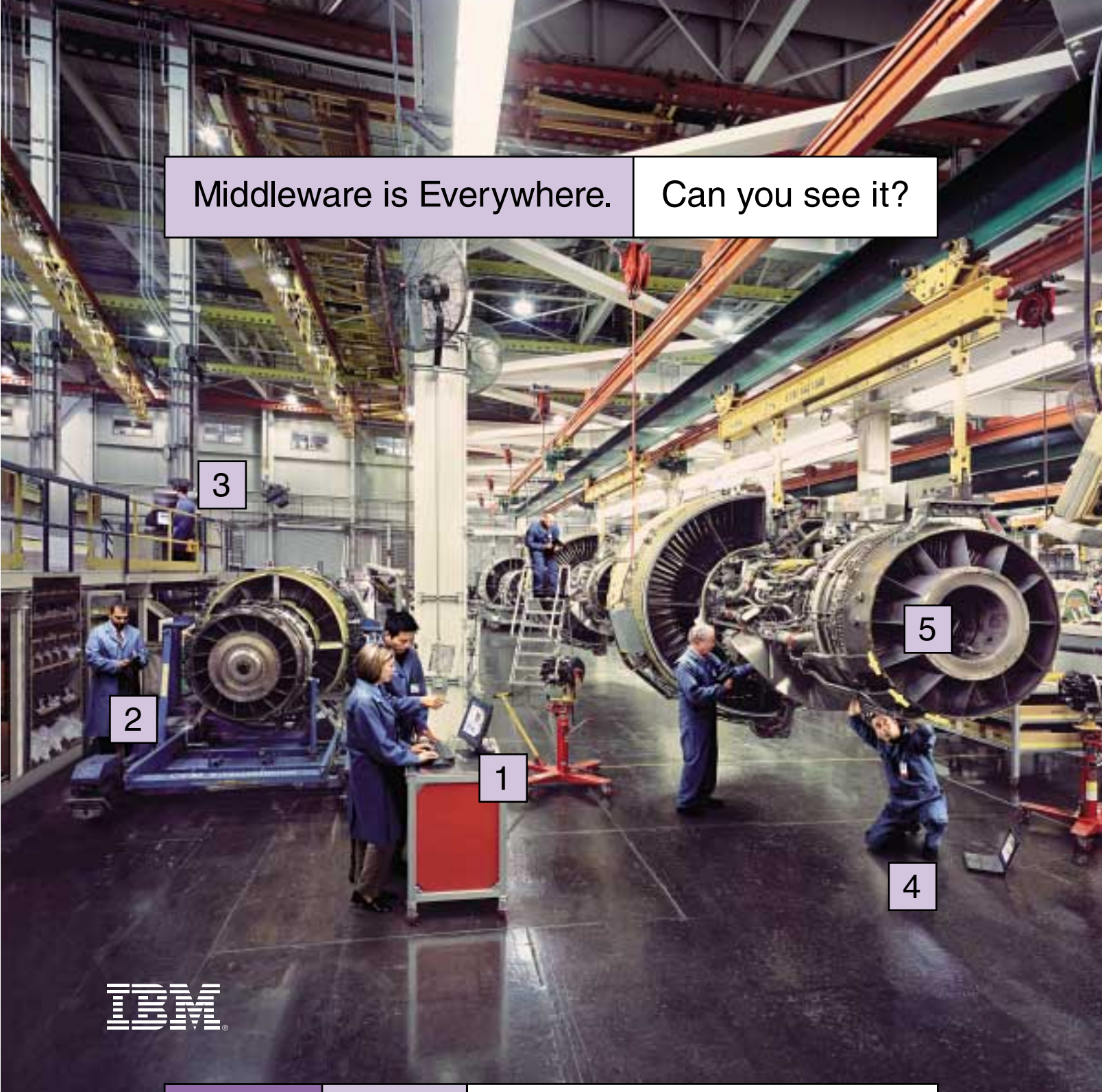
$99/mo Reseller

If you cannot afford a
Dedicated server for
reselling web hosting,
for $99/mo the Shared
Reseller plan gives you:

5 Professional Plans
10 Basic Plans
NGASI Control Panel
with your own Logo
50% Discount
No System Admin
Easy Account Mgt

more...

Prices, Plans, and Terms subject to change without notice. Please log on to our website for the latest Prices and Terms.



IBM

WebSphere.

Key

1. **New design already tested.**
2. **Suppliers already linked.**
3. **Procurement already automated.**
4. **Blueprints already updated.**
5. **Engine all ready for takeoff.**

MIDDLEWARE IS IBM SOFTWARE. Powerful software like IBM WebSphere.® Using an open and scalable foundation, WebSphere software lets you swiftly respond to change. On demand. Applications are easily updated, tested and deployed. Lead time is shortened. And everything clicks, regardless of platform. WebSphere delivers it all. On the money. On demand.

e **business on demand**™ at **ibm.com**/websphere/middleware