

XML JOURNAL

THE ULTIMATE XML ENTERPRISE RESOURCE

December 2003 Volume: 4 Issue: 12

XML-JOURNAL.COM

GUEST EDITORIAL

The Key to Success with Web Services

Look to intelligent network products for help

by Jeff Browning pg. 5

COMMENTARY

The Future of XML

Despite challenges, the core XML vision provides a point of unity

by Michael Champion pg. 7

PERFORMANCE

XML Acceleration: The Truth Behind the Myths

Don't assume that bandwidth and processing will be problems

by Dan Foody pg. 14

STANDARDS

Finding the Fit for XSLT

Filling a hole in the puzzle

by David S. Linthicum pg. 40

pg. 32

edge 2004
EAST

DEVELOPMENT TECHNOLOGIES EXCHANGE

February 24-26, 2004

Hynes Convention Center, Boston, MA

ARCHITECTING JAVA, .NET, WEB SERVICES, MX, XML, AND OPEN SOURCE

REGISTER BY DECEMBER 19, 2003

SAVE UP TO \$400

DISPLAY UNTIL February 29, 2004

\$6.99US \$7.99CAN



SYS-CON
MEDIA

CONTENT MANAGEMENT?

Dynamic? Static? The answer may surprise you 24

Malicious Attack Protection for XML Web Services

Communication behind a firewall isn't always safe 8



Andrew Yang

Feature: Open Integration and Security

The good news about XML and Web services...and the bad 10



John Lilly

Security Challenges Inside the Firewall

Outside hackers get all the attention, but what about internal threats? 16



K. Scott Morrison

E-Government: What's Your Government Doing with XML?

It's more than you might think 20



Bryan Baker

Designing an Open, Standards-Based Reporting System

XML meets the challenges and design goals 22



Joe Marini

Feature: Building a High-Traffic Web Site with Static Delivery

Dynamic and static sites: the best of both worlds 24



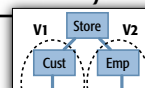
Todd Price

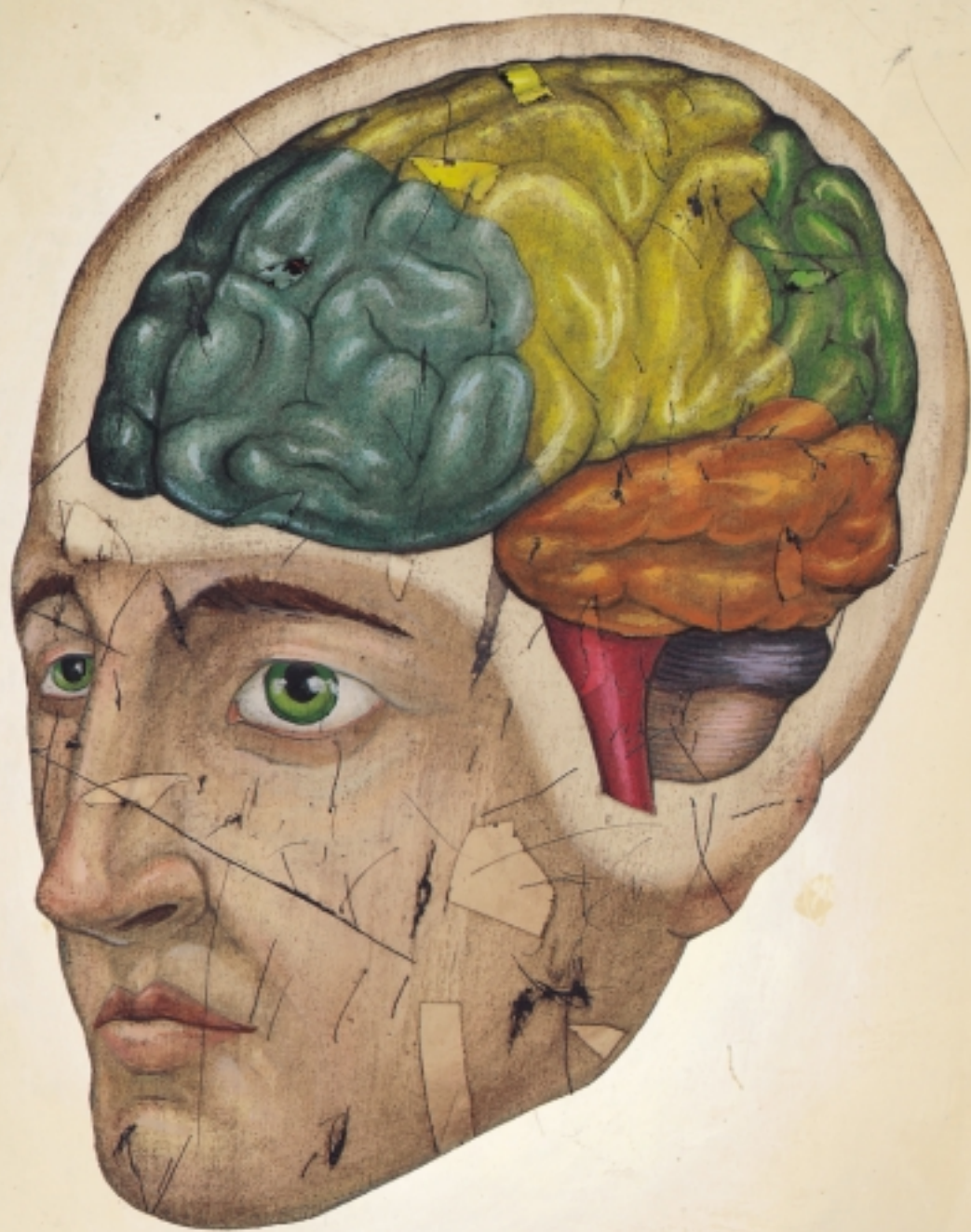
Advanced ANSI SQL Native XML Integration Part 2

Supporting advanced XML capabilities 28



Michael M David





DO YOU HAVE A BRAIN?

CEREBRUM: CONTROLS THOUGHT.

*What is 2+2? What color is the sky?
If you can answer these, then you have a cerebrum.*

MEDULLA OBLONGATA: CONTROLS INVOLUNTARY FUNCTIONS.

*Check your pulse. Do you have one?
Then you have a medulla oblongata, too.*

BROCA'S AREA: CONTROLS LANGUAGE.

*Say the following sentence: "Frankly, Hector, I'm a bit surprised."
Did it work? Then you have a Broca's Area.*

PITUITARY GLAND: CONTROLS HORMONES.

*When you were about 13, did your voice change?
Did you grow hair in special places?
Then you've got a pituitary gland, friend.*

YOU HAVE A BRAIN.

So why spend so much time on mindless coding?

Download Weblogic Workshop Now.

BEA WebLogic Workshop™ 8.1 is a more efficient way to develop in J2EE.
And that means less grunt work and more real work. dev2dev.bea.com/thought.



Introducing

SOAPscope 3.0

Debug

Test

Learn

4 Ways to Know Web Services

Whether you are learning how a Web service works, or troubleshooting a tough problem, you need the help of a "smart" tool. SOAPscope leverages your intelligence and satisfies your curiosity by digging deeper, faster.

Try It Find patterns to solve problems by testing your Web service with different inputs without writing any code.

See It Look under the covers at WSDL and SOAP to understanding what's happening. Capture from any toolkit, and see just the right detail for the task at hand.

Diff It Compare a problem message or WSDL with a similar, working one.

Check It When the problem's not obvious, rigorous interactive analysis finds inconsistencies, errors, and interoperability problems.

Look What's New in 3.0

- Interactive Message Analysis
- Interoperability Testing System
- SSL Support
- HTTP Authentication Support
- HTTP Compression Support
- Visual Studio & .NET Integration
- Support for multi-byte encoding
- Best usability of any web services tool

The most comprehensive Web services diagnostic system available, and still **only \$99!**

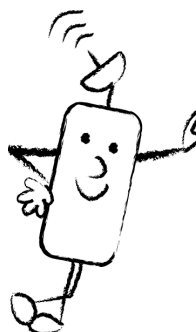
Try It...at XMETHODS

Mindreef has partnered with XMethods to allow you to easily invoke hundreds of different web services without writing code. Just click on 'Try It', next to any service name at xmethods.net.

"Comparing SOAPscope and other SOAP sniffing tools is like the difference between shooting a bullet and throwing it."

Scott Hanselman
Chief Architect
Corillian Corp.

Mr. SOAPscope says -
"Try SOAPscope free at
www.mindreef.com!"



© Copyright 2003, Mindreef, Inc. The names of companies and products mentioned herein may be the trademarks of their respective owners. This product uses Hypersonic SQL. This product includes software developed by the Politecnico di Torino and its contributors.

"The Web Services Diagnostic Experts" Mindreef

FOUNDING EDITOR

Ajit Sagar ajit@sys-con.com

EDITORIAL ADVISORY BOARD

Graham Glass graham@themindelectric.com

Coco Jaenicke cjaenicke@attbi.com

Sean McGrath sean.mcgrath@propylon.com

Simeon Simeonov talktosim@polarisventures.com

EDITORIAL

Editor-in-Chief

Hitesh Seth hitesh@sys-con.com

Managing Editor

Jennifer Van Winckel jennifer@sys-con.com

Editor

Nancy Valentine nancy@sys-con.com

Associate Editors

John Evdemon jevdemon@sys-con.com

Jamie Matusow jamie@sys-con.com

Gail Schultz gail@sys-con.com

Jean Cassidy jean@sys-con.com

Assistant Editor

Kelly Flynn kelly@yachtchartersmagazine.com

PRODUCTION

Production Consultant

Jim Morgan jim@sys-con.com

Art Director

Alex Botero alex@sys-con.com

Associate Art Directors

Louis F. Cuffari louis@sys-con.com

Richard Silverberg richards@sys-con.com

Assistant Art Director

Tami Beatty tami@sys-con.com

CONTRIBUTORS TO THIS ISSUE

Bryan Baker, Jeff Browning, Michael Champion,

Michael M David, Dan Foody, John Lilly,

David S. Linthicum, Joe Marini, K. Scott Morrison,

Todd Price, Andrew Yang

EDITORIAL OFFICES

SYS-CON MEDIA

135 CHESTNUT RIDGE ROAD, MONTVALE, NJ 07645

TELEPHONE: 201 802-3000 FAX: 201 782-9637

XML-JOURNAL (ISSN# 1534-9780)

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:

XML-JOURNAL, SYS-CON Publications, Inc.,

135 Chestnut Ridge Road, Montvale, NJ 07645.

Worldwide Newsstand Distribution

Curtis Circulation Company, New Milford, NJ

FOR LIST RENTAL INFORMATION:

Kevin Collopy: 845 731-2684,

kevin.collopy@edithroman.com

Frank Cipolla: 845 731-3832,

frank.cipolla@epostdirect.com

©COPYRIGHT

Copyright © 2003 by SYS-CON Publications, Inc. All rights reserved.

No part of this publication may be reproduced or transmitted

in any form or by any means, electronic or mechanical,

including photocopy or any information storage and retrieval

system, without written permission. For promotional reprints,

contact reprint coordinator. SYS-CON Publications, Inc.,

reserves the right to revise, republish and authorize its readers

to use the articles submitted for publication.

All brand and product names used on these pages

are trade names, service marks, or trademarks of their respective

companies. SYS-CON Publications, Inc., is not affiliated

with the companies or products covered in XML-Journal.



The Key to Success with Web Services

WRITTEN BY JEFF BROWNING



Web services provide a way to allow efficient communication between disparate services. For years, enterprises have struggled to find reliable, cost-effective ways

to integrate and automate critical processes between different application packages. Web services technology has the potential to answer an enterprise's needs, providing the ability to integrate different systems and application types regardless of platform, operating system, or location.

The key to the success of Web services is the use of a common data exchange standard such as XML. Through the use of this common language, enterprises can create reusable application components (Web services) that can be linked together to cost-effectively create distributed enterprise applications with minimal development efforts.

Vendors continue to design the framework and development environments necessary to build Web services. However, the projected adoption rate is tremendous. According to estimates from Gartner, Web services will represent the "dominant mode of deployment for new application solutions for Fortune 2000 companies" by 2004.

Because Web services could involve millions of users, the need to provide security, high availability, and reliability for these services is critical. To enable Web services to flourish and reduce the high costs and complexity associated with additional application servers, companies must look to network technologies for help. Businesses must consider highly intelligent network products that can quickly process any application or Web service, ensuring quick response times, reliable sessions, adaptive scalability, and application-level security, all through a single network device. Additionally, the products they choose must be flexible to easily handle future applications and protocols while protecting and improving the performance of their application investments. Key challenges include:

- **Increasing reliability:** The distributed nature of Web service applications demands a stable and reliable network environment and server infrastructure. With multiple components scattered across geographically dispersed networks, reliable communication and application performance become paramount to deployment success.

- **Improving quality of service:** In addition to communication reliability, organizations will

need mechanisms to prioritize requests. Requests will need to be intercepted, analyzed, and directed to the proper resource to provide quality of service granularity based upon various organizational business policies.

- **Ensuring high availability:** As the demand for Web services increases, the availability of each component within the service and the applications that process the requests will be critical. Key systems and devices that assure Web service availability and validity will be required.

- **Providing scalability:** Flexible deployment scenarios will be necessary as demand for Web services increases. Organizations will be required to act quickly to add resources to support Web service requests without interruption.

- **Enhancing performance:** The quick adoption rate and ease of deployment of Web services will place increasingly large demands on network infrastructures. Traffic generated by Web services can be significant. For example, a user request for a stock quote might initiate as many as eight related services to perform functions to serve that user's request. In total the entire transaction could require thirty to forty related requests.

- **Increasing application security:** The need to secure applications without sacrificing performance is extremely important. Enterprises must offload intensive SSL processing from application servers, allowing them to handle the explosion of performance demands required in a Web services environment.

- **Increasing network security:** When designing their Web services infrastructure, organizations are challenged with finding traditional network devices and tools that increase reliability and provide an extra layer of security. Network devices need flexible, comprehensive, and secure feature sets to increase control over network traffic and protect the organization from existing and future attacks.

Enterprises should look for products that provide enterprises the ability to switch, persist, and filter any type of Web service based upon content encapsulated in the header or payload of a packet. The resulting benefits of this capability are extremely significant, allowing businesses to support the complex security and high availability requirements of today's Web services – making them simpler to implement and maintain while dramatically increasing operational efficiencies and cost savings.

J.BROWNING@F5.COM

Ektron Success Story #1531

Enterprise Web Content Management without the enterprise integration.



Let us show you how an Ektron XML Web Content Management Solution can enhance your Web site.

Learn More at:

www.ektron.com/xmlj

A leading baking products company needed to build a website that reflected brand and offered consumers access to up-to-date product information, recipes and promotional programs. With Ektron's Web content management solution they were able to achieve a high degree of site functionality and automation including the use of XML and Web Services to syndicate content to internal and third party web sites.

Ektron - Redefining Web Content Management

PRESIDENT and CEO

Fuat Kircaali fuat@sys-con.com

VP, Business Development

Grisha Davida grisha@sys-con.com

Group Publisher

Jeremy Geelan jeremy@sys-con.com

Technical Director

Alan Williamson alan@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 of Sales & Marketing

Megan Mussa megan@sys-con.com

Advertising Sales Manager

Alisa Catalano alisa@sys-con.com

Associate Sales Managers

Carrie Gebert carrieg@sys-con.com

Kristin Kuhnle kristin@sys-con.com

Beth Jones beth@sys-con.com

SYS-CON EVENTS

President

Grisha Davida grisha@sys-con.com

Conference Manager

Michael Lynch mike@sys-con.com

National Sales Manager

Sean Raman raman@sys-con.com

CUSTOMER RELATIONS

Circulation Service Coordinators

Niki Panagopoulos niki@sys-con.com

Shelia Dickerson shelia@sys-con.com

Edna Earle Russell edna@sys-con.com

JDJ STORE

Manager

Brunilda Staropli bruni@sys-con.com

WEB SERVICES

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

Accounts Receivable

Charlotte Lopez charlotte@sys-con.com

Financial Analyst

Joan LaRose joan@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)

Back issues: \$12 U.S. \$15 all others

The Future of XML

WRITTEN BY **MICHAEL CHAMPION**



To look backward for a while is to refresh the eye, to restore it, and to render it the more fit for its prime function of looking forward.

— Margaret Fairless Barber

The end of the year is here again, a time when we traditionally take a long look at the progress we've already made and then turn our eyes toward the future, attempting to forecast the year to come. With this in mind, Hitesh Seth sought input from industry leaders on their end-of-year predictions, and caught up with Michael Champion, research and development specialist at Software AG.

Michael has been active in the World Wide Consortium's Document Object Model (DOM) Working Group for more than three years and was an editor of the core XML portion of the DOM Level 1 Recommendation. He is now co-chair of the Web Services Architecture Working Group. Michael's current focus is on technical business development activities, writing articles on XML technology, and building example integrations between XML applications and Software AG's database and enterprise integration products.

Here, Michael shares his thoughts on the current state of XML and his expectations for the future.

Current Realities

XML is becoming pervasive, the "safe" choice for documentation, data interchange, application integration, business messages, and other areas. Everyone in the IT industry has an XML story, and to a very great extent these stories are based on XML standards that make interoperability much easier than it was in the past.

While some lament the complexity and diversity of the overall body of specifications from the W3C, OASIS, and other organizations, the real power of XML comes from its most basic and universal properties: the ability to represent any kind of data, with labels ("tags") associated with each value to make it easier to work with, in convenient packages that conveniently group together related information, leveraging the Unicode standard to support all computing platforms and human languages.

A Look Ahead

Consumers will vote with their feet against the complexity around the edges of XML just as they vote for the simplicity at its core. Products

such as Microsoft Office 2003, which support XML only via the complex and widely criticized W3C XML Schema specification, will fail to meet their full potential until simpler approaches to document structure definition are supported by "aftermarket" products or a subsequent edition of the core product.

XML itself will become increasingly taken for granted as part of the "plumbing" rather than something visible to end users. This implies that XML infrastructure products and vendors will be chosen on the basis of their support for real standards, reliability, performance, and so on, and not on the basis of marketing and mind share.

Because XML is such an important part of the IT infrastructure, increasing attention will be paid to the obstacles that the current standards offer to those who deploy XML in environments where data communication is slow and XML's verbosity is a limitation, and in environments where processing speed is critical and XML's human-oriented grammar and features just slow things down. It's not clear whether these challenges will be overcome by improving the processors or improving the specifications, and there will be much experimentation in both directions.

The XML standards will be "refactored" (as the term is used in the Extreme Programming community) to support high-performance processing and to put what is really universally useful in the core, and relegate the complex and specialized features to an optional periphery.

All this will cause a certain amount of fragmentation: the data-oriented specifications such as W3C XML Schema will not be widely supported by those focusing on producing human-oriented documents, and the features of XML that appeal to human authors but pose great processing burdens (such as general entities and references) will not be widely supported in data-oriented applications. Alternative serializations of the XML data model that are more compact and/or faster to process will find support in specialized areas where these considerations are critical.

In spite of the challenges and fragmentation we'll see, however, the core XML vision and standards will provide a point of unity across all the diverse communities that develop specialized tools and formats to meet their own needs. ☼

AUTHOR BIO

Michael Champion is a research and development specialist with Software AG.

MICHAEL.CHAMPION@SOFTWAREAGUSA.COM



Malicious Attack Protection for XML Web Services

Communication behind a firewall isn't always safe

As the vast majority of Global 2000 organizations are transitioning pilot projects into production, the hype over XML Web services might finally be turning into reality. Web services usage has varied from simple data information sharing between two applications to corporate-wide, service-oriented architecture (SOA) initiatives.

There are many reasons why Web services are gaining traction, including ease-of-use, their loosely coupled nature, and effectiveness in application integration with the lowest cost and least effort required. Yet the use of Web services is not without challenges. Security continues to be a top concern, and with good reason, as Web services interfaces are different than Web site pages and cannot rely on the same mechanisms to protect them.

Security as Top Priority

Security has taken a priority role in organizations as the threat of breached assets and downtime is compounded with regulatory requirements, including HIPAA and Gramm-Leach Bliley. Chief security officers (CSO) are now commonplace and are charged with maintaining not only digital integrity, but physical security as well. As these C-level executives are pulled in two opposite directions, it is up to internal developers, QA staff, internal operations, and compliance officers to be aware of their company's security and auditing requirements.

With regards to XML security, Web services interfaces are generally much more complex, expose more functionality, and have a more sophisticated level of interactions than Web site pages. Though network firewalls do a good job of stopping network-level attacks, they

do not provide the granularity or the proper rule sets to handle complex application attacks. Implementation of Secure Sockets Layer (SSL) and other existing technologies is useful but hinders value in scalability and interoperability costs.

The majority of early Web services projects have been conducted internally, behind the firewall, yet a common misconception is that communication hidden behind a firewall is safe from both internal and external attackers. While external attacks are better known, internal attacks can be just as costly, if not more so, than external attacks. Though it is unlikely that an internal employee will mount a distributed Denial of Service (DOS) attack to bring down systems, an employee with inside knowledge may break privacy rules or perform illegal transactions.

Security represents only a subset of problems related to the overall management of Web services. When breaking up the typical application paradigm into an SOA with loosely coupled endpoint applications, security, monitoring, interoperability, and reliability become important concerns. While security continues to be a prevalent and visible concern to end users, these other problem areas should be addressed, preferably by the same solution.

Malicious Attack Scenarios

This article will focus on security risks associated with malicious attacks on XML Web services. There are plenty of other security problems not associated with intentional attacks, including access control where security may be inadvertently breached. Many of the protections recommended will solve both problems.

Denial of Service

Protecting Web site technologies from standard DOS activities is well understood. Web service interfaces are much more heterogeneous and require more knowledge and granularity to protect. For instance, a Web service that provides simple information may be able to comfortably handle 1,000 requests per second; however, a loan approval application may only be able to handle at most 5 requests per second. Sending a loan approval interface 10 requests per second may constitute a DOS attack but be undetectable by normal means, such as firewalls. Collecting and understanding profile data on each service provides the necessary information to identify such attacks. Other forms of DOS can be achieved by sending overly large messages that clog up the system being attacked. Detecting and blocking invalid message sizes per interface can help prevent system downtime.

Replay attack

Similar to DOS, Replay attacks involve copying valid messages and repeatedly sending them to a service. Techniques similar to those for detecting and handling DOS can be applied to Replay attacks. In some ways, Replay attacks are easier to detect with Web services because payload information is more readily available. With the right tools, patterns can be detected more easily even if the same or similar payload is being sent across multiple mediums like HTTP, HTTPS, SMTP, or across different interfaces.

Message of application buffer overflow

With XML Web services, information about data parameters is exposed. In addition, much more data is likely to be

AUTHOR BIO

Andrew Yang is the senior director of marketing at Westbridge Technology, which provides Web services management (WSM) solutions that enable enterprises to secure and manage XML Web services networks.

sent between systems, creating the opportunity for Buffer Overflow attacks. For example, an attacker can send a parameter that is longer than the program can handle, causing the service to crash or the system to execute undesired code supplied by the attacker. A typical method of attack is to send an overly long request, for instance, a password with many more characters than expected. Many legacy systems that will be Web service enabled are designed for controlled, well-behaving requests and may not be prepared to handle unusual requests.

Similar to Buffer Overflow attacks, hackers often send malformed content to produce a similar effect. Sending in strings such as quotes, open parentheses, and wild cards can often confuse a Web service interface.

Dictionary attack

Many systems have weak password protection, and Web service interfaces are no different. However, unlike portals, XML Web service interfaces are heterogeneous in nature, with each system having its own authentication system and methods for deterring undesired behavior. Dictionary attacks, where a hacker may either manually or programmatically attempt common passwords to gain entry into a system or multiple systems, are common. Administrators should ensure that passwords are difficult to guess and are changed often. Unlike standard user credentials, application credentials are determined by the administrator. Password-strengthening rules that are desirable for users should also apply to administrators of Web service interfaces. Some solutions offer a "lockout" feature once a certain number of failed attempts have been reached.

SQL Injection

SQL Injection attacks involve adding special characters or terms to cause SQL statements to return unintended data. For example, strings that may end up in a WHERE clause of a SQL statement may be tricked into including more information. For instance, a parameter value of:

X' OR 1=1

may cause the whole table to be returned because 1=1 is always TRUE. Other types of SQL Injection attacks can enable an attacker to execute any SQL command.

Virus or malicious payload

One challenge with encryption is virus detection. Web service traffic may

include attachments. When content is encrypted, viruses that may be a part of the message are also encrypted. This makes it difficult for a virus-checking program to detect malware. When using encrypted data, virus checking can be performed at the destination or by an intermediary that can decrypt the data to be virus scanned and reencrypted for transport.

These represent just a few well-known attacks common with XML Web services interfaces. Other attacks can involve "hijacking" the WSDL file or taking advantage of some XML-specific characteristics such as external entity attacks. Hackers, of course, are highly creative and come up with new ways of attacking so solutions must continue to innovate to keep pace.

Top 10 Requirements in Securing Web Services

While many current pilot projects have only a handful of security requirements, ensuring that these networks are secure as they grow in usage is essential to creating a cost-effective and secure environment down the road. There are a number of requirements that are essential to build into the system early in the development process to ensure that security is an enabler of secure communications rather than a hindrance to the project.

1. Authentication.
2. Authorization and access control.
3. Encryption via SSL, XML encryption, or dedicated lines.
4. Signature support, XML Signature.
5. Malicious attack protection for Web service interfaces, including content filtering. Consider third-party products that keep pace with new hacker innovations.
6. Automated exception-handling and threat-containment capabilities for quarantining bad messages and blacklisting attackers.
7. Auditing of every transaction and change to the system.
8. Schema validation for message well-formedness.
9. Hiding complexity and back-end internal resources through service virtualization or service views.
10. Building security in the abstraction layer versus at each endpoint. Analysts agree that building security at each endpoint results in higher costs. Developers should focus more on business logic than on application security.

Of course, existing standards such as

LDAP, PKI, and SSL play an important role in securing Web services. To handle the special needs of security for Web services, numerous additional standards are being introduced, some of which are covered here.

SAML

Security Assertion Markup Language is being developed by the W3C and is a protocol for asserting authentication and authorization information. SAML-compliant servers can be accessed for authentication and authorization data in order to enable single sign-on.

XKMS

The XML Key Management Specification is a protocol developed by the W3C that describes the distribution and registration of public keys. Web services can access an XKMS-compliant server to receive updated key information for encryption and authentication.

XML Encryption

XML Encryption is a protocol developed by the W3C that describes the encryption of digital content. The XML Encryption standard includes protocols for encrypting sections of XML documents. XML Encryption enables end-to-end encryption because the actual message is encrypted. Session-level encryption can only provide data privacy between two servers.

XML Signature

XML Signature is a protocol developed by the W3C that describes the signing of digital content. The XML Signature standard includes protocols for signing sections of XML documents. XML Signature enables capabilities such as message integrity.

WS-Security

WS-Security was originally developed by Microsoft and IBM and is now being shepherded by OASIS. WS-Security is used to provide core facilities for protecting the integrity and confidentiality of a message as well as mechanisms for associating security-related claims with the message. Currently, WS-Security describes how to attach signature, encryption, and security tokens to SOAP messages

XACML

XACML is a standard used to describe access rights policies and is being driven by OASIS. XACML represents authorization and entitlement information.

~continued on pg. 19~

Open INTEGRATION and SECURITY

WRITTEN BY
JOHN LILLY

XML firewalls provide ease of integration and security

The good news about XML and Web services is that they're easier than ever to develop and deploy – inside the firewall between internal applications, on the Internet with your customers and partners...anywhere.

The superb new development tools from companies like Microsoft, BEA, IBM, and others, combined with the simplicity and openness of the standards, are providing real dividends to enterprises today. The examples are numerous: financial services companies outsourcing employee compensation, high-tech manufacturers integrating their supply chains, and global banks integrating their trading systems with partners, just to name a few.

But the bad news about XML and Web services is this: because they're so much easier to develop and deploy and for your customers and partners to connect to, it's that much easier (1) for your customers and partners to connect to them in ways you don't like, and (2) for everyone else to connect to them in ways you really don't like. With your critical systems exposed on the Internet, there are many new ways they can be vulnerable to any number of new threats (see Figure 1).

What's exhibiting itself here is an age-old tension between ease-of-integration concerns and security concerns. In general, the easier you make it to integrate your systems, the harder it is to secure them, and vice versa.

As an example, consider e-mail. E-mail was originally designed for ultimate interoperability, and has evolved to be able to work on any system, regardless of technology. It uses plain, unencrypted text (or sometimes simple HTML) and very little in the way of security techniques. Efforts to make e-mail infrastructure more secure (like digital signatures) have largely failed because they broke the interoperability aspect of the system. As a result, e-mail has become the single most exploited entry mechanism for malicious attacks on individual and enterprise systems, with the Sobig and Blaster attacks this summer and the NIMDA and the Code Red viruses in the recent past causing significant personal and professional disruption.

At the opposite end of the spectrum, think about a technology like public key infrastructure (PKI). Designed first and foremost to provide strong cryptography for a variety of security concerns, PKI has proven extremely difficult to use in practice – in particular, in getting different systems to interoperate with each other and managing distributed certificates. It's the opposite of e-mail: very secure, but not very easy to use or maintain.

So what can you do? Integration concerns are of primary importance; and speed of integration is becoming more important every day. But privacy and security remain key. Under-

standing the specific tensions between integration and security provides some insight into solutions along four key areas of tension: verbosity, time, metadata, and standards.

Verbosity

Verbosity refers to the amount of information that systems share with each other; in particular, the amount and type of information they share with each other in error conditions. For example, consider Web site log-on where a user enters the correct username but an incorrect password. The Web site could respond in many different ways. It could simply say "Incorrect Login," which doesn't differentiate between a problem with the username or password, and conveys nothing about how to log in correctly. On the other end of the spectrum, it could respond with something like "Incorrect Password." In that case, the user gets a little information about how to make the system work (fix the password), but it is a little bit less secure because it confirms that the username is a valid one.

In the Web services version of this scenario, you'd like very verbose error messages when you're integrating two systems with WSDLs and SOAP to be able to ascertain what's happening and fine-tune the system. Once in production, however, it's best to reduce the amount of information returned to failed responses so that hackers are not inadvertently given information they could use against the service.

A good, crisp example in today's development tools can be found in Microsoft's Studio.NET. For each service, you can set a flag called "IncludeStackTrace" so that when errors occur in the response message, it will also include detailed information about why the error happened. For development, this makes a lot of sense, but for actual deployment, security is best served by disabling this flag.

Time

Time creates another tension between interoperability and security – specifically, the rate of change of systems. For ease of integration, developers endeavor to put programming interfaces in place (described by WSDLs, in the case of Web services) that rarely or never change because the more you change the interfaces, the more the consumers of your services will need to change their own code.

On the other hand, for security, you have to assume a state of constant change for a number of reasons. Most obviously, you want to do things like change passwords, certificates, and the like periodically in order to reduce the risk of attacks that may compromise the system. More acutely, you need to monitor and constantly update patches at all levels of your system to fill

security holes. As the waves of Code Red attacks show, vulnerabilities may persist long after solutions are made available when organizations are not diligent about patches and updates. To reconcile these competing needs, enterprises must carefully weigh the impact of each change made for additional risk prevention versus the amount of business disruption it may cause.

Metadata

The concept of metadata is central to making XML and Web services work. WSDLs, XML Schemas, and UDDI are all systems for providing data about the data in Web services systems. XML draws on a long heritage of SGML (the Standard Generalized Markup Language), and as the name implies, SGML is a very generic way to describe syntax and markup languages. As a result, XML contains a wide variety of mechanisms for metadata or ways to describe the data contained in the document.

One example is how XML documents can specify what schema they should conform to. For instance, a purchase order might have information embedded in its XML message indicating that it is a certain type of purchase order which is defined by a schema file on a different server. At a more fine-grained level, XML allows documents to define entities in terms of other entities in the document, and even allows recursive definitions (defining terms in relation to themselves).

For the purposes of easy integration, this is a fabulously powerful idea – messages that you’ve never seen before can describe to your system exactly how to interpret and use them. For security, though, self-description poses a problem: what and who to trust? Letting messages determine their own schema is a little bit like allowing automobile drivers to define what they consider would be an appropriate driving test, practical exam, and license. It’s pretty good for having lots of people drive, but not such a good idea for safe and secure roadways.

In order to implement a secure system, architects must be a bit more prescriptive about what messages are allowed to look like, and take self-description with a grain of salt. The standards bodies are starting to catch up here, too: the WS-I (a Web services interoperability body), for example, has started by disallowing entity definitions as well as doc type and processing instructions. For security devices dealing with Web services, architects must be even more prescriptive, disallowing documents that point to their own schemas, and instead applying what are known to be appropriate and valid schemas, irrespective of what messages point to.

Standards

Standards are the key point of Web services: by having simple, standard ways for applications to interact with each other, whole new worlds of application integration open up. Systems are easier to understand, they’re more predictable, and they are a lot easier to get to work well with each other.

But establishing standard ways of integrating systems also means there are standard ways of attacking systems. Because the interfaces are so open, structured, and well defined, it’s easier than ever to build hacks that exploit standard vulnerabilities. A Web-based version of this is the so-called script kiddies, the Web’s version of a pack of wild dogs. Increasingly, tools are available for download on the Web by relative novices, enabling them to look for standard vulnerabilities that have yet to be patched and attack these Web sites in standard ways. The tools are, unfortunately, so simple that it doesn’t take more than a few minutes for first timers to get started. We expect Web services to be targeted in much the same way.

The best way to address the tension on standards is to put tools and practices in place that let you stay current with the industry’s best countermeasures and enforce the security policies of your organization.

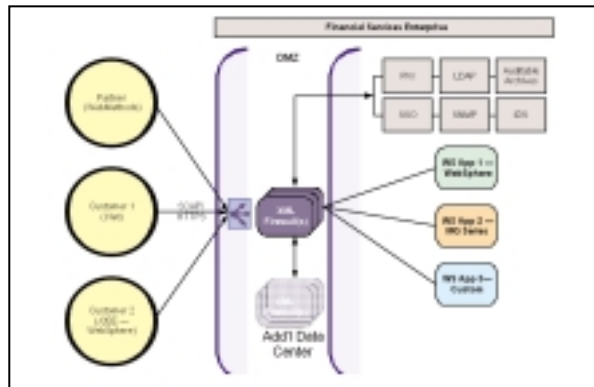


Figure 1 • Security threats

How to Cope

The bottom line is this: you want to be able to build systems that are easy to integrate when you’re setting them up, but act in an extremely secure manner when they are running over time. The only real way to do that is to consider both integration and security concerns from the beginning, and abstract their implementation and management from development efforts.

A proven successful strategy for balancing integration and security is to introduce the new breed of XML firewalls into your organization. XML firewalls act as the Internet-facing gateway to all your Web services, and take care of many of the security tasks that are tedious or impractical for application developers to implement. By moving the responsibility for some of the security tasks to a device at the edge of the network, the XML firewall can catch problematic messages before they’re inside your network, and deal with them before they can do any damage.

XML firewalls can provide robust integration and interoperability points. There are several XML firewall products on the market – one example is the Reactivity XML Firewall, which maximizes interoperability of XML Web services while providing robust and dynamic security policy management (see Figure 2). Using an XML firewall, security architects and business managers can define the level of security enforcement they need to protect the enterprise and also meet their business requirements.

Conclusion

At many levels, the interests of ease of integration and security will always compete – there are simply too many divergent concerns. As a practical matter, though, technologies such as XML firewalls can provide a way for businesses to develop applications that have both ease of integration and best-of-class security designed in from the start. ☛

AUTHOR BIO

John Lilly, Reactivity’s vice president and CTO, leads Reactivity’s product and technical direction. In addition to his role at Reactivity, John currently sits on the board of directors at the Open Source Applications Foundation, is a board observer for CenterRun Inc., and is an advisor to the NewSchools Venture Fund. He holds bachelor’s and master’s degrees in computer science from Stanford University and holds two U.S. patents.

INFO@REACTIVITY.COM

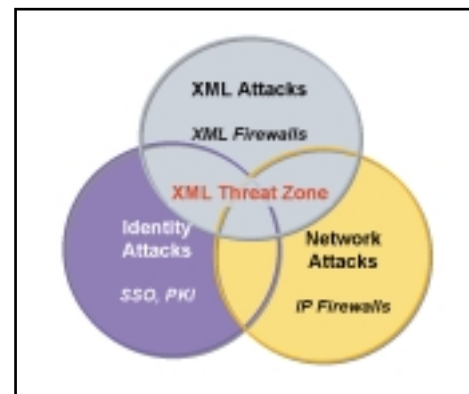


Figure 2 • The XML firewall



Where
Open Minds
Meet



Keynote Speakers



Chris Stone
*Vice Chairman, Office of the CEO,
Novell, Inc.*

Wednesday, January 21
9:15 am – 10:00 am



Dave Dargo
*Vice President, Linux Program Office,
Oracle*

Wednesday, January 21
11:45 am – 12:30 pm



Tom Killalea
*Vice President of Infrastructure,
Amazon.com*

Wednesday, January 21
2:45 pm – 3:30 pm



Sam Greenblatt
*Senior Vice President and Chief Architect,
Computer Associates Intl.*

Thursday, January 22
11:30 am – 12:15 pm



Ross A. Mauri
*General Manager, e-Business on demand,
IBM Systems Group*

Thursday, January 22
1:45 pm – 2:30 pm

CONFERENCE: January 20 – 23, 2004

EXPO: January 21 – 23, 2004

THE JAVITS CENTER: New York, NY

LinuxWorld Conference & Expo is the world's leading and most comprehensive event focusing on Linux and open source solutions. **This January, discover how companies across the globe have achieved higher profits and increased their productivity by utilizing Linux, the fastest growing operating system in the world.**



- Get in-depth coverage on the latest Linux and open source developments and learn about innovative product solutions from the industry's top companies.
- Network with the leaders in the open source movement and discuss with your peers how to best leverage the technology for your organization.
- Participate in LinuxWorld's world-class education program and benefit from interactive training in the all-new Hands-on Labs!
- Attend the 2nd annual Linux Financial Summit and hear how Wall Street firms are leveraging Linux to achieve a lower total cost of ownership. (Sponsored by BEA)
- Discover real world practical solutions and find answers to today's most critical IT issues.

Cornerstone Sponsor

ORACLE®

Platinum Sponsors



Novell.



REGISTER ONLINE WITH REFERENCE CODE: A-XJD

www.linuxworldexpo.com

IDG
WORLD EXPO



XML Acceleration: The Truth Behind the Myths

Don't assume that bandwidth and processing will be problems

As information technology professionals progress in their knowledge and use of XML and Web services, the question of XML performance persists. In hallway chats, one might hear that "XML takes up too much bandwidth" or "XML takes too many CPU cycles to process."

Unfortunately, these beliefs lead to behaviors inconsistent with best practices for building and deploying Web service-based systems that will stand the test of time. These behaviors include continuing to operate with a proprietary non-XML architecture, and designing the architecture around network devices that do hardware XML processing.

This article examines the myths that surround XML performance issues to help IT professionals avoid the pitfalls associated with the behaviors described above.

A Closer Look at XML Bandwidth

Local area networks have lots of bandwidth at a very low cost – but wide area networks are another matter. While they've been improving by leaps and bounds, it can still be prohibitively expensive to expand the capacity of a WAN link and, as such, WANs can be bandwidth constrained, leading, in some cases, to a raw bandwidth issue.

So, what is it about XML that takes up so much bandwidth? There are really two separate issues. The first is that XML is text, which inherently takes up more space than binary formats. A 32-bit integer could be represented in 4 binary bytes, but take over 10 bytes when transmitted in text form (2.5X larger).

The second is that XML is self-describing, which results in lots of repeating patterns of text. For example,

each element name must be explicitly spelled out in both the start tag of the element and the end tag of the element. This adds a lot of extra repetitive text into the document. So, while there are distinct advantages to a self-describing message, it still consumes a lot of bandwidth on the WAN...or does it?

The reality is that many organizations are starting to use hardware compression on their WAN links to reduce bandwidth consumption. And text has the highest compression ratio – in fact, text with lots of repetition can often be compressed 10X. XML is extremely compressible. So, if your WAN links are bandwidth constrained, you should probably be running compression – and if you are running compression, XML will be as efficient as other binary formats and potentially even more efficient since it's much more compressible than a binary stream.

CPU Cycles and XML

What about the CPU cycles required to process XML – doesn't that create a performance drag? It's true that XML is expensive to process. A typical single-processor 1GHz machine can process XML at a rate of about 4MB to 8MB per second, depending on whether you are using DOM or SAX. Now, the real data in that 4MB to 8MB of XML per second is actually significantly less (because of all of the XML tags): that 4MB to 8MB of XML might be equivalent to 1MB or 2MB of actual data processed per second.

But don't forget Moore's law – processing power is increasing rapidly, so what creates a bottleneck today will likely be inconsequential in the near future. Further, if the processing power for XML is a broadly recognized issue, it's highly likely that microprocessor vendors will

add instructions to accelerate XML processing. Don't assume that XML processing performance will be limited by Moore's law – it's likely to surpass it if processing moves into the hardware.

Options for Boosting XML Performance

The promise of microprocessor-based acceleration is great, but where does that leave those who need that extra performance today? A simple but perhaps not so obvious option is to reduce the amount of XML processing being done. For example, when choosing an XML proxy or intermediary (such as a Web service management broker), choose one that processes only the portion of XML required to perform each specific function. Also, avoid chaining together products that perform redundant XML parsing and processing serially. For example, instead of doing XML security processing separate from Web service management and routing, choose a product that integrates both of these into a single processing step.

Another option is to deploy a stand-alone "XML accelerator" appliance. Unfortunately, these don't actually provide the expected benefit. For example, many assume these appliances offload XML processing from the application – in fact they don't. An XML-based application still must parse the XML – there's no way to get around that. What XML accelerators can do very effectively is help convert XML in transit. For example, if a Web service application returns a purchase order in XML, an XML accelerator could convert that order to HTML very effectively. So, it might be better to consider these appliances "XSLT accelerators," since that's their most effective function. While

~continued on pg. 19~

AUTHOR BIO

As chief technology officer at Actional, Dan Foody leverages his extensive experience in enterprise systems software toward designing robust and manageable service-oriented architectures. 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 Web Services Distributed Management Committee (WSDM). Dan holds both a BS and an MS in electrical engineering from Cornell University.



Rational software

See yourself writing code.
See yourself writing better code.
See yourself writing better code, faster.

Can you see it?

You want modeling, diagnostics, debugging, configuration management, all of your development tools—invisibly embedded within your IDE. You want to collaborate seamlessly with your team and eliminate wasted time. Bottom line, you want to write better code faster. You want IBM Rational software. For a free CD with demos, visit ibm.com/rational/seamless

@business on demand™ software



IBM, the e-business logo and e-business on demand are registered trademarks or trademarks of International Business Machines Corporation in the United States and/or other countries. © 2003 IBM Corporation. Rational is a trademark of International Business Machines Corporation and Rational Software Corporation in the United States, other countries or both. All rights reserved.



The Challenge of Web Services Security Inside the Firewall

A true story from the consulting trenches

TTrue story from the consulting trenches: the operations staff had left hours ago, shaking their heads and reluctantly leaving the consultants to resolve a problem with their code. It was well past midnight, in the middle of winter, in a town many time zones from home. The project was late. Altogether, this was an awkward situation that you probably know well.

The consultants – falling into that murky classification of not quite outsider, nor regular employee – worked from hobbled accounts; the security staff were pros and took their charge seriously. By 2:00 a.m., the group was stuck. They needed to change a properties file residing on a remote server, but the distributed file system wouldn't allow it, rightfully sneering at the group like the grubbiest serfs in the kingdom. But there was a Web server...

...And this server was running as root. Before you could say "exploit," our team had all of the rights and privileges of the king of the castle. They tweaked the configuration, muddled the logs, and lo and behold, the software began to run as designed. The client was thrilled the following day; the application moved into production; everybody got paid.

Is this an allegory illustrating the virtues of hacking on the job? No, as it was unethical, possibly illegal, and certainly grounds for termination. No, this is a story about a clash between security models. At the OS/file system level, the consultants were exactly where they should have been: contractors, a little weary, and not entirely trusted. It was a failure at the application level, that is, across HTTP and the Web server, where policy broke down, allowing any one of our friends to become Neo flying about the Matrix. This collapse of the identity model is a common security

problem. It's becoming a particular issue with Web services deployed inside an organization's firewall.

The Internal Threat

It's the outside hackers who receive all the attention. The media is intoxicated with the idea of the teenage misfit outwitting the corporate giant, and all of this attention diverts much of a security professional's time and energy toward addressing the threat. But the threat of the insider can often be as severe, and it may have greater consequences. In its 2003 Global Security Survey, Deloitte discovered that 39% of respondents in the financial services sector had experienced some kind of system compromise in the previous year, and of these, 10% were internal breaches. This might not seem like a lot, but the cost stemming from an internal compromise can be much higher. Gartner estimates that of the security breaches that result in actual measurable loss to an enterprise (as opposed to mere annoyances), an astonishing 70% involve insiders.

Web services are simply a new avenue for the internal hacker to exploit – and it's some very fertile ground to work on. Years of emphasis on perimeter defense have left internal networks open and vulnerable, a mishmash of unpatched systems and obsolete technologies. The irony is that it's into this environment that we're deploying the majority of our Web services applications. Lacking faith in the immature state of Web services security, it just feels safer than tentatively creeping past the firewall. Yet in doing so, we may be placing our data assets at even greater risk.

Principles of Good Internal Security

Security is a game of risk manage-

ment. It's impossible to eliminate risk – that's like eliminating all bugs from software. But risk can be reduced, and the real challenge is finding the balance between comfort and investment.

Good security is a process, not a technology. While technology certainly has its place, as we will see, it's a mistake to think that it alone will solve the problem. Instead, technology fits into four critical areas everyone must address when building an effective internal security model:

1. Instill a culture of security
2. Apply the principle of least privilege
3. Implement a strong identity model
4. Monitor, monitor, monitor

Instilling the Culture of Security

The social process of security is a reflection of corporate culture. Accountability and integrity begin with the executive team and filter down among the ranks. Security policy has to be clear, simple, and well known – on their first day, employees must be made aware of what the expectations are and what the consequences are for violation.

Policy, however, has to be backed up with continuous monitoring and audit. Technology helps here, but it isn't the only answer. Employees themselves are probably the best resource for uncovering security violations, but they need to know that they will be supported when they report that the person in the next cube is accessing records they shouldn't. Clearly, a strong network identity model plays a critical role here. If transactions are anonymous, monitoring is of little value and the social process will break down.

Principle of Least Privilege

After culture, a rigorous application of the principle of least privilege is the most important strategy to confront the

AUTHOR BIO

K. Scott Morrison is director of architecture at Layer 7 Technologies. Layer 7 provides technology for managing and coordinating Web services security and transaction policy across loosely coupled systems.

The 3rd Annual

XML *for* FINANCIAL SERVICES

Maximizing Interoperability, Efficiency & Cost Savings Through Integrated XML-Based Web Services

January 26-28, 2004 • Flatotel • New York, NY

PARTICIPATING COMPANIES & ORGANIZATIONS:

Lehman Brothers

ZapThink

Baltimore Technologies

Deutsche Bank

Trade West Systems

Celent Communications, LLC

Merrill Lynch

International Swaps and Derivatives Association (ISDA)

IFX Forum

FT Interactive Data

DataPower Technology

Federal Reserve Bank of NY

Reuters America Inc.

Dow Jones & Company, Inc.

EDGAR Online Inc.

Fourthought, Inc

Conformative Systems

MDDL Vocabulary Committee

Sarvega

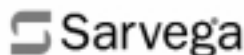
SnapBridge Software

GigaSpaces Technologies Ltd.

THIS CONFERENCE WILL GIVE YOU THE TOOLS, TECHNIQUES AND KNOWLEDGE TO:

- Understand how emerging XML industry standards, including XBRL, FpML, FIX, MDDL, IFX, SOAP, and SAML will facilitate efficient exchanges of information critical to communication and the infrastructure of the financial industry
- Implement a new framework for the collection of financial performance data
- Establish workable, interoperable standards for XML
- Differentiate technology to manage cost while developing dynamic, real-time platforms to access & update information & data
- Expand your flexibility and maximize opportunities with better interoperability and integration of web services
- Understand the evolution & value of Web services, & XML security
- Identify the practicalities of incorporating Web services management into financial services sector IT architecture
- Synchronize systems by incorporating an XML-based office management system
- Unlock the value of market data and solve data integration problems with MDDL & XML
- Improve distribution of content, information and services to clients
- Outsource data collection using XBRL
- Transform your transactions product using Web services
- Understand the challenges & value of automation & implementing validation, pricing & risk management
- Secure XML documents and remote procedure calls (RPCs)
- Use XML to improve central bank web sites
- Recognize the role of FIX in communicating trading and financial information
- Understand federated identity management, using XML, in Web services security

Sponsored by:



Official Website:

WebServices.Org™

Media Partners:



Organized by:



TO REGISTER, CALL 1-800-647-7600 OR 1-781-939-2438

Fax: 781-939-2490 or 781-939-2495 • e-mail: info@worldrg.com • www.worldrg.com

insider threat. This can also be thought of as a need to know, where a person's access privileges are closely mapped to their business function – no more, and no less. It sounds trivial and obvious, but this is a common failing. How many companies have you worked for where several people share a general account? Conversely, how many times have you found yourself in possession of several separate accounts, each under an independent security domain, but all within a single organization? And how many accounts from your previous jobs remain active, overlooked in the ether between HR and IT?

Herein lies much of the challenge for creating secure Web services inside the firewall – that is, managing identity across domains so that the principle of least privilege has any meaning at all. Only when we associate identity with transactions can we harden our internal networks into zones of trust.

This is a particularly acute problem for internal Web services because of the nature of these applications. Web services are most often sold with the promise of driving new revenue through integration of external trading partners; but for most organizations, the reality is much less glamorous: they adopt them for that most dismal of problems, internal EAI.

With EAI comes the challenge of security domain integration, the integration and reconciliation of different security principles that have evolved independently to support a particular application. It's further complicated by a lack of identity in machine-to-machine communications (as compared to human-to-machine, where identity is simpler to establish using manually entered credentials). The most common approach to this issue is to map principals n:1 when crossing application boundaries, aggregating all users into a general access proxy account. It works, but offers only very weak accountability. Now some emerging Web services security standards can provide a much more finely grained identity model.

Identity and Web Services

The first challenge is to establish reciprocal identity between service producers and consumers. The OASIS Web Services Security (WSS) group addresses this. Their core SOAP message security specification defines the fundamental security model for Web services, specifying how to apply the existing XML encryption, signing, and canonicalization standards to SOAP messages. It

defines a generic encapsulation mechanism for security tokens used to establish identity; this accommodates new or existing authentication technologies, such as basic authentication and x509 certificates. Each of these is defined separately, in detail, as a profile. The username profile, for instance, supports basic authentication, and provides mechanisms for digest schemes to enable identification without explicit transmission of passwords. It's very similar to what is already in HTTP, but it's decoupled completely from transport, so that the SOAP message remains secure using SMTP, message-oriented middleware, or even written on a piece of paper.

For many applications, this is fine; as credentials are required, the client pops up a dialog to a user requesting a password and the remote service is accessed, establishing a security context between client and server. It's worked well on the Web for years, gating entrances to subscription services, protecting personal data, etc. Behind the firewall, however, it's somewhat awkward – after all, these credentials might be the same you entered to sign on to your PC when you arrived in the morning. Why isn't this security context simply replicated to the remote server? Or for that matter, why do you have to sign on again when you switch between related applications, say the HR system and payroll?

This is where we find our first real problem, because sharing security context is complex, often very system dependent, and largely not transferable, despite the desirability of this from a convenience and consistency perspective. Fortunately, there are some further Web services standards efforts that are beginning to address this.

Security Assertions

SAML is the cornerstone of this effort. Measured against other Web services standards, it's mature – even ancient – having evolved over a few years and been officially accepted to version 1.1 by OASIS. It suffers from specification that is dense and difficult to read (it's almost completely lacking in examples), but conceptually, SAML is simple. It's an XML framework for exchanging security information about an identity and its entitlements. It does this by defining a schema to declare facts, called assertions, about subjects (a representation of identity). Assertions describe acts of authentication, authorization decisions (allow/deny), or the attributes associat-

ed with the subject. For example, an authentication assertion declares that a subject S was authenticated by means M at time T. An authorization assertion might declare that subject S is authorized to use resource R for access type A based on evidence E.

It's equally important to realize what SAML doesn't do. It does not provide yet another means for an application to perform authentication against a security server—that is, credential transfer and validation. There's no need; this is already well defined elsewhere: in WS-Security, in LDAP, in Kerberos, etc. Instead, SAML defines how applications can check that authentication has previously taken place. It's a crucial point, because it provides the mechanism to synchronize with an existing authentication context, rather than establishing a new, independent one. This is the basis of single sign-on.

SAML on its own has great value when there are multiple systems under a single unified security domain, and in many organizations that have moved toward a single source identity model, this is sufficient. Application servers publishing Web services can be configured in a trust relationship with a centralized SAML issuing authority, which can assert whether a subject has been authenticated, is authorized to use a particular resource, or is associated with specific attributes.

In keeping with its agnostic approach for encapsulating and processing security tokens, the WSS effort simply defines SAML as another profile. Assertions, or references to assertions, can be transported within WSS security headers, providing a means for service producers and consumers to exchange proof of events that established identity. Alternatively, SAML itself defines query mechanisms so that services can question an identity server about authentication events, or request an authorization decision.

We are still missing a crucial piece of the complete identity puzzle. The real challenge, to unite disparate security domain within an EAI project, remains. SAML on its own does not address this, though it could become the plumbing to enable it. For this level of security integration, we need still more standards.

Federation

Federated security is the solution we need. Federation allows different, independent security domains – each of which might use entirely different

means to represent and authenticate identity – to share information about authentication events, authorization, attributes, etc. For example, the HR applications might reside in a domain organized around a central LDAP server that accepts binds with basic authentication credentials. The payroll system, in contrast, runs under control of the accounting department that uses Kerberos. It would be ideal to unite these independent security domains into a trust relationship, under which they share security information.

This is the goal of WS-Federation, another effort by the IBM/Microsoft team. WS-Federation provides the framework for brokering of identities between security domains, including mapping of identities, global sign-out, etc. At the time of this writing, it's quite new, and it relies heavily on other emerging specifications, such as WS-Trust, and on profiles of wire protocols

such as SAML to serve as the glue bonding authentication domains. It will be some time before it sees formal implementation in commercial products, but it shows great promise and is a necessary component to implement a unified security model inside the firewall.

The hype behind federation builds off a vision of independent, but cooperating, businesses (air-car-hotel!); the reality will probably be somewhat more prosaic, but arguably more valuable to the organization. The real triumph of federation will be internal EAI projects, uniting those islands of information inside the corporation that so offend CIOs. And they will work precisely because they are inside a single organization. Federation requires significant stakeholder buy-in. It compels participants to concede a lot of trust to make it work, and often this can only happen under corporate mandate. So in a way, Web services security inside the firewall

is as much opportunity as challenge: an opportunity to protect critical assets from insiders, but also an opportunity to perform EAI with a sophistication that has not previously been attainable. ☛

References

- 2003 Deloitte Touche Tohmatsu Global Security Survey: www.deloitte.com/dtt/research/0,2310,sid%253D3489%2526cid%253D15232,00.html
- Web Services Security Technical Committee: www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss
- WS-Federation Specification: www-106.ibm.com/developerworks/web-services/library/ws-fed/
- Security Services TC (SAML): www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

SMORRISON@LAYER7TECH.COM

WEB SERVICES

~continued from page 9~

Liberty Alliance Project

The Liberty Alliance Project is an alliance formed to deliver and support a federated network identity solution for the Internet that enables single sign-on for consumers as well as business users in an open, federated way. The Liberty Alliance is led by Sun and other industry players.

.NET Passport

.NET Passport is a Microsoft-based initiative that provides a suite of Web-based services that help make Internet and purchasing online easier and faster. .NET Passport provides users with single sign-in (SSI).

Conclusion

Security can be implemented in a cost-effective manner for XML Web services using a combination of the new technologies available as well as existing infrastructure. The use of WSM solutions enables a one-stop shop for security and other critical management and monitoring capabilities. Using these solutions, many organizations are already reaping the benefits of secure Web services today, cutting costs, increasing information sharing, and gaining a competitive advantage. ☛

References

- XML.Gov: www.xml.gov
- CIO Council XML Working Group: www.infoworld.com/article/03/09/01/34NNxml_1.html
- Schafer, Scott Tyler. "XML Exposes Rich Network Data." InfoWorld.
- Jabber Technology Overview: jabber.org/about/techover.html
- Web Services Policy Framework (WS-Policy): <http://xml.coverpages.org/ws-policyV11.pdf>

ANDY@WESTBRIDGETECH.COM

PERFORMANCE

~continued from page 14

they can perform some other forms of XML processing, generally there's little performance advantage in these cases – all of which weighs against the heavy disadvantage of the appliance as a "black box" that can't be managed or extended.

One other alternative looks promising (though the market is still evolving). A number of companies are now building PCI hardware boards that accelerate XML processing. Unlike the appliance "XML accelerators" these boards plug into your application servers and take over the XML processing tasks from the main processor. They do this by plugging in an alternate "provider" under XML

processing libraries that applications use (for example, the Java JAXP XML processor APIs). So, they can actually accelerate any XML-based application (whether developed in-house or purchased) transparently without any changes to the application or the network topology. If performance is critical, these hardware boards are a good tactical solution as you wait for Moore's law or the microprocessor vendors to catch up to your performance needs.

Conclusion

If you find yourself worried about XML processing bandwidth and per-

formance, don't take actions that affect your overall architecture or approach to deploying XML and Web service applications. Often the simplest solutions (such as reducing unnecessary and redundant processing) have the biggest bang for the buck. Beyond this, the best approach is to architect your overall strategy assuming bandwidth and performance will not be a problem. Then if you discover there is an XML processing issue in a specific case, address this with a tactical solution that doesn't undermine your overall strategy. ☛

DAN.FOODY@ACTIONAL.COM



What's Your Government Doing with XML?

With the advent of XML authoring tools, it's more than you might think

Popular wisdom dictates that governments are slow to adopt anything new, especially when it comes to new technologies. But if you look closely, you might find something unexpected.

Across the U.S. and around the world, governments are integrating XML into their workflow. From legislation creation, to Web portals, to intra-department data integration, XML is having a dramatic impact on government workflows. In fact, you may already be using some form of XML-based government service.

Government departments have been working on XML adoption strategies for some time, and in some government corners the process is nearly complete. The U.S. House of Representatives, for example, is currently close to drafting all its introduced bills in XML.

This article will take a closer look at the use of XML for government legislation – an accelerating segment of XML adoption within state and federal governments. It will also examine the history, workflow, and tools being used to spur this rapid adoption.

AUTHOR BIO

Bryan Baker is the product manager for XMetaL at Corel Corporation. Throughout his career, Bryan has been involved with several standards organizations, including ebXML and UDDI. He has spoken at numerous industry events and has authored several XML technology papers. Bryan holds a degree in computer science from the University of British Columbia.

The Need for XML for Government Legislation

To understand how XML is becoming a preferred format for legislation, it is helpful to consider the history of legislation creation. In the U.S. government, the history of structured documents starts with the Government Printing Office (GPO). For decades, the GPO has been responsible for producing paper versions of a massive variety and high volume of government documents. The documents the GPO produces are critical because paper hard copies are still

the document of record (or the golden master, as is said in the software industry).

Fifty years ago, when the GPO started to print documents, technology and standards were sparse. Nonetheless, by the mid 1960s, the GPO deployed a system that used locator codes to enable the specification of typesetting instructions for printing documents. These early computerized publishing systems were using metadata. In fact, locator codes are similar to the kind of metadata that is often captured via XML attributes.

But the actual process of capturing metadata had to be done by someone. Attorneys creating legislation were also trained to insert the GPO locator codes into the documents they were creating. Today this is called hand tagging. Over time, the editing tools used to create the legislation evolved into PC-based software products, but the hand tagging and insertion of GPO codes continued, aided by an enormous series of keyboard shortcuts.

Fast forward to 2003 and locator codes are still alive and well at the GPO. Now they are used to generate PDF documents that can be easily printed. But attorneys are still an ever-present force in government, and their time is more expensive than ever – arguably too expensive for them to be wasting time inserting locator codes.

As a result, many governments have been looking for a more effective alternative. Over the last few years (stretch-

ing back over a decade), government technology experts have been working with XML, and previously SGML, as a way to capture structured content like legislation. Although the content has always been structured, XML also provides a standard syntax and format to markup content.

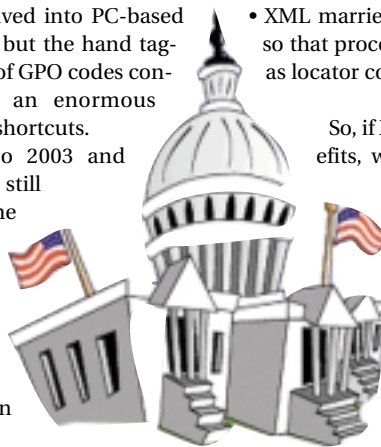
XML is a perfect fit for legislation for a number of reasons:

- XML was designed for creating highly structured content and documents.
- XML enables the legal language in documents to be easily reused.
- XML enables content to be precisely validated, ensuring the accuracy of legal documents.
- XML enables contextual searching of content, including the ability to track relevance to preceding discussions – this is particularly useful for large bodies of content.
- XML marries metadata and content so that processing instructions (such as locator codes) can be acted upon.

So, if XML offers all these benefits, why didn't more governments start using it years ago? Call it anglebracketphobia – a fear of the markup itself. After decades of locator codes, attorneys were hesitant to learn how to use XML syntax to do this kind of markup.

The process of hand tagging a new markup language was too cumbersome. Something needed to be done to make XML simpler to create and manage.

The turning point for government XML adoption came with the availability



ty of XML authoring tools. These tools enable users to easily create XML by hiding the complexity of the markup language. XML authoring tools gave lawmakers the ability to create XML without knowing they were creating it; without worrying about document structure, validity, and well-formedness; and without the need to insert locator codes.

Government legislation has been structured for decades, even centuries. XML provides a standard format in which to express that structure, but XML tools that hide the markup are what allowed legislators to move forward with the full adoption of XML for government legislation.

How It Comes Together

Despite slight variances, the core components of the legislation-creation systems being deployed are consistent – data model, repository, and authoring tool.

Each system revolves around a DTD or XML Schema for the legislation data model. There have been attempts to come to terms on a standard for XML-based legislation, but so far without success. As a result, each jurisdiction typically develops its own.

Each system has a repository. These are a mix of custom database applications and content management systems (CMS), both built on standard RDBMS platforms. Legislative content is stored as reusable XML components, often called “chunks” in the language of CMS. Although many state and federal government offices have invested heavily in content management systems for general content and document management, a smaller percentage are using these systems for XML legislative content.

This brings up a question. If many governments already have a CMS that can reuse content, version it, and publish it, why isn't this good enough for legislation? Why not just use Word? It allows for styles and templates while enabling content chunks that can be reused. And end users are comfortable working in its familiar interface.

The problem is that the ability to validate content is not inherent to a traditional, proprietary word processor file format like .doc.

But unlike proprietary formats, XML is standard and vendor neutral – easily validated and managed by a host of different tools. However, governments interested in adopting XML for legislation faced a significant hurdle. They simply did not have the ability or inter-

ests in adopting a system that required users to mark up content by hand. As a result, XML adoption within modern legislation systems required the introduction of one more crucial component, an easy-to-use XML authoring tool.

An effective XML authoring tool must provide a word processor-like authoring environment that includes features like spell checking, find & replace, revision marking, and keyboard and navigation behaviors. In addition, the authoring tool must understand the XML being created under the covers. This means it must be able to validate the XML content in real time, preventing authors from creating invalid documents that are not well formed.

“XML tools that hide the markup are what allowed legislators to move forward with the full adoption of XML for government legislation”

Word does let users save or publish to an XML format from the native .doc file format, but it doesn't support DTDs, nor does it allow for real-time validation against a schema. This process also requires transformations between the output XML and the target legislation XML. And although Word now also offers direct XML editing and validation of XML Schema-based documents (no support for DTDs), it can require additional coding efforts to hide the element tags and provide customized interfaces.

The ability to run scripts and macros is crucial to building an XML authoring environment within a legislation system. For example, when a bill section is being inserted, it will have mandatory and optional elements as “children” and may have different context depending on its “parent” element. A script may be required to insert a section template complete with replacement text that prompts the author for required input.

Some vendors offer a form-based approach to XML legislation creation. These forms generate XML and provide an interface that is familiar to the content creator. The trick is to offer forms that are a combination of common ele-

ments such as list boxes and combo boxes, along with rich content text boxes. There is a great deal of rich prose in government legislation that may not be generated by a standard form, but needs to be authored nonetheless and combined with the form element data to produce an XML document. This total environment – form elements plus rich content text boxes – can be a powerful authoring front end for those who need to create legislation.

Once the content has been created, the last step is to publish it. In the case of the U.S. federal government, the GPO handles the publishing. The U.S. House of Representatives, for example, uses the XML authoring tool XMetaL for XML legislation creation. Once complete, the House then hands off a valid XML document to the GPO. The GPO system performs a transformation to add the locator codes to the document, ensuring that the final PDF is formatted correctly.

In other environments, publishing may be as simple as applying an XSL or XSL:FO transformation against the legislative content to format for Web or PDF presentation.

As legislative content builds up in governments where such legislative systems are being deployed, additional features of an XML legislation system will be useful. Differencing engines that work reasonably well on semi-structured content (word-processor files) can excel when tackling the differences between versions of XML documents. Likewise, contextual searching of XML archives may be significantly more accurate than generalized full-text searching.

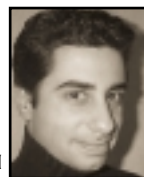
Conclusion

Despite a reputation for moving slowly, governments are demonstrating tremendous leadership in adopting XML for creating legislation. Other organizations, both public and private, can learn valuable lessons from the government example and the benefits XML is providing their publishing workflow.

Structured government legislative content has finally found its format with XML. Now, with XML-enabled content management systems and XML authoring tools, governments can create powerful XML content quickly and efficiently.

For more information about the GPO and the U.S. House of Representatives' use of XML for legislation, please visit <http://xml.house.gov>.

BRYAN.BAKER@COREL.COM



WRITTEN BY JOE MARINI

Designing an Open, Standards-Based Reporting System

XML meets the challenges and design goals of a business reporting system

As XML has grown more prevalent as a data delivery mechanism, so too has the need to use it for presentation in a wide variety of reporting formats. XML is useful for more than just the delivery of information, however. It can be used to help solve a wide range of problems encountered when designing a business data reporting solution, from specifying the layout of the reports themselves to controlling where the data used in the report comes from.

At Panscopic, we develop an enterprise-class data reporting and analytics product that consists of two main elements: the Panscopic Scope Server, which takes report definition files and executes them to produce finished output, and the Scope Creation Suite, a set of client-side authoring tools that create the report definitions that the server executes (reports are called "Scopes" in Panscopic's parlance). Some of the challenges and design goals that we faced when designing the product architecture were:

- Report definitions needed to be in a form, preferably text-based, that was familiar to developers and could be edited outside of our tools if necessary.
- The report definition syntax needed to maintain a clear separation between the report data content and its presentation, and promote the reuse of basic report objects such as queries, layouts, and parameters.
- The system had to be extensible, so that new data sources, layout components, etc., could be introduced over time and from outside of our development organization.
- The product needed to be capable of extracting information directly from XML-formatted data in a natural, standardized way.

We found that XML was particularly well suited to solving these design challenges. For example, the reports themselves are created and stored as XML files, which are then loaded and executed by the Scope server. Also, each report file contains a reference to one or more data sources that are exposed by the server. The server maintains this list of available data sources in another XML file, which can be edited by an administrator to point to whatever data source is desired. Since the reports' references to these data sources go through this abstraction layer, administrators can change the data source pointers as often as they like without affecting the reports (as long as they follow the same schema, of course). Finally, we used XPath, the W3C standard for referring to tags within an XML structure, to extract information from within XML-formatted data.

The Report Definition Language

To achieve our higher-level design goals, such as separating content from layout, promoting object reuse, and extracting specific information from XML data, we needed to go a little further with our design than just finding a representation for everything as XML tags. The solution we came up with was the Report Definition Language, or RDL (pronounced "riddle"). RDL is an XML-based file format that describes how a report retrieves its data, manipulates it, and realizes the result. RDL files are divided into top-level sections, each represented by an XML tag, that contain the different parts of the report. The most important sections of an RDL file are `<rdl:parameters>`, `<rdl:content>`, and `<rdl:layout>`. Listing 1 shows an example RDL file.

The parameters section contains

descriptions of the reports parameters, which act essentially like variables passed to the report at runtime. Parameters can either be fixed-form, meaning that the value is restricted to one from a predefined list of values, or free-form, in which case the value can be anything. Each of these parameters can be assigned to a form control on a Web page that supplies its value, or the value can be directly assigned in the URL that is used to request the report from the server. Parameters can also be assigned a default value to be used if none is supplied by the user, and can be marked as mandatory, indicating that the user must supply a value for it.

The content section indicates where the data for the report will be drawn from. Inside the content section are one or more `<rdl:data>` tags, each of which specifies a data source that supplies data to the report. These data sources are maintained in a list by the server (as described earlier); Listing 2 shows an example of a data source entry that might appear in the server's configuration file. Each data tag contains sub-tags that are specific to the type of data source being accessed. The example shown in Listing 1 is using a connection that resolves to a relational data source (as indicated by the `<rdl:rdbms>` tag). The type of data source and the way it exposes data columns is kept abstracted from the layout by the `<rdl:return>` section contained within the data section. It is the job of the `<rdl:return>` section to expose the columns of data returned by the data source to the layout section in a uniform way. This approach allows vastly different types of data and layout components to be hooked together seamlessly.

The layout section determines how the report will be visualized for the user.

AUTHOR BIO

Joe Marini is a senior engineer at Panscopic Corporation (www.panscopic.com) and XML and J2EE standards-based reporting solution provider. Joe has written and collaborated on a series of books about Web development.

Of course, not all reports are necessarily consumed by humans: the report may be delivered in XML format for consumption by another service. Inside the layout section are one or more `<rdl:useComponent>` tags, which refer to layout components used to format the data specified in the content section. Layout components are specified and configured in XML, but are implemented on the back end as JSP pages. Each component has the built-in ability to realize the report as one of a number of different formats, such as HTML, XML, or PDF.

By keeping these sections distinctly separate, the report is broken up into its constituent parts, each of which can be saved and reused in other reports. For example, a query that is written for one report can easily be saved and stored in the server's network-accessible catalog for use by another developer in a different report, possibly with an entirely different layout. Similarly, a particular layout can be used again and again with other data queries.

Extracting Data from XML Data Sources

To address the design requirement of being able to extract information directly from an XML data source, we turned to

XPath, the W3C standard for navigating among the nodes of an XML structure.

The example shown in Listing 3 illustrates a content section that is using an XML data source (indicated here by the `<rdl:xmlsource>` tag). You can see from this example that the `<rdl:return>` section mentioned earlier makes use of certain attributes that contain XPath syntax within them. The `<rdl:return>` tag itself has a "selectNode" attribute, and each of the `<rdl:column>` tags has a "fieldPath" attribute. These attributes contain XPath expressions that refer to specific tags in a returned XML data structure.

The `selectNode` attribute identifies a set of nodes in the XML data that corresponds to repeating, "record-style" information from which data is to be extracted. The Scope server iterates over the set of nodes that matches this expression and evaluates the `<rdl:column>` tags' `fieldPath` attribute expressions against each of those nodes. In this way, the data is extracted from the XML and presented to the layout section in the same way that two-dimensional data from traditional JDBC sources is, reducing the complexity and required learning curve for the developer. In addition, the XPath expres-

sions can be written to provide further filtering and processing on the returned data.

Conclusion

Using XML to solve our design requirements had several beneficial results. First, we were able to take advantage of a wide range of available open-source code to perform common XML operations, such as parsing the code to build DOM trees for editing and using SAX to process the files on the server. Second, using XML allowed us to keep the format of our reports open and text-based, which in turn allows developers to use whichever tools they are comfortable with and to work with a syntax with which they are already familiar. This also made it easy for us to define extensibility APIs that allow customers to add their own components to the product in a uniform, easily understood way, and that simplify administration tasks such as adding new data sources to the system. Finally, we are better able to take advantage of new XML technologies as they become available, such as XPath and XQuery, for working with native XML data. ☛

XML@PANSCOPIC.COM

LISTING 1

```
<?xml version="1.0"?>
<!DOCTYPE rdl:RDL SYSTEM "ReportDescriptionLanguage.dtd">
<rdl:RDL name="PanscopicScope"
xmlns:rdl="http://www.panscopic.com/RDL">
<rdl:parameters>
</rdl:parameters>
<rdl:content>
<rdl:data name="Query1" readonly="false">
<rdl:rdbms prefetch="false">
<rdl:connection name="sample"/>
<rdl:sql><![CDATA[
SELECT CUSTOMERS.COMPANY_NAME, CUSTOMERS.CONTACT_NAME,
CUSTOMERS.ADDRESS, CUSTOMERS.POSTAL_CODE,
CUSTOMERS.COUNTRY
FROM CUSTOMERS
]]></rdl:sql>
<rdl:return>
<rdl:column name="CUSTOMERS_COMPANY_NAME" type="string"
index="1"/>
<rdl:column name="CUSTOMERS_CONTACT_NAME"
type="string" index="2"/>
<rdl:column name="CUSTOMERS_ADDRESS"
type="string" index="3"/>
<rdl:column name="CUSTOMERS_POSTAL_CODE"
type="string" index="4"/>
<rdl:column name="CUSTOMERS_COUNTRY"
type="string" index="5"/>
</rdl:return>
</rdl:rdbms>
</rdl:data>
</rdl:content>
<rdl:layout>
<!-- Layout components go here -->
<rdl:useComponent name="PersonalInfo" type="StandardTable">
```

```
<rdl:arg name="tableStyle">width:75%;</rdl:arg>
<rdl:arg name="tableBorder">1</rdl:arg>
<rdl:arg name="titleLabel">Personnel Report</rdl:arg>
<rdl:arg name="showColumnHeaders">true</rdl:arg>
</rdl:useComponent>
</rdl:layout>
</rdl:RDL>
```

LISTING 2

```
<DBConnection type="jdbc">
<name>sample</name>
<driver>{sample.jdbc.driver}</driver>
<url>{sample.jdbc.url}</url>
<username>{sample.jdbc.username}</username>
<password>{sample.jdbc.password}</password>
</DBConnection>
```

LISTING 3

```
<rdl:content>
<rdl:data name="XMLQuery1" readonly="false">
<rdl:xmlsource
url="http://some.data.source.com/datafile.xml">
<rdl:return selectNode="//DataNodes[@state='NY']">
<rdl:column name="FirstName" type="string" field-
Path="./firstname"/>
<rdl:column name="LastName" type="string"
fieldPath="./lastName"/>
<rdl:column name="Address" type="string"
fieldPath="./address"/>
<rdl:column name="Phone" type="string"
fieldPath="./phone"/>
</rdl:return>
</rdl:xmlsource>
</rdl:data>
</rdl:content>
```

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



Building a High-Traffic Web Site with Static Delivery Using XML

WRITTEN BY
TODD PRICE

Dynamic and static sites: get the best of both worlds

More and more companies are experiencing a need to effectively manage ever-changing content on high-traffic Web sites. These high-traffic sites receive as many as 1 million hits per day and require significant amounts of technical and financial support. The question of how to create a cost-effective, efficient system to support these sites has fueled a debate among IT professionals over the value of delivering them dynamically versus statically.

Dynamic sites – sites that pull and assemble content stored in a repository – are easy to customize and capable of providing rapidly changing content. These sites provide the richest feature set for contributors and are easier and cheaper to build and manage. However, a high-traffic dynamic site requires hefty investments in software and hardware to support the high consumption requirements. And, as the site grows, so do the costs of additional equipment and staff to manage the site's burgeoning content and increasing traffic.

On the other hand, static sites – sites built and delivered

with HTML code – are capable of handling more traffic with less hardware and software. Static sites are cheaper to deploy and generally have the highest guarantee for both the consumers' response time and overall availability. However, they tend to be much more difficult to service behind the scenes. For example, because changes to content on static sites must be coded in HTML, Web masters, rather than nontechnical employees, must manage all updates, which can cause time-consuming, costly delays in providing fresh content to users.

So how can companies tap the best qualities of dynamic and static sites to efficiently manage a high-traffic Web site? Build the Web site dynamically but deliver it statically.

Dynamically Build a High-Traffic Web Site with XML

Companies using a Web content management (WCM) system to edit and submit content to a high-traffic site that is dynamically built and statically delivered may experience increased efficiencies and savings in time and cost. Following

is a description of how the “dynamic creation/static delivery” model works (see Figure 1).

Most Web sites today are made up of parts, with each Web page containing multiple pieces of content. Fundamentally, a Web page consists of four pieces: content fragments, look-and-feel of content, navigation, and look-and-feel of navigation. For a Web site to be manageable, these four pieces have to be separated and managed independently. Simply put, the content contributor should not have to worry about which color schemes to use, while the Web designer should not have to modify, or “re-brand,” each Web page after content is contributed.

Almost all of these pieces can be managed as XML. For example, content fragments – generated either by converting content authored in common desktop applications or by capturing data in template-based HTML form – can be stored as XML. Similarly, navigation information, which is usually hierarchical, can be stored as XML. And, a Web page’s layout can be driven by XSL files, and its color scheme can be driven by XML or Cascading Style Sheet (CSS) files. All of these pieces can be assembled into a dynamic Web site using standard technologies, such as ASP, .NET, and JSP.

In-Context Contribution, Preview, and Workflow

A fundamental aim of the “dynamic creation/static delivery” model is to simplify the contribution and content-approval process for nontechnical contributors while providing a fast and scalable static site for general consumers.

An underlying WCM system enables contributors to navigate to any section of the Web site and edit content within certain Web pages. For instance, a company’s public-relations person should be able to log in to the dynamic Web site; navigate to the “Press Releases” section; and click a button or link to add a new press release. Similarly, a customer-support staff member should be able to navigate to the “Knowledge Base” section of the Web site, and click a button or link to add a frequently asked question (FAQ). Contributors can “Preview” their changes in real time, in full context of the entire page, and even in full context of the entire site, when a particular change affects more than one page.

In addition, security permissions built into a WCM system dictate contributors’ rights to edit content on Web pages. For example, when a Web page has two content fragments, one person may be able to edit the first fragment, while another person may be able to edit both of them based purely on their security privileges. This method of editing content from within the context of the Web site is referred to as “In-Context Contribution.” Application code in the Web site ensures that all content is converted to XML, if necessary, and tagged with appropriate metadata.

In this model, workflow capabilities of the WCM system should also be exposed via the Web site. Application code in the Web site should ensure that members of a workflow are able to preview content in full context of the Web site, as well as approve or reject content right from the Web page. A dynamic site is the cornerstone to “In-Context Contribution,” “Preview,” and “In-Context Workflow” since it would not be practical to have a static rendition of a Web site for every piece of content in workflow at any given time.

With these WCM features in place, contributors can efficiently add content to the Web site; content and editorial errors can be reduced significantly, if not eliminated completely; and accurate content can be added to a Web site in a timely fashion.

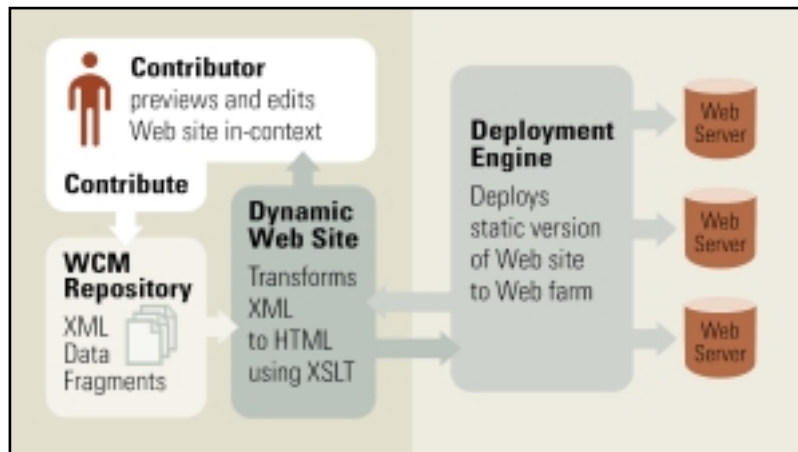


Figure 1 • Dynamic creation/static delivery

Static Consumption

Once a Web site has been dynamically populated with content, a static version of the site needs to be captured, and the static Web pages need to be deployed to the Web servers running the Web site.

Publishing and deployment software offered by some WCM vendors have the ability to capture static versions of dynamic sites. However, before adopting such software, it is important to ensure the technology has the capability to run on demand, on a scheduled basis, or both. In addition, the software should be capable of capturing either complete sites or only sections of a site. The software also should be able to distribute content to all assigned Web servers, either directly from the content management repository or from a file system.

Another important feature to look for in publishing and deployment software is “guaranteed delivery” – the ability to push content updates out to all assigned Web servers but withhold the updates from all of the servers if deployment to one server fails. This feature ensures content consistency in load-balanced or fail-over environments. Finally, the ability to run incremental deployment cycles is also important. With this feature, only pages that have changed since the last deployment cycle are pushed out to Web servers.

Benefits of Building Dynamically and Publishing Statically

The dynamic creation/static delivery model taps the best qualities of dynamic and static sites to efficiently manage a high-traffic Web site. The dynamic contribution environment enables contributors to efficiently submit and edit content on the site, while the static version ensures rapid delivery to the general public.

Using XML for such a deployment helps in the separation of content from branding and navigation, enabling the reuse of content and simplified Web site management. For example, content updates do not require Webmaster or developer involvement. These advantages are noticeably absent in static sites, where content, layout, and branding are intermingled.

Since the dynamic Web site in the dynamic creation/static delivery model caters to a small audience, the infrastructure needed to maintain it is significantly smaller than what is needed for a public-facing dynamic site. Therefore, companies experience significant cost savings. A single WCM-based dynamic site developed using JSP technology and running on an application server or servlet engine can easily support tens to hundreds of content contributors. This same Web site could be deployed statically on a similar-size server and support mil-

Blockbuster Entertainment Studio Taps Dynamic Creation/Static Delivery Model to Manage Popular Web Site

One of the largest entertainment studios in the world uses the dynamic creation/static deliveryZ approach to simplify and expedite frequent updates to a Web site for its various products, including movies, home videos, and DVDs. The site regularly has 4–6 million visitors per week and more than 10 million hits per day. All of this activity is handled by two dual processor servers, neither of which is fully utilized.

By using this approach, the studio has reaped significant time and cost savings because the system allows nontechnical employees, rather than a few IT staff members, to add and change information to the site. This approach not only serves the dynamic needs of contributors, but also the high-volume needs of consumers, all the while reaping the benefits of low hardware and software requirements. Best of all, the visitors to the studio's site

see relevant, up-to-date content.

The studio has 30 content producers, ranging from graphic designers to content editors, who contribute information to the site. Using a content management system, the content producers use simple updating mechanisms, such as pre-built templates; in-context editing; and automatic conversion and publishing capabilities to easily create and maintain content on the studio's Web site – without relying on IT staff.

When dynamically building its Web site, the studio has one production instance of the dynamic site, which is written in JSP and uses XML and XSLT to display content for the contributors. To ensure that the site is updated regularly, the studio has scheduled deployment of the dynamic site to each static rendition for general consumption. The studio's static site also can be updated on demand.

lions of consumers since the static delivery of a site allows for a smaller and easier-to-manage infrastructure. For instance, a 2GHz, single-processor Windows server can deliver roughly


500 static HTML pages per second, which equates to about 1.8 million pages served in one hour.

Finally, Web content management packages used in this model offer numerous benefits. Features such as workflow, conversion of content to XML, and template creation and assembly tools play a crucial role. Out-of-the-box capabilities for in-context editing and workflow ease the contribution process and reduce development costs. And, on-demand publishing and regularly scheduled publishing cycles ensure that content on the public-facing Web site is always up to date.

Dynamic vs Static Delivery – Consult Your Bottom Line

As companies consider whether to deliver their high-traffic sites dynamically or statically, it's best to consult the bottom line, as the long-term gains from using the dynamic creation/static delivery model outweigh initial investments.

Web sites built and delivered using this model enable anyone to contribute content quickly and efficiently to the site, which reduces staff costs and dramatically increases staff efficiency and productivity. Moreover, content is updated with high levels of security and built-in corporate guidelines and style sheets, reducing costly, time-consuming errors.

But perhaps most important, the millions of valuable visitors and customers who use high-traffic sites every day see fresh, up-to-date content, increasing the chance of repeat visits and/or online purchases, as well as improved brand loyalty. 

AUTHOR BIO

Todd Price is vice president of product management for Stellent, Inc., a global provider of content management solutions. Todd is responsible for driving the strategy and development of Stellent's universal content management solutions. He has more than 18 years of experience in the software industry, and has managed all levels of the software life cycle. Todd holds a BS degree in mathematics and computer science from Saint Cloud State University.

TODD.PRICE@STELLEN.T.COM

XML-J ADVERTISER INDEX			
ADVERTISER	URL	PHONE	PAGE
Altova	www.altova.com	978-816-1600	44
BEA	dev2dev.bea.com/thought	925-287-5156	2, 3
Edge 2004 East	www.sys-con.com/edge	201-802-3069	32-39
Ektron	www.ektron.com/xmlj	603-594-0249	6
IBM	ibm.com/developerWorks/toolbox/seeit		43
IBM/Rational	ibm.com/rational/seamless		15
LinuxWorld Conference & Expo	www.linuxworldexpo.com		12-13
LinuxWorld Magazine	www.sys-con.com	888-303-5282	41
Mindreef	www.mindreef.com	603-465-2204	4
MX Developer's Journal	www.sys-con.com	888-303-5282	27
XML for Financial Services	www.worldrgr.com	800-647-7600	17

General Conditions: The Publisher reserves the right to refuse any advertising not meeting the standards that are set to protect the high editorial quality of *XML-Journal*. All advertising is subject to approval by the Publisher. The Publisher assumes no liability for any costs or damages incurred if for any reason the Publisher fails to publish an advertisement. In no event shall the Publisher be liable for any costs or damages in excess of the cost of the advertisement as a result of a mistake in the advertisement or for any other reason. The Advertiser is fully responsible for all financial liability and terms of the contract executed by the agents or agencies who are acting on behalf of the Advertiser. Conditions set in this document (except the rates) are subject to change by the Publisher without notice. No conditions other than those set forth in this "General Conditions Document" shall be binding upon the Publisher. Advertisers (and their agencies) are fully responsible for the content of their advertisements printed in *XML-Journal*. Advertisements are to be printed at the discretion of the Publisher. This discretion includes the positioning of the advertisement, except for "preferred positions" described in the rate table. Cancellations and changes to advertisements must be made in writing before the closing date. "Publisher" in this "General Conditions Document" refers to SYS-CON Publications, Inc. This index is provided as an additional service to our readers. The publisher does not assume any liability for errors or omissions.

A new tool for MX professional developers and designers...



ADVERTISE

Contact: Robyn Forma
robyn@sys-con.com
(201) 802-3022
for details on rates
and programs

SUBSCRIBE

[www.sys-con.com/
mx/subscription.cfm](http://www.sys-con.com/mx/subscription.cfm)
1 (888) 303-5282



Advanced ANSI SQL Native XML Integration—Part 2

Supporting advanced XML capabilities

Part 1 of this article demonstrated how standard ANSI SQL can integrate fully, naturally, and seamlessly with XML. This was accomplished by naturally raising SQL processing to a hierarchical level, enabling relational data (including XML-shredded data) to integrate at a full hierarchical level with native XML. Hierarchical processing and the utilization of the hierarchical semantics were also shown in Part 1, along with the hierarchical joining of hierarchical structures.

Part 2 will cover how standard SQL can naturally support more advanced XML capabilities such as node promotion, fragment processing, structure transformation, variable structures, and the handling of shared and duplicate elements.

Node Promotion and Fragment Processing

SQL's hierarchical processing capabilities do not stop with what was presented in Part 1. We will look at supporting XML's more advanced capabilities starting with node promotion and fragment processing. Node promotion occurs when a node in a hierarchical structure moves up past its parent and ascendants that have not been selected for output (this is controlled by projection in relational terms). This slicing out of nodes in the structure definition has the same effect as slicing them out when the data selected is transferred from the relational Working Set to the Result Set. This is shown at the bottom of Figure 1. This means that this standard XML hierarchical processing capability is also performed naturally in relational processing when a node (table or element) is not selected for output.

From what we saw in Part 1, the basic operation of SQL is to use the SELECT clause to specify the desired output data; the FROM clause to specify its input data; the LEFT JOIN to specify its data structure; and the WHERE clause to specify optional data filtering criteria. Using these simple and intuitive SQL language constructs, even more complex node promotion leading to fragment processing can be easily performed. The query in Figure 1 joins a structure fragment selected from the lower structure to the upper structure. The structure fragment is shown encircled in a dashed oval. Dashed boxes represent nodes that are not selected for output.

A fragment is a portion or grouping of nodes from a hierarchical structure that retains its basic hierarchical structure when unselected nodes are removed, enabling it to be manipulated as a unified structure. It can be embedded below the original root of the input structure and can be a loose collection of nodes like the structure fragment shown in Figure 1. It is defined by the nodes that are selected for output from its defined input structure. In this example, the fragment is located below the root node D of its original input structure, which contains nodes O and I, which are not selected for output and will not be included in the fragment. This causes nodes Y and J to be automatically promoted over nodes O and I so they both naturally collect under the fragment root node F. In fact, the whole process of fragment creation and processing is made possible by the natural action of node promotion.

With the structure fragment identified and isolated by data selection, it can be naturally joined to the upper struc-

ture as shown in Figure 1. When linking to a lower-level fragment, it is usually to its root node as in this example, where the link point is below the root of the original input structure node. Linking below or above the root of the fragment is also permitted because the root of the original lower structure remains the defining input structure's root (node D in this case) because it still affects the entire input structure. This is particularly important to logical structures that must be built on the relationships from the root down. However, this doesn't mean that these structure-defining nodes need to be in the output structure. For example, in Figure 1 the root node D and the link point node P in the upper structure are not selected for output, but they are necessary for processing the query. Node E, on the other hand, was not selected and is not necessary to the query so it can be optimized out, which is why it is not in the Working Set. The shaded columns in the Working Set are not selected for output, so they are not transferred to the Result Set shown at the bottom of Figure 1. This relational processing action naturally performs hierarchical node promotion, which in turn collectively supports fragments. These advanced procedures and concepts are being performed automatically and are easily controlled by the data fields specified on the SQL Select list, which can also be specified dynamically.

The advantage of linking below the root is that it allows linking using the most appropriate or convenient link criteria. The filtering of the lower-level structure matches the semantics of the link point, which makes sense semantically. Whether you want to link to the lower-level structure in Figure 1 based

AUTHOR BIO

Michael M David is founder of Advanced Data Access Technologies, Inc. He has been a staff scientist and lead XML architect for NCR/Teradata and their representative to the SQLX Group. He has researched, designed, and developed commercial query languages for heterogeneous hierarchical and relational databases for over 20 years. He has authored the book *Advanced ANSI SQL Data Modeling and Structure Processing* (Pitman House Publishers) and many papers and articles on database topics.

on a value in node D, F, O, or even J, the Result Structure shown would retain the same hierarchical structure for any of these link points because root D is still the defining root, and the resulting data (not shown) would match the actual link point semantics.

In SQL, users don't need to know about the concept of fragments, they just select the data they're interested in and the fragments will naturally form. The example in Figure 1 neatly generated a single fragment because there was a single fragment root node F, but this is not a requirement imposed on the user. Suppose node E had also been selected for output, creating two unrelated fragments. This does not present a problem because root D is still the defining root that controls how all the fragments are structurally linked to the upper structure. In this case, root E would be located in the Result Set between nodes B and F. This demonstrates the significant power, flexibility, and ease of use of SQL's nonprocedural hierarchical processing. While the underlying fragment-processing logic may seem complex to perform, its specification is very logical, naturally intuitive, and is performed automatically in standard SQL.

Duplicate and Shared Element Support

Now let's look at how to process XML's duplicate and shared elements, which can occur in an XML data structure when accessed from standard SQL. Duplicate elements occur where the same (named) element type occurs in multiple locations in the XML data structure. Shared elements are created by the XML IDREF and ID attribute constructs, which create a logical pathway in the physical XML data structure (usually creating a network structure). Both of these unconventional structures are demonstrated in Figure 2 as the top two structure diagrams. The Addr node in these diagrams represents the duplicate and shared element. The dotted arrow in the XML shared element diagram in Figure 2 represents the logical IDREF pathway. The problem with these two structures is that they are both ambiguous for a nonprocedural hierarchical query language such as SQL because there is no single unambiguous access path to a specific Addr node location. What makes nonprocedural hierarchical query languages so powerful is that the hierarchical data structures they operate on are naturally unambiguous because they only have a single path to each node. In this way, the query can be spec-

ified unambiguously because each node in the structure has its own unique access path and specific semantics that can be utilized automatically to answer the query.

The Alias feature of SQL allows the duplicate and shared element structures shown in Figure 2 to be data modeled as unambiguous hierarchical structures by using the optional AS keyword to rename the elements (nodes). In this example, both of the ambiguous structures can use the same data modeling SQL to produce an unambiguous structure that maintains the original semantics of both input structures. This is possible because the semantics of both input structures with differing storage use of the Addr node, are the same and produce the same result. With the unambiguously modeled structure shown in Figure 2, each specific Addr field can be unambiguously referenced by using its unique node name as a prefix to the field name. In this way each Addr field name reference has its own logical path with its own hierarchical semantics. This allows the full nonprocedural hierarchical power of SQL to be easily controlled with simple intuitive queries. Avoiding the use of node name prefixes can be accomplished by using the SQL Alias feature on the SELECT list to rename the duplicate field names to unique names. The underlying XML access module's logic will adapt transparently to the physical storage representation of the Addr element whether it is shared or duplicated.

Hierarchical Data Filtering

There are two levels of hierarchical data filtering supported with ANSI SQL: the WHERE clause query-level data filtering, and the ON clause path-level data filtering. WHERE clause filtering is the most common of the two. It can affect the entire multi-leg hierarchical structure by not only filtering down the structure, but also up the structure from the filtered node points. The ON clause data filtering, on the other hand, filters only downward from its filtered node point and is compatible with XPath data filtering. Both of these SQL data-filtering operations work consistently across ANSI SQL relational processing and also follow standard hierarchical structure processing semantics.

Figure 3 demonstrates WHERE clause query-level data filtering. Notice how it also filters up the structure affecting the ascendants of the explicitly filtered Dpnd node. In this example, the only Dpnd node "Dpnd2" for the Emp

node occurrence "Emp2" is filtered out, which causes "Emp2" to be filtered out because the "Dpnd2" path occurrence is filtered. The associated root node data occurrence "DeptX" is qualified because there are other active paths leading to it from the node occurrence "Dpnd1", which was qualified. Also notice that node occurrence "Proj1" is also qualified because its path is still active and is a descendent of the qualified node occurrence "DeptX". If no Dpnd node occurrences had been qualified, then no data

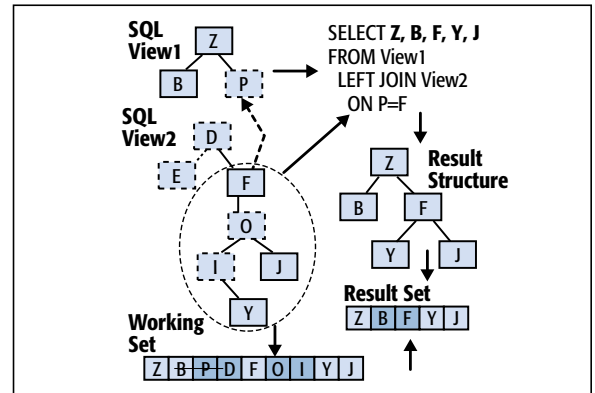


Figure 1 • ANSI SQL hierarchical fragment processing

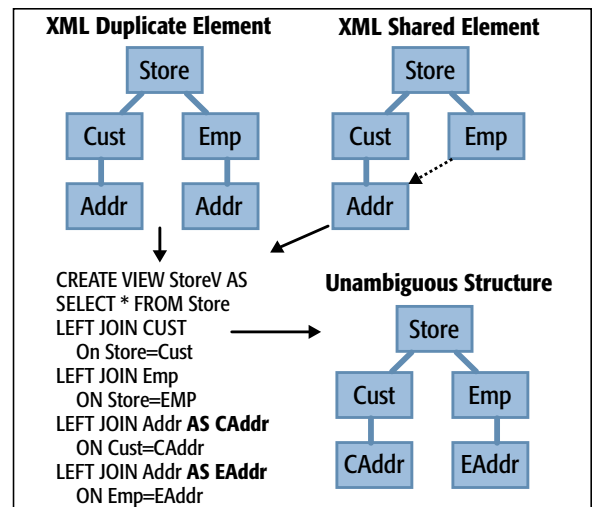


Figure 2 • 2 SQL mapping of duplicate and shared elements

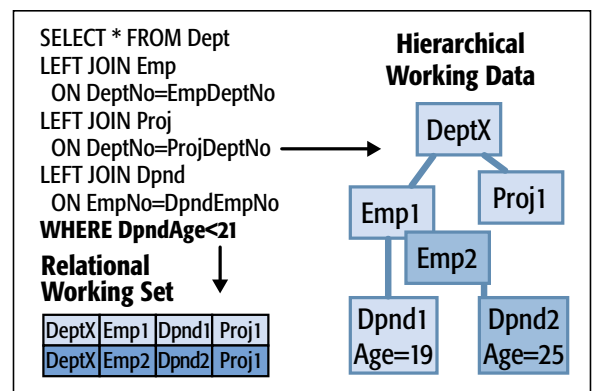


Figure 3 • WHERE clause query-level data filtering

from this structure occurrence would be output. This behavior is logical and intuitive for the hierarchical result you would expect with this hierarchical WHERE clause filtering and is how hierarchical processors operate. This is also the result of relational processing (shown in Figure 3) by the Relational Working Set in this example, where the deleted row is filtered out, indicated by a darkened row. This is why the WHERE clause filtering process can affect the entire multi-leg structure occurrence, taking into account (correlating) the semantics between the data and relationships in its sibling legs. As shown in Figure 3, this complex hierarchical semantic processing occurs automatically thanks to the restricted Cartesian product, producing the required combination of hierarchical related data values allowing data selection to be carried out a row at a time by the relational engine.

Figure 4 demonstrates ON clause path-level data filtering using the same filtering criteria used with the Where clause query-level data filtering. ON clause filtering is similar to WHERE clause data filtering, but only filters from the point on the path where it is used and downward to its dependents. Its power is its fine hierarchical data filtering that is isolated to a single node in the structure. This operates just like XPath data filtering. It is also the result produced from relational processing, shown by the Relational Working Set in this example, where only the filtered Dpnd node occurrence "Dpnd2" is deleted. If there were other descendant nodes under the filtered node, they would also be filtered out. The reason that the entire row was not filtered is that the higher-level structure side (left side) of the LEFT JOIN is always preserved, enabling filtering to only occur from its right structure side down the structure. The WHERE clause filtering, on the other hand, is performed using INNER JOIN logic, which means that either side of the join operation (up or down the structure in this case) can cause nodes to be filtered out. Additionally, WHERE clause processing occurs after the entire row is built, causing it to be entirely filtered out or not filtered at all, while ON clause filtering occurs as the row is being built, allowing separate legs of the structure to be filtered out replacing their values with Null values to keep the alignment. This ON clause path filtering mirrors XPath's operation on XML and can be simulated

for legacy database access when specified using SQL.

Structure Transformation

By combining fragment processing and the SQL Alias feature used in duplicate and shared element processing in Figures 1 and 2, it's possible to easily perform powerful structure transformations. This operates by enabling the specification of different fragments from the same structure by using the Alias capability to logically create multiple copies of the same structure so that different fragments can be isolated and then independently manipulated. The Alias feature is used to rename views, which enables duplicate input structures to be logically defined so that references to them can be made unambiguously (also useful for processing XML namespaces). Figure 5 demonstrates this by

"standard ANSI SQL can integrate fully, naturally, and seamlessly with XML"

creating two separate and independent fragments from the StoreView view (encircled by dashed ovals). These fragments are then recombined into a different structure by rejoining them. This is a simple example of structure transformation; multiple structures can each have multiple fragments extracted, which can all be combined in any order, allowing fragments to be joined as they are needed. Structural transformations can also be stored in a SQL view for abstraction, easy reuse, and use in constructing larger structures.

Variable Structures

XML can define variable structures, which allow for considerable variability of structure formats for a single definition of the structure. This means that from structure occurrence to occurrence, or even within a single structure occurrence of a record or document, the structure format can vary. While XML does not require it, a variable structure usually has some piece of information from a higher-level node that indicates how a variable substructure is formed or

is to form. With this information and using the ON clause filtering described earlier when hierarchical data filtering was covered, SQL data modeling (using Left Joins) can control the building of each structure occurrence to define the appropriate substructures dynamically. An example is shown in Figure 6.

Figure 6 is a simple example of how the generation of the data structure can vary depending on the value of the field StoreType in the Store node. In this case, only one area of the structure was affected, but there's no limit to the number of variations that can be controlled by ON clauses testing for any number of structure indicator values to control how they are generated depending on their current data values. In fact, variable substructures can contain variable substructures. These tests can be coded to duplicate the rules specified in XML DTD and schemas for varying element generation, which can become quite complex. The SQL for specifying a variable structure, as shown in Figure 6, defines how a logical (relational) structure is to be constructed in memory, or can be used to control the navigation of a physical (XML) structure when being retrieved into memory; and in either case, it controls how the variable output structure is generated.

Comparison with XQuery and the SQL/XML Standard

XQuery's use in SQL requires learning another query language and having to program the query logic into its FLWR (For, Let, Where, Return) expression. SQL's SELECT list functionality is contained in the FLWR expression, which is controlled by its FOR loops. This gives XQuery considerable procedural processing power and control, but also means that specifying additional output values is not trivial, requiring program logic modification. The FLWR statement is also used to control or drive database access using XPath navigation. This requires the XQuery developer to be knowledgeable of the data structure being processed. The XML output is constructed with the use of XML templates, which requires that the developer know XML and can include placing the templates in FLWR FOR loops for additional control. XQuery uses functions to abstract and reuse program logic.

With the ANSI SQL native XML integration solution shown in this article, the SQL developer does not need to

know XML or the data structure (once it is modeled in a SQL view) and does not need to specify the query logic or database navigation even for the processing of the most complex multi-leg hierarchical structure. The naive user or developer can specify an ad hoc request or easily add an additional relational or XML data item to the SELECT list and it will be retrieved and hierarchically formatted automatically, utilizing the hierarchical semantics in the data. The SQL hierarchical view offers the highest level of data abstraction and reuse, allowing the processing logic to be dynamically tailored to the runtime requirements. These capabilities do not mean that SQL is better than XQuery, in fact XQuery is more powerful, but there is the standard tradeoff – with the increase in control of a computer language, there is a decrease in ease of use. XQuery requires its additional programming control to handle advanced text processing and complex transformations, required in an XML environment. The full extent of these capabilities may be required less in the SQL environment offset by the use of SQL's hierarchical capabilities.

The SQL/XML standards group has done an excellent job specifying and standardizing many useful mappings between XML and SQL that will be used by the standard SQL native XML integration described in this article. The SQL/XML standard also specifies XML-centric functions in SQL for producing XML documents from the standard flat relational data result of the SQL query, which may include input from XML. The desired XML-formatted hierarchical structure is produced by nesting the SQL/XML standard's XMLElement function within itself to control the desired hierarchical structure. As with XQuery, the SQL/XML standard user must know XML and have knowledge of the input and output hierarchical data structures. The XML-centric SQL functions require a programming addition when a new data item is added to the SELECT list for output. Alternatively, the ANSI SQL native XML integration solution described in this article is seamless and transparent, and produces a valid and accurate hierarchically processed result. It can automatically and dynamically publish XML documents without the introduction of XML-centric SQL functions and their limitations described above. This is made possible by seamlessly utilizing standard SQL's significant inherent hierarchical processing capa-

bility, described in Part 1 of this article.

Conclusion

This two-part article has demonstrated how standard ANSI SQL can integrate fully, naturally, and seamlessly with XML by raising SQL processing automatically to a hierarchical processing level, enabling relational data to integrate at a hierarchical level directly with native XML. This was proven not only with examples, but by demonstrating at each stage of SQL processing how it works, from SQL syntax and semantics through the Cartesian product relational engine described in Part 1 of this article. It was also shown that the level of hierarchical support was significant, handling complex multi-legged hierarchical queries intuitively, and joining hierarchical structures easily. This allows SQL to fully utilize the hierarchical semantics in the data and the data structure. By operating at a hierarchical level, the memory and processing efficiencies are greatly increased, and because SQL itself is performing the majority of the integration work, the XML support is very efficient and its footprint is very small, making it excellent for embedded use. All of these features and capabilities support dynamic processing. This ad hoc processing includes powerful parameter-driven query specification in the form of SQL SELECT list specification and WHERE clause filtering that dynamically tailors and optimizes the most complex stored hierarchical views.

In Part 2 of this article, advanced XML processing features that can also be performed by ANSI SQL's standard and inherent hierarchical processing and capabilities were covered. These advanced capabilities include seamless support for shared and duplicate elements in the data structure using the SQL Alias capability; node promotion and structure fragment processing automatically controlled by what data fields are selected on the SQL Select list; and structure transformations using a combination of the structure fragment and Alias capability. Since all the capabilities mentioned in this article inherently exist in ANSI SQL, they operate together in a seamless and unrestricted (orthogonal) fashion. This includes the multi-leg hierarchical data filtering, which automatically operates on the SQL modeled data structure and the full unlimited use of SQL views. This SQL native XML integration operation is naturally standard

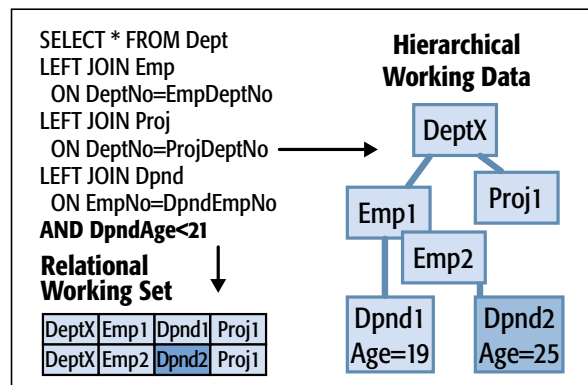


Figure 4 • ON clause path-level data filtering

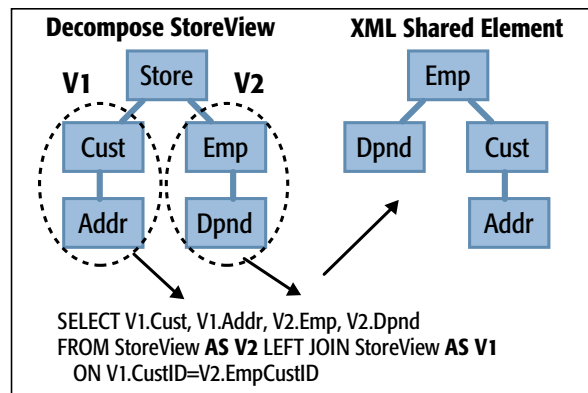


Figure 5 • Structure transformation

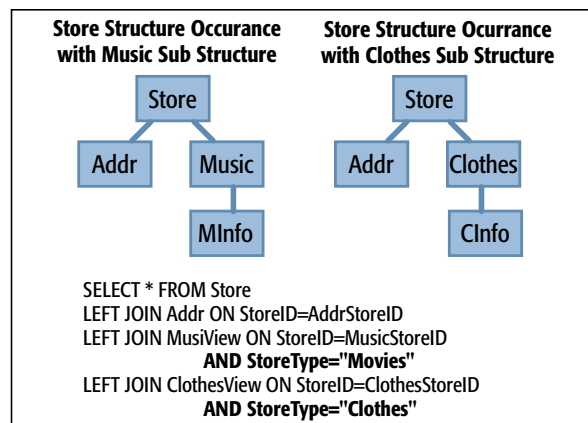
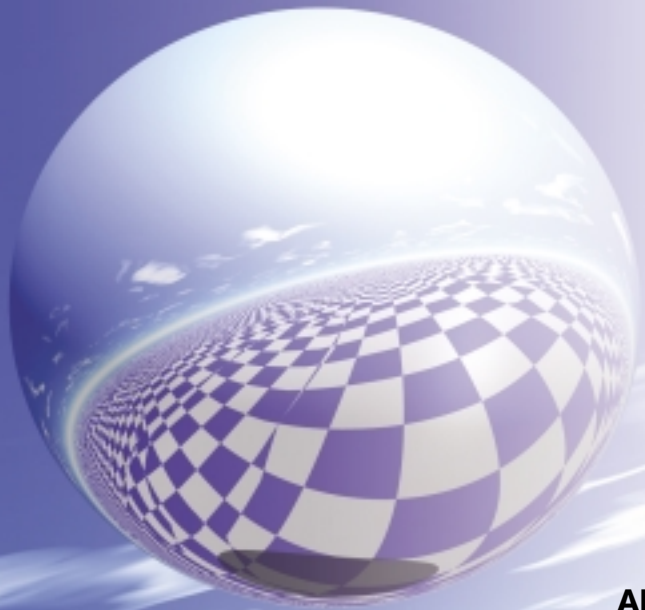


Figure 6 • Variable structures

because it seamlessly and naturally stays within ANSI SQL. It does not require the addition of SQL standardized XML-centric functions, producing a powerful and easy-to-use hierarchical ad hoc query language for XML and other hierarchical forms of data, including SQL hierarchically modeled relational data. For more information on all these topics and additional ANSI SQL-supported XML capabilities, visit www.adatinc.com.

MIKE@ADATINC.COM

4TH Annual International Deve



DEVELOPMENT TECHNOLOGIES EXCHANGE

February 24-26, 2004
Hynes Convention Center, Boston, MA

**ARCHITECTING JAVA, .NET, WEB SERVICES, MX, XML, AND
OPEN SOURCE**



Hynes Convention Center
Boston, MA



PROVIDING DEVELOPERS THE LATEST IN:

- Application Integration
- Desktop Java
- Mobility
- ASP.NET
- VS.NET
- JavaServer Faces
- SOA
- Interoperability
- Rich Internet Applications

Preregister for **FREE** full-day tutorials featuring the hottest Java, .NET, XML, Web Services, and Security issues. (ADVANCE REGISTRATION REQUIRED)

FREE Exhibition Floor (ADVANCE REGISTRATION REQUIRED)

Meet with industry-leading companies showcasing the newest products and services.

- Product Demonstrations Theater
- Opening Night Reception
- Internet Café
- Case Study Forum

Register By
December 19, 2003
SAVE Up
To
\$400



MX

WebSphere

XML

For more information
visit www.sys-con.com/edge
call 201 802-3069
e-mail events@sys-con.com

Over 100 participating companies will display and demonstrate over 300 developer products and solutions.

Over 3,000 C-Level Executives, Systems Integrators, System Architects, Developers, and Project Managers will attend the conference and expo.

Over 100 sessions – training, certifications, seminars, case studies, and panel discussions – promise to deliver real-world benefits, the industry pulse, and proven strategies.

Full-Day Tutorials Targeting

- Java
- .NET
- XML
- Web Services
- Open Source
- Security

Conference program available online!
www.sys-con.com/edge

PLATINUM SPONSOR: **macromedia**

GOLD SPONSORS:

Microsoft



Computer Associates

SILVER SPONSOR:

Novell

OWNED BY
SYS-CON MEDIA ASSOCIATES
PRODUCED BY
SYS-CON EVENTS

LinuxWorld

LinuxWorld.com

JAVA DEVELOPER JOURNAL

meta

JavaWorld

WebServices

LinuxWorld.com

XML

OASIS

SDTimes

Leaf Industry Consortium

PerfectXML

.NET JOURNAL

asp.netPRO

WebSphere

WebLogic

wireless

CF Advisor

ColdFusion

ASP street

HSP STREET

PowerBuilder Journal

MX

MX

Developer Conference & Expo

Attend a **FREE** One Day Security Tutorial Presented By



Strategies for Web Services Security Success

Are you a developer, software architect, IT operator, or security administrator deploying or planning to deploy XML Web services? If so, this technical seminar is designed to arm you with the practical information and best practices you need to securely deploy XML Web services in your environment. Many questions will be addressed, such as: Why do Web services need special security? What standards are being created and what do they address? How do I leverage my existing environment to secure Web services? What are the different architectural and technical approaches to solving the problem? How can I centrally manage security in a decentralized environment?

Course Highlights/Benefits

- Overview of XML Web services security: Why is it important?
- Discussion of various standards (WS-security, SAML, XML-Enc, XML-Sig, XKMS)
- Architectural considerations
- Malicious Web services attacks
- Strategies for securing XML Web services today and in the future
- "Nuts and bolts" demonstrations of security solutions

As part of the tutorial, we will show you how to secure your entire XML environment without adding any additional code.

Seating is limited.

Full-conference attendees will receive priority seating for all tutorials; all other seating is on a first-come, first-served basis.

Attend a **FREE** One Day Tutorial Presented By



.NET—The Smart Client Perspective

For the business world, one of the most exciting promises of the information age is the opportunity to provide employees with access to information and the tools to act on that information whenever and wherever they need to. To a certain extent, that promise has already been achieved. Today, most businesspeople work on PCs that provide access to information, applications, and resources far beyond the boundaries of their local machine.

There are limits, however. Today's Internet model for information and application distribution assumes access to a network connection, but ubiquitous Web connectivity still lies in the future. And some computing tasks require robust functionality that can only be provided efficiently by "rich" client applications that reside on the local computer.

A challenge arises when your organization requires both the flexibility and immediacy that comes with online access to data and applications, and the full functionality of traditional client software.

The answer: smart client software.

This day-long tutorial focuses on developing and deploying smart client applications.

Seating is limited. Full conference attendees will receive priority seating for all tutorials, all other seating is on a first-come, first-served basis.

VISIT www.sys-con.com/edge FOR DETAILS and REGISTER TODAY

The program, including topics and times, is subject to change. Please refer to www.sys-con.com for all updates.



JAVA SESSIONS

■ Aspect-Oriented Programming and Java

This session introduces Aspect-Oriented Programming (AOP) and how it applies to enterprise Java application development, with an emphasis on applications for service-oriented architectures such as Web services. AOP has become a major topic in the future of enterprise Java development. This session will present a conceptual road map, including tangible examples of how AOP works, and provide an understanding of both the potential and challenges of applying AOP in a J2EE context.

■ Squeezing Java

Java is a very powerful language; while it offers the developer a rich array of tools, the fundamentals mustn't be overlooked. Improving your code at the core layer will result in great improvements in efficiency and produce (hopefully) fewer bugs. We'll look at the do's and don'ts of programming and learn lots of hints and tips that will accelerate your Java coding.

■ Enterprise Architecture and Open Source

Use of open source software within the enterprise is gaining momentum. The vast majority of organizations are using some form of open source software in production environments, including Linux, Apache, and JBoss. The enterprise architecture, however, needs to incorporate the best thinking of the industry; this includes not only using open source but contributing to it.

The model in which open source software gets developed has practices that could assist an organization in becoming agile in their software development practices and allow them to develop software faster, with cheaper costs and better quality. In this session, you will learn:

- Two models of development: the cathedral and the bazaar
- The value proposition of using open source
- Harnessing the power of the mob: the value proposition of contributing to open source
- Making the build versus buy decision: additional thoughts

■ J2EE v1.4

Day-to-day work with deadlines makes it difficult to keep abreast of the rapidly evolving landscape of J2EE, especially given the numerous constituent J2EE technologies. J2EE v1.4 is chockful of new services that affect and benefit a wide range of enterprise development tasks. This talk will extract core material from the speaker's new J2EE Developer's Handbook and describe what's embodied in J2EE v1.4. In particular, the new Web services features provided by J2EE v1.4 will be highlighted. The talk will also briefly address those services missing from the current J2EE standards but still needed when building enterprise applications.

■ Apache Axis

Apache Axis is the popular SOAP engine that includes everything you need to start producing Web services. Discover just what Axis is, and how you can utilize the power of this free engine to kick start your Web services.

■ Empowering Java and RSS for Blogging

One of the fastest-growing areas over the last few years is the blogging community. The ease with which you can post and publish information has enabled everyone to become their own publisher. One of the powers of blogs has been the syndication of data via the RSS (XML) protocol. Discover how you can easily produce and consume RSS feeds within your Java applications for wider appeal and hook into JavaBlogs, for example.

■ JUnit/Ant

A defined and easily repeatable process is one of the most necessary but often least-used aspects of good software development. A defined build process ensures that your project's software is built, deployed, and tested identically each time. Without this type of control and predictability, valuable time is often lost chasing down bugs that don't exist or rejecting solutions that were only partially implemented.



A critical measure of the success of software can be found in whether or not it executes successfully. Equally important, however, is whether or not that software does what it was intended to do. JUnit is an open source testing framework that provides a simple means for developers to define their intentions of how their software should work. JUnit then provides test runners that process your intentions and verify that your code performs as intended. The result is software that not only works, but works in the correct way.

Apache's Ant is a powerful scripting tool that enables developers to define and execute routine software development tasks using the simplicity and extensibility of XML. Ant provides a comprehensive mechanism for managing software development projects, including compilation, deployment, testing, and execution. In addition, it is compatible with any IDE or operating system.

■ Next Phase in the Evolution of J2EE

J2EE has been making major inroads in the enterprise space for a number of years. However, it is only with the 1.4 release that we have had uniform and easy access to Web services. Discover how to leverage the new features of J2EE 1.4 and why this release is a significant milestone in the evolution of J2EE.

■ Simplifying J2EE Applications

J2EE is a large, complex specification for server-side, Web-enabled application development. Over the past few years, the presenter has led many teams through the J2EE jungle, trying to steer them away from the hype and keep them focused on delivering rock-solid, end-user applications. This tutorial will discuss a variety of tips, tricks, and lessons that he has learned so you and your teams can develop J2EE applications better, faster, and simpler than before.

WEB SERVICES SESSIONS

■ Exploring the Dark Side

The growing use of services-oriented architectures puts pressure on application developers relying on Web services for key features of their applications. Performance, scalability, and reliability of these components affect the ability of applications to meet service-level agreements, yet can't easily be analyzed as a part of the application when developers have a problem. In fact, the Web service may be on a different software platform than the rest of the application. This session describes how developers can shed light on memory use in Web services written in either .NET or Java, even if they didn't write the code and do use another platform.

■ Web Services Progress Report

Web services have been the buzz for the last couple of years. There have been many technical and some market success stories but the concept remains confusing. New "standards" are proposed on a regular basis, but they overlap one another and seem to form rifts along the same fault lines as previous industry politico-strategic controversies. A group of people from the W3C Web Services Architecture working group have been arguing that many of the ideas coming from the Web services community are antithetical to the principles of the Web itself and are unlikely to ever work on an Internet scale. This presentation provides a progress report on the effort to distinguish Web services architectural principles from the marketing agenda of individual companies.

■ ID, Please. The Case for Giving Web Services an Identity

Without identity management, Web services can be consumed by anyone. The challenge for Web services developers is to provide appropriate access based on the user's identity. As identity management moves into the forefront of technology, directory services will evolve from simple LDAP repositories used for authentication and storage to robust engines that provide identity integration, access management, and policy enforcement. This presentation will discuss how identity management and directory services provide a robust solution for Web services authentication, authorization, and single sign-on.

■ Web Services Orchestration, Management, and Security: Will They Play Together?

Web services orchestration, management, and security are among the principal challenges facing implementers of service-oriented architectures today. There is still much confusion in the IT community about the standards themselves, which are at various stages of maturity. Their relevance to enterprise IT and how they might someday be able to effectively work together is often unclear. This session provides an overview of standards in these three critical areas; and more important, how each affects the other. Attendees will then gain practical knowledge and a deeper understanding of future trends and the need to address certain real-world issues in order to create a more cost-effective and agile IT infrastructure.

■ Service-Oriented Integration: Making the Right Choices To Support The Next-Generation of Integration

Applications are increasingly being developed "built-to-integrate," pro-

viding the ability to easily expose key functionality through commonly defined interfaces. Gartner calls this concept SODA, or service-oriented development of applications. When applied to the ever-present integration challenge, SODA represents a transition to service-oriented integration.

But making the right architectural decisions is absolutely vital to ensuring success with service-oriented integration projects – whether applications were built to integrate or not.

This presentation will examine the leading choices for supporting service-oriented integration: enterprise service buses, integration brokers, and application suite platforms.

■ Government Real-Time Fraud Detection Using Web Services

Government agencies are faced with increasing amounts of data and are challenged to make sense of, and act on, that data in real time. Failure to interpret and execute on data can result in security threats and, potentially, loss of life. Government agencies are increasingly investing in Web services solutions to address their need for real-time access to information.

The Canadian Passport Office is an example of a government agency leveraging Web services to exchange information in real time to combat terrorism and other illicit uses of fraudulently obtained passports. They selected IT consulting firm Pentelar and Sybase, Inc., technologies to electronically authenticate identity document data through the use of Web services and ebXML.

This session will discuss this pilot project and highlight the ebXML capabilities that enable the Canadian Passport Office to address real-time information exchange.

■ WS-CAF: Standardized Web Services Transactions and Composite Applications

The Web Services Composite Application Framework is a collection of three specifications – Web Service Context (WS-CTX), Web Service Coordination Framework (WS-CF), and Web Service Transaction Management (WS-TXM) – designed to solve problems that arise when multiple Web services are used in combination ("composite applications") to support information sharing and transaction processing. As coauthor of the specification, we will discuss how WS-CAF addresses the underlying issues of Web service context propagation and transaction management to expand the scope, usability, and reliability of Web services for business process automation.

■ Securing Web Services: What Can Be Done Today?

Security is listed as one of the main barriers to the adoption of Web services today. With the proliferation of security standards, there is a lot of confusion over which ones are mature enough to use and how they fit together. This session will present the current and emerging security standards for Web services and show how they can fit together architecturally to address various security concerns.

4

XML SESSIONS

■ Universal Business Language

Web service technologies promise to revolutionize electronic business, but global interoperability of business processes cannot occur without the semantic standardization of the messages exchanged in business transactions. This session will describe the OASIS UBL project to create standard XML Schemas for basic business documents, explore the relationship of UBL-based business to traditional EDI, and note the explosive potential of standard markup combined with reliable XML messaging.

■ Real Best Practices for XML Web Services Management and Security

Companies deploying Web services in a meaningful way are increasingly finding they need to address Web services management and security early in the architectural phase. Basic Web services connections are easy to do, but managing the security, performance, scalability, and inevitable changes to the production environment requires some knowledge, expertise, and planning. This session cuts through the hype and outlines real-world mistakes many companies make when deploying Web services and the real best practices from companies that have successfully captured the value of XML Web services. It provides practical advice on how to successfully manage and secure your XML Web services environment.

■ SOA Foundation Components: Building an XML Content Router

One of the fundamental components for any burgeoning SOA will be an XML content router. This session explores the concepts, patterns, and open source software available that facilitate building an XML content routing system. The system can be exposed as a Web service or simply as a stand-alone J2EE component for use in your enterprise. The "restaurant" pattern is introduced as the principal design pattern for building the service, and this pattern's applicability to building generic services is discussed. Applying the router as an XML data integration tool is also discussed, as well as its potential for acting as a service orchestrator.

■ What's New in XSLT 2.0?

XSLT 2.0, which may achieve W3C Recommendation by conference time, offers unparalleled power in conjunction with XPath 2.0 for transforming XML documents. In this engaging, example-rich session, Steve Heckler demonstrates the most important new features of XSLT 2.0, including Sequences, new data types and XML Schema support, regular expressions, multiple document output, grouping, new control-flow operators, and much more. Current and future support for XSLT 2.0 on the Java and .NET platforms will also be discussed. Most examples will use Saxon, but .NET examples will be included if .NET supports XSLT 2.0 by conference time.

■ Using XML Schemas Effectively in WSDL Design

Developers building Web services today are beginning to see the value of using the document-style approach over RPC. Recent experi-

ence shows that to take full advantage of document-style Web services requires a strong knowledge of XML Schemas and related XML standards. This presentation presents a number of important tips and techniques for properly using XML Schemas in the design of a Web services interface (WSDL), including XML-based development tools, binding considerations between XML and underlying objects, WSDL reusability through XML Schemas, and XML Schema naming best practices.

■ XML: A Manager's Guide

As more and more IT projects utilize XML and its derivatives as fundamental technologies, it is key for today's manager to be aware of the various ingredients of XML. The objective of the session is to provide an essential introduction around XML from a manager's perspective. From core XML processing; transformation; metadata definition and schemas; applications in Web, wireless, and speech applications; Web services; industry-standard vocabularies; and more, this session provides a comprehensive review of the various technologies related to XML.

■ Using Rules to Clean Up XML

Garbage in, garbage out – it's an axiom that applies to many aspects of enterprise development, but none more so than building reliable and robust Web applications and integration projects with XML. Since its inception, XML has been seen as the cure-all for problems related to Web applications and integration projects. However, poorly written XML can slow down an integration project, or worse, cause the integration project to collapse. The key to successfully using XML in an integration project is to first understand the inefficiencies that may cause poorly written XML, and then apply a rule-based system that establishes policies to follow.

EXPO HALL

TUESDAY, FEBRUARY 24,

11:00AM-4:00PM, Welcome Reception 4:00PM

WEDNESDAY FEBRUARY 25,

11:00AM-4:00PM

■ XForms: Simplifying the Development of Transactional Web Forms

XForms is a W3C specification that specifies a declarative language for solving a common requirement for advanced user interaction, data validation, and XML processing. XForms is designed to be integrated into XHTML, but is not restricted to being a part of that language alone. It can be integrated into any suitable markup language. This session will give an introduction to XForms and explain how XForms fits in the client tier of the J2EE application architecture. In addition, it will cover the benefits of XForms and why it is a perfect fit for interacting with J2EE and Web services. A demonstration of XForms in a J2EE environment, using an XForms-compliant browser and a sample application, will further illustrate the advantages.

VISIT www.sys-con.com/edge FOR DETAILS and REGISTER TODAY

The program, including topics and times, is subject to change. Please refer to www.sys-con.com for all updates.

.NET SESSIONS

■ .NET Compact Framework Performance Tips and Tricks

Learn the techniques that can be used to increase the responsiveness of user interface and network operations for users of applications built on the .NET Compact Framework. Look under the covers at advances and changes in the "Whidbey" release to significantly improve performance. Get a general overview of how the .NET Compact Framework works under the hood at runtime, with specific focus on performance implications. Then we will cover general user interface tips to increase performance. Explore how asynchronous infrastructure, such as threading, in the .NET Compact Framework can be leveraged to optimize both user interface and network operations. Learn about the architectural guidelines for creating applications that perform well under frequently changing network conditions.



■ Best Practices and Techniques for Building Secure ASP.NET Applications

When the enterprise depends on your application, careful attention to security is essential. This session provides specific recommendations to follow when developing secure ASP.NET Web applications and services, and focuses on the details of configuring IIS for security. Understand how to use authentication, authorization, threat modeling, configuration settings, and secure database access to create secure systems and learn common coding techniques for storing secrets, error handling, data validation, and code access security.

■ Using the Enterprise Instrumentation Framework

The Microsoft .NET Framework 1.1 and Windows Server 2003 offer a number of new features to help developers instrument their code. In this session we'll learn about the challenges facing application management in today's distributed world. We will examine the new unified instrumentation API in the Enterprise Instrumentation Framework (EIF), including the new Windows Event Trace available in Windows Server 2003, configurable at-source event filtering, and how request-based event tracing using EIF allows you to put a request context around the trace messages that map to a business process flow in your application. We will also discuss the benefits of using EIF in your application for both the developer and the application administrator.

■ .NET Framework: Exploring What's New in the Base Class Library for "Whidbey"

The base classes serve as the essential libraries for any developer. Continued evolution of the base classes provides numerous benefits,

including the ability to develop more reliable, faster solutions, easier-to-write code, and more solutions entirely in managed code. Take a look at the many features that are a part of that evolution, including features in IO, event-logging, and various features in System.Collections.Generic classes and interfaces.

■ Microsoft Office 2003: A Solutions Platform

For all developers who would like to integrate custom business solutions with Microsoft Office products, this session will introduce you to the expanded developer features that have been included in the newest version of Microsoft Office. Come explore new XML-based programmability in everything from Word 2003 and Excel 2003 to FrontPage 2003 and SharePoint. Build powerful, modular solutions with Web services. Learn about InfoPath 2003 support for XML standards. Discover how to use the Microsoft Visual Studio Tools for the Microsoft Office System to automate and extend Microsoft Office Word 2003 and Microsoft Office Excel 2003 using Visual Basic .NET and Visual C# .NET. More than ever, Office has a solution for you.

■ BizTalk Server 2004 Technical Drilldown

Biztalk Server 2004 is designed to enhance Enterprise Application Integration (EAI), Business Process Automation (BPA), and Information Worker Integration. Join us for a technical drilldown into the new features and toolsets available.

■ Moving your Architecture to .NET

This session's emphasis is on how to migrate existing business components from VB6 COM objects to VB.NET assembly components. We'll spend time discovering how to best move different tiers of a multi-tiered application from COM to .NET, as well as effective strategies on how to wrap existing COM components for interoperability. We'll also examine best practices for moving your application from a COM-based architecture to a .NET-based architecture.

Who Should Attend

- Software Developers
- Software Engineers
- CTOs
- CIOs
- Development Managers
- Application Developers
- IT Directors
- Technical Directors
- Analysts
- Consultants
- Programmers
- IT Managers
- Technical Architects
- Team Leaders
- Software Consultants



VISIT www.sys-con.com/edge FOR DETAILS and REGISTER TODAY 

The program, including topics and times, is subject to change. Please refer to www.sys-con.com for all updates.

MX SESSIONS

■ Enterprise Infrastructure for Rich Internet Applications

Learn how Macromedia's technology initiative "Flex" fits seamlessly into today's new service-oriented architectures (SOA). We'll cover design patterns for rich clients, accessing Web services, and securing your Flex application.

■ ColdFusion Components from the Ground Up

ColdFusion Components (CFCs) are considered to be the most important enhancement to the CFML language since it was created some eight years ago. CFCs combine the power of objects with the simplicity of CFML, and in this session, you'll discover that not only are they incredibly powerful, they're also remarkably easy to master.

■ Code-Based Rich Internet Applications

Learn how to use Macromedia's technology initiative "Flex" to create rich Internet applications. This session will cover using components, layouts, and managers to build user interfaces, as well as using Flex's XML-based language to create and manipulate client-side data models.

■ Tips and Tricks for Writing and Using CFCs

ColdFusion Components are simple to write and simpler to use. But that simplicity hides a series of powerful features and technologies that you can (and should) take advantage of. In this session, you'll learn how to use (and how to not use) inheritance, "super," persistence, constructors, and more.

■ Leveraging Web Services

Web services technology is changing the way we think about designing and building applications. Come and learn what all the fuss is about, find out exactly which problems Web services solve, see Web services created and used, and even discover how Web services expose the world of .NET.

■ Building an RIA with Macromedia Flash and ColdFusion Web Services

Learn about the quickest and easiest way to build rich Internet applications using Macromedia Flash with connections to Web services built in ColdFusion.

■ ColdFusion Components

ColdFusion Components combine the power of objects with the simplicity of CFML. This is the way object-based development was intended to be, and in this session, you'll learn about this combination first hand. Starting with a simple data-driven application, you'll gradually convert it into a highly scalable and manageable multitier application, and in the process, will be amazed at just how easy ColdFusion makes this process.

■ Rapidly Build Web Services Applications with ColdFusion and Studio MX

The last year has shown that Web services are not just another passing fad and their promise of platform-independent distributed applications has been realized. Compared to other application server platforms, ColdFusion makes creating Web services easy. This session covers how to create a ColdFusion Component (CFC) in Dreamweaver, as well as how to expose that CFC as a Web service by just toggling one attribute of the CFC. That's right: in ColdFusion, it is just that easy.

■ Using Macromedia Flash with Web Services

Web services, a technology that allows developers to execute remote procedures, is emerging as a revolutionary tool for Web application development. Macromedia Flash MX 2004 Professional is a powerful tool for building applications that consume Web services built in any technology, including Macromedia ColdFusion, Java, ASP.NET, and PHP. In this session you will explore the visionary computing model that Web services represent as you use Macromedia Flash components to develop a Web service-based application. You will learn how to discover Web services, work with data and UI components, perform data binding, examine security issues, and aggregate multiple Web services into a cutting-edge Web service consumer.

■ Using Web Services with ColdFusion

You're a ColdFusion user and want to be able to take advantage of Web services? You're in luck. No other language or platform makes Web services consumption as painless as ColdFusion does. In this hands-on session, you'll experience Web services for yourself by building a complete application around some of the most popular Web services available today.



Sheraton Boston Hotel
Official Hotel of Edge 2004 East

REGISTRATION FORM

Edge 2004 East Development Technologies Exchange

CONFERENCE: Feb. 24 – 26, 2004 EXPO: Feb. 24 – 25, 2004

Hynes Convention Center, Boston, MA

THREE WAYS TO REGISTER FOR CONFERENCE

- 1) **On the Web:** Credit Cards or "Bill Me." Please make checks payable to SYS-CON Events.
- 2) **By Fax:** Credit Cards or "Bill Me" 201-782-9651
- 3) **By Mail:** 135 Chestnut Ridge Road, Montvale, New Jersey 07645, Attention: Registration

Please note: Registrations are not confirmed until payment is received.

Please complete sections 1, 2, 3 and 4

1 YOUR INFORMATION (Please Print) ☐ Mr. ☐ Ms.

First Name _____ Last Name _____
Title _____
Company _____
Street _____
Mail Stop _____
City _____
State _____ Zip _____ Country _____
Phone _____
Fax _____ E-Mail _____

2 PAYMENT METHOD: (Payment in full due with registration)

☐ Check or Money Order Enclosed (Registration confirmed upon receipt of payment)
Check # _____ Amount of Check \$ _____
Charge my ☐ Visa ☐ MasterCard ☐ American Express ☐ Discover
Name on card _____
Card # _____ Exp. Date _____
Signature _____
Billing Address (if different from mailing address) _____

3 PLEASE INDICATE

YOUR CONFERENCE CHOICE

Total Registration fee \$ _____

	Before 12/19/04	Before 1/24/04	Before 2/21/04	Onsite
<input type="checkbox"/> GP: Gold Passport Three-Day Conference Good for all three days of the conference program, including keynotes, all conference sessions & one tutorial	\$1,395.00	\$1,495.00	\$1,695.00	\$1,795.00
<input type="checkbox"/> 3D: Three Day Conference Good for all three days of the conference program, including keynotes, all conference sessions (excludes tutorials).	\$1,295.00	\$1,395.00	\$1,495.00	\$1,695.00
<input type="checkbox"/> 2D: Two Day Conference Good for two days of the conference program, including keynotes, all conference sessions (excludes tutorials).	\$995.00	\$1,195.00	\$1,295.00	\$1,395.00
<input type="checkbox"/> 1D: One Day Conference Good for one day of the conference program, including keynotes, all conference sessions (excludes tutorials).	\$595.00	\$695.00	\$695.00	\$795.00
<input type="checkbox"/> VIP PASS FREE with preregistration Select one: <input type="checkbox"/> FREE - The Smart Client Perspective (Feb. 24) <input type="checkbox"/> FREE - Strategies for Web Services Security Success (Feb. 25)	FREE	FREE	FREE	\$99.00
<input type="checkbox"/> EO: Expo Hall Only Includes keynotes, expo presentations, and workshops.	FREE	FREE	FREE	\$50.00

SPECIAL DISCOUNTS AVAILABLE

Take advantage of the Early Bird and Preregistration values available right now, or save even more with a group of 5 or more. For special group discounts contact Michael Lynch at mike@sys-con.com, or by phone at (201) 802-3058.



If you require special assistance covered under the Americans with Disabilities Act, please call 201-802-3058 by February 10, 2004.

CANCELLATIONS, SUBSTITUTIONS, REFUNDS
Fax written request to SYS-CON Registration 201-782-9651. Requests for refunds received prior to January 16, 2004 will be honored less a 10% handling charge; requests received after January 16, 2004 and before February 6,

2004, will be honored less a 20% handling charge. No requests for refunds will be honored after February 6, 2004. Requests for substitutions must be made in writing prior to February 20, 2004. No one under 18 is permitted to attend. No warranties are made regarding the content of sessions or materials.

Speakers, sessions, and schedule are subject to change without prior notice.

No solicitation by anyone other than official exhibitors, sponsors or marketing partners is permitted. Such behavior is cause for expulsion without refund.

4

A. Your Job Title

- ☐ CTO, CIO, VP, Chief Architect
- ☐ Software Development Director/Manager/Evangelist
- ☐ IT Director/Manager
- ☐ Project Manager/Project Leader/Group Leader
- ☐ Software Architect/Systems Analyst
- ☐ Application Programmer/Evangelist
- ☐ Database Administrator/Programmer
- ☐ Software Developer/Systems Integrator/Consultant
- ☐ Web Programmer
- ☐ CEO/COO/President/Chairman/Owner/Partner
- ☐ VP/Director/Manager Marketing, Sales
- ☐ VP/Director/Manager of Product Development
- ☐ General Division Manager/Department Manager
- ☐ Other (please specify) _____

B. Business/Industry

- ☐ Computer Software
- ☐ Computer Hardware and Electronics
- ☐ Computer Networking & Telecommunications
- ☐ Internet/Web/E-commerce
- ☐ Consulting & Systems Integrator
- ☐ Financial Services
- ☐ Manufacturing
- ☐ Wholesale/Retail/Distribution
- ☐ Transportation
- ☐ Travel/Hospitality
- ☐ Government/Military/Aerospace
- ☐ Health Care/Medical
- ☐ Insurance/Legal
- ☐ Education
- ☐ Utilities
- ☐ Architecture/Construction/Real Estate
- ☐ Agriculture
- ☐ Nonprofit/Religious
- ☐ Other (please specify) _____

C. Total number of employees at your location and entire organization (check all that apply):

	Location	Company
10,000 or more	01 <input type="checkbox"/>	01 <input type="checkbox"/>
5,000 – 9,999	02 <input type="checkbox"/>	02 <input type="checkbox"/>
1,000 – 4,999	03 <input type="checkbox"/>	03 <input type="checkbox"/>
500 – 999	04 <input type="checkbox"/>	04 <input type="checkbox"/>
100 – 499	05 <input type="checkbox"/>	05 <input type="checkbox"/>
100 or less	06 <input type="checkbox"/>	06 <input type="checkbox"/>

D. Please indicate the value of communications and computer products and services that you recommend, buy, specify, or approve over the course of one year:

- ☐ \$10 million or more
- ☐ \$1 million – \$9.9 million
- ☐ \$500,000 – \$999,999
- ☐ \$100,000 – \$499,999
- ☐ \$10,000 – \$99,999
- ☐ Less than \$10,000
- ☐ Don't know

E. What is your company's gross annual revenue?

- ☐ \$10 billion or more
- ☐ \$1 billion – \$9.9 billion
- ☐ \$100 million – \$999 million
- ☐ \$10 million – \$99.9 million
- ☐ \$1 million – \$9.9 million
- ☐ Less than \$1 million
- ☐ Don't know

F. Do you recommend, specify, evaluate, approve or purchase wireless products or services for your organization?

01 ☐ Yes 02 ☐ No

G. Which of the following products, services, and/or technologies do you currently approve, specify or recommend the purchase of?

- ☐ Application Servers
- ☐ Web Servers
- ☐ Server-Side Hardware
- ☐ Client-Side Hardware
- ☐ Wireless Device Hardware
- ☐ Databases
- ☐ Java IDEs
- ☐ Class Libraries
- ☐ Software Testing Tools
- ☐ Web Testing Tools
- ☐ Modeling Tools
- ☐ Team Development Tools
- ☐ Installation Tools
- ☐ Frameworks
- ☐ Database Access Tools/JDBC Devices
- ☐ Application Integration Tools
- ☐ Enterprise Development Tool Suites
- ☐ Messaging Tools
- ☐ Reporting Tools
- ☐ Debugging Tools
- ☐ Virtual Machines
- ☐ Wireless Development Tools
- ☐ XML Tools
- ☐ Web Services Development Toolkits
- ☐ Professional Training Services
- ☐ Other [Please Specify] _____

SYS-CON Events, Inc., and SYS-CON Media make no warranties regarding content, speakers, or attendance. The opinions of speakers, exhibitors, and sponsors do not reflect the opinion of SYS-CON Events and SYS-CON Media, and no endorsement of speakers, exhibitors, companies, products, or sponsors is implied.



Finding the Fit for XSLT

Filling a hole in the puzzle

Although a number of standards exist for information interchange and process definition, industry standards have yet to emerge for defining common integration server and B2B integration server services such as routing, rules processing, and transformation. In the absence of such standards, individual vendors have created proprietary approaches to these basic information-processing services. As a result, we are confronted with features that are not interchangeable, require specialized training, and do not provide a common framework of services.

Even as we begin to implement standards such as XML as mechanisms to manage information interchange, we are also looking to create standards that support information processing within the middleware. These standards will define services common to most integration servers and to B2B integration servers, including rules and transformation.

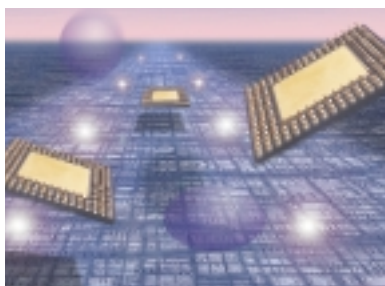
XSLT seeks to fill the need for a standard approach to both rules and transformation processing. Like XML, XSLT could become the preferred standard mechanism for transforming content and application semantics as information moves from application to application and business to business.

The power of XSLT resides in its simplicity, tight integration with XML, and the completeness of the standard. Although it does not provide every type of out-of-the-box transformation service currently found in most integration servers and B2B integration servers, XSLT provides the infrastructure to create such services through an extensible and declarative programming language.

XSLT has been covered in great technical detail in *XML-Journal*, so we

won't dwell on the details here. However, I can say that XSLT is a language designed to transform one XML document into another, changing both its schema and content in the process. At its most primitive, XSLT is a text-processing system that enables the programmer to transform XML documents, or, if required, generate other standard markup languages such as HTML (or any text, for that matter). The transformation capability that XSLT offers is a fundamental service required within almost all application integration problem domains; however, it is indeed redundant to other proprietary transformation services already existing as features of modern application integration technology.

XSLT is the preferred method of con-



verting data structured within XML. You can leverage XSLT for the following:

- Extracting data
- Validating data
- Persisting data
- Converting attributes to elements
- Converting elements to attributes
- Changing the metadata and content of an incoming XML document to create a new outgoing XML document

The Value of XSLT

It is important to remember that XML documents are like messages. And

because each application has its own unique set of application semantics, documents moving from application to application need to be transformed. Both data structure and content must be semantically correct in order to load into the target application. If the data is not in the proper format, the update operation is likely to fail.

Although XSLT has tremendous promise, a huge disconnect remains between what XSLT provides and what middleware needs to offer in terms of transformation, the primary differences being:

- Support for complex transformations
- Efficiency during transformation processing
- The need for programming

The state of the technology of most integration servers, supporting complex but valuable information transport, may indeed be a better fit for binary messaging, which is more efficient than text-based messaging and is easier to manage and process. However, the tradeoff is that binary messaging does require that specialized systems (such as message-oriented middleware) manage it, and it is not as easily managed by external systems as text-based messaging (e.g., XML).

Because that is the case, to process messages using XSLT the messages must first be transformed into XML text (or any text) for XSLT transformation and then be transformed back into a binary message. Efficient? Clearly not. Even as a few integration servers and application servers look at XSLT as their standard mechanism for transformation, it is apparent that building text processing into existing binary messaging systems will be difficult.

AUTHOR BIO

David S. Linthicum is the former CTO of Mercator, and author of the ground-breaking book *Enterprise Application Integration*. His latest book is *Next Generation Application Integration*.



The **Leading Magazine** for Enterprise and IT Management

LinuxWorld Magazine

There is no escaping the penetration of Linux into the corporate world. Traditional models are being turned on their head as the open-for-everyone Linux bandwagon rolls forward.

Linux is an operating system that is traditionally held in the highest esteem by the hardcore or geek developers of the world. With its roots firmly seeded in the open-source model, Linux is very much born from the "if it's broke, then fix it yourself" attitude.

Major corporations including IBM, Oracle, Sun, and Dell have all committed significant resources and money to ensure their strategy for the future involves Linux. Linux has arrived at the boardroom.

Yet until now, no title has existed that explicitly addresses this new hunger for information from the corporate arena. *LinuxWorld Magazine* is aimed squarely at providing this group with the knowledge and background necessary to make decisions to utilize the Linux operating system.

Look for all the strategic information required to better inform the community on how powerful an alternative Linux can be. *LinuxWorld Magazine* does not feature low-level code snippets but focuses instead on the higher logistical level, providing advice on hardware, to software, through to the recruiting of trained personnel required to successfully deploy a Linux-based solution. Each month presents a different focus, allowing a detailed analysis of all the components that make up the greater Linux landscape.

Regular features include:

Advice on Linux Infrastructure

Detailed Software Reviews

Migration Advice

Hardware Advice

CEO Guest Editorials

Recruiting/Certification Advice

Latest News That Matters

Case Studies

SAVE 30% OFF!

REGULAR ANNUAL COVER PRICE \$71.76

YOU PAY ONLY

\$49⁹⁹

12 ISSUES/YR

*OFFER SUBJECT TO CHANGE WITHOUT NOTICE

SUBSCRIBE TODAY!

WWW.SYS-CON.COM

OR CALL

1-888-303-5282

FOR ADVERTISING INFORMATION:

CALL 201 802.3020 OR

VISIT WWW.SYS-CON.COM

XSLT will fit into some middleware products and not others. XSLT almost certainly will succeed as a standard transformation mechanism in products that already process information as raw text or XML. If XSLT continues to pick up speed, other middleware vendors will inevitably follow the crowd. However, those of you moving from advanced proprietary application integration middleware to XSLT will notice the lack of features right away, and may indeed pass on XSLT due to the comparative limitations.

Those moving from the more primitive mechanisms for transformation to XSLT may have a much better opinion. XSLT provides several advantages over SAX and DOM. Its design is based on the fact that most transformation programs use the same design patterns, and therefore can be automated using a higher-level, declarative language. (Stating that the XSLT language is declarative means that it describes the transformation behavior rather than a sequence of instructions necessary to perform the transformation. In other words, XSLT describes the transformation and then leverages the XSL processors to carry out the deed.) Moreover, when XSLT is used,

the requirements of transformation can be expressed as a grouping of rules that define what output should be created when a particular pattern is encountered.

XSLT does not operate directly on the XML text. Instead, it relies on a parser (DOM or SAX compliant) to convert values into an object tree for processing. It uses this tree to manipulate the structure in memory. XSLT enables the user to take advantage of the native language to navigate around the node tree, select nodes, and alter the nodes as the transformation requires.

XSLT and Application Integration

XSLT is not the solution to every application integration requirement. It is, however, an important piece in the puzzle. XSLT's great potential lies in its ability to finally provide a standard transformation mechanism that everyone can agree upon, one that does not require application integration architects to relearn technology as they move from vendor to vendor.

There are several other advantages to using XSLT for transformation, including the following:

- A common language for transforming application semantics as text or XML moves between applications.
- A common standard for representing transformation behaviors
- A common input and output message/document structure
- Backing from most major B2B and intracompany application integration technology vendors and consultants

However, there are limitations to consider as well, including the following:

- Complex transformations are difficult.
- Programming is almost always a requirement.
- The slow emergence of vendor support because of the limitations of existing technologies (e.g., moving from binary messaging to XML text).
- The technical limitations, including performance and security, of using text, and only text.
- The fact that not all standards make it, and XSLT could lose momentum and thus lose wide support, like so many other standards in the past.

LINTHICUM@ATT.NET



FEATURE

More Than Just Security

From just-in-time integration to Web services



PROCESS CONTROL

Monitoring Air Pollution in Real Time Using XML

A successful application in Tuzla Canton, Bosnia



GRAPHML

Tree Structured Data and XML

A comprehensive and easy-to-use file format for graphs



APACHE MAVEN

Evolving Beyond Ant

Using Maven to manage your Java project

DON'T MISS XML-J JANUARY

REAL-WORLD SOLUTIONS



developerWorks™

See where developers gain insight.
See where developers find answers.
See where developers get ahead.

Can you see it?

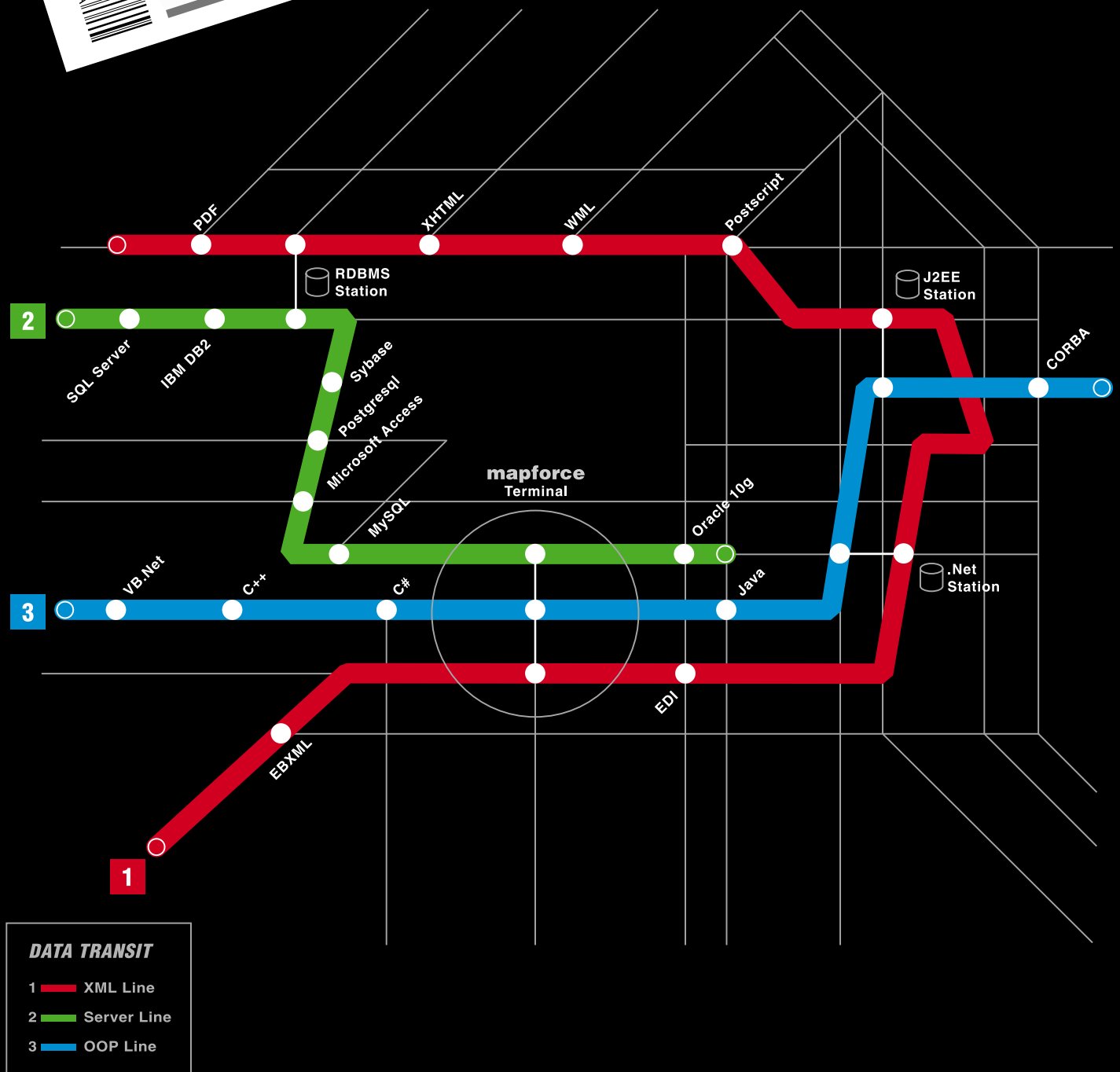
Every day, IBM developerWorks™ helps thousands of developers. On demand. Need tips, tools, tutorials? Get 'em. Want demos and downloads? We've got thousands. All ready to help you work and excel in Java, Linux®, XML, Web services, emerging technologies, and IBM software products and services. Regardless of your current tool brands or platform, developerWorks works. See it work. Start developing on demand applications today with the newly updated developerWorks Toolbox subscription. Register now at ibm.com/developerWorks/toolbox/seeit @ **business on demand™ software**

IBM®

IBM, developerWorks, the e-business logo and e-business on demand are registered trademarks or trademarks of International Business Machines Corporation in the United States and/or other countries. Linux is a registered trademark of Linus Torvalds. Other company, product and service names may be trademarks or service marks of others. ©2003 IBM Corporation. All rights reserved.



Next Stop, Mapforce 2004!



Introducing Mapforce 2004! A powerful new XML mapping tool from Altova, producer of the industry standard XML Development Environment, XMLSPY. Mapforce is a visual data mapping tool, which auto-generates custom mapping code in multiple output languages such as XSLT, C++, C#, and Java, to enable programmatic XML-to-XML or XML-to-Database data transformations. Mapforce 2004 is the hassle-free transit for moving data from one format to another, enabling efficient information reuse. **Download a free trial now!**

ALTOVA®

www.altova.com

All other trademarks are the property of their respective owners.