

XML JOURNAL

The World's Leading XML Resource

Volume: 3 Issue:4

XML-JOURNAL.COM

XMLEDGE
conference & expo

Page 33 **JUNE 24-27**
NEW YORK, NY

- Five Information-Packed Conference Tracks
- The Largest Gathering of Thought Leaders
- The Leading XML Expo in the World

REGISTER NOW!

FROM THE EDITOR

Got XML?

by Ajit Sagar pg. 5

INDUSTRY COMMENTARY

XML IDEs vs Classic IDEs: Competition or Synergy?

by Alexander Falk pg. 7

XML NEWS

pg. 60

AND NOW...

XML! Woo-ha-ha-ha-ha!

by Tod Emko pg. 62

SYS-CON
MEDIA

Deploying WEB SERVICES on WEBSphere

Written by Ron Ben-Natan

**A REAL-LIFE
EXAMPLE,
USING THE
TOOLS AND
SERVICES
PROVIDED**

10

XML & Web Services: .Net Web Services: The 'Three I' Monster

Remember Inform, Inquire, and Invoke

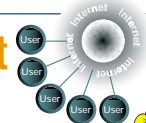


David Weller

16

Web Services: XML-Beyond Transport

XML is a key enabling technology to execute new business models



Kevin Huck

22

Xmatters: Middle-Tier Data Management

Dealing with the middle tiers in the XDBMS architecture can be confusing



Coco Jaenicke

28

XML Feature: ADO.NET & ASP.NET

...Here's how you can use them to build XML applications



Steven Heckler

30

Contrary Opinion: A Pragmatic Convergence of ebXML and Web Services

Is this a second chance for the industry?

John Ogilvie

36

XML Feature: XML Messaging with JAXM

PART 2 Send and receive messages asynchronously and use JAXM JSP tags



Mike Jasnowski

40

Wireless XML: XML Without Wires

PART 2 of 2

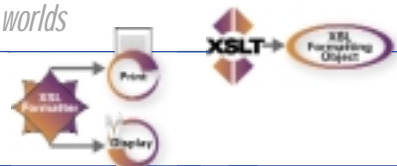
XML - Relevant to both the wireless and the wired worlds

Mark O'Neill

48

XSLT Tutorial: Got XSLT?

How to transform XML to PDF using FOP



Shouki Souri

50

Sonic Software Corporation

www.sonicsoftware.com

Sonic Software Corporation

www.sonicsoftware.com

Macromedia

www.macromedia.com/downloads

XML JOURNAL

EDITORIAL ADVISORY BOARD

JOHN EVDEMON jevdemon@vitria.com
GRAHAM GLASS graham@theminelectric.com
COCO JAENICKE cjaenicke@mediaone.net
SEAN MCGRATH sean.mcgrath@propylon.com
JP MORGENTHAU jpmorgenthal@kimbo.com
SIMEON SIMEONOV simeons@macromedia.com

DEPARTMENT EDITORS

EDITOR-IN-CHIEF: Ajit Sagar
EDITORIAL DIRECTOR: Jeremy Geelan
E-BUSINESS EDITOR: Israel Hilerio
JAVA TECHNOLOGY EDITOR: David Johnson
PRODUCT REVIEW EDITOR: Jim Milbery
WEB SERVICES EDITOR: Graham Glass
EXECUTIVE EDITOR: M'lou Pinkham
MANAGING EDITOR: Cheryl Van Sise
EDITOR: Nancy Valentine
ASSOCIATE EDITORS: Jamie Matusow
Gail Schultz
Jean Cassidy

WRITERS IN THIS ISSUE

Ron Ben-Natan, Tod Emko, Alexander Falk,
Steve Heckler, Kevin Huck, Coco Jaenicke,
Mike Jasnowski, John Aloysius Ogilvie, Mark O'Neill,
Ajit Sagar, Shouki Sourl, David Weller

SUBSCRIPTIONS

SUBSCRIBE@SYS-CON.COM

For subscriptions and requests for bulk orders,
please send your letters to Subscription Department

Cover Price: \$6.99/issue

Domestic: \$77.99/yr (12 issues)

Canada/Mexico: \$99.99/yr

all other countries \$129.99/yr

(U.S. Banks or Money Orders)

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.

©COPYRIGHT

Copyright © 2002 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.



SYS-CON
MEDIA



WRITTEN BY AJIT SAGAR EDITOR-IN-CHIEF

from
the
editor

GotXML?

A large part of business application development deals with abstracting the software components and services that enable the implementation of industry domain-specific business processes into a design environment that is used to model the design of the business process itself. One of the biggest challenges in achieving this goal is to bridge the gaps between disparate environments that combine to provide a comprehensive solution. Gaps exist at different levels and on different levels and manifest themselves in different tiers of the distributed enterprise.

A business application deals primarily with two types of process flows. The user at the uppermost echelon of an *n*-tier application gets a view of the application through a series of "synchronous" or procedural flows. In a Web application these are implemented via a series of Web pages that interactively guide the user through the presentation flow. At the other end of the spectrum are the "asynchronous" flows that execute calls into the back-office systems to perform business tasks. EAI is the glue that enables the application to execute these asynchronous flows. Designing the complete end-to-end solution is a paradoxical problem. How can a design environment support the ability to model both these paradigms simultaneously, yet offer decoupled environments so the actual implementation of software components that drive these flows can be done by different development teams with different skill sets? The problem is magnified enormously when the same environment needs to support the needs of a variety of industrial domains.

As we know, XML provides the ability to extract and exchange information between the different tiers in the form of a common format. An XML-based application couples the ability of XML to separate data from the processes that surround it with the enterprise-level transport layer provided by the Internet. Hence we get the means to construct the "glue." Data access from the legacy systems may be via SQL, messaging, or a variety of other mechanisms. Data presentation to the user interface tier may be done via HTTP or proprietary, non-Web protocols. However, as long as the data format for the knowledge exchanged between the layers is well defined and universal, there is some hope for achieving the nirvana of a common modeling environment for enterprise applications.

Today's design environments are moving toward supporting this type of enterprise modeling. The IDEs (integrated development environments) support modeling enterprise components from business requirements by supporting modeling paradigms like UML. And workflow engines that are built on application server frameworks allow us to model the business process flows that constitute a company's business. But there is still a gap between the environments available for modeling the synchronous versus asynchronous flows. IDEs and integration environments have come a long way since their inception and offer many features for automating the generation of software components and legacy integration modules. The advent of Web services has definitely helped in creating the next stage of enterprise application development. Today's environments allow application developers to convert their programming environment-specific code to Web service components with very little effort.

However, modeling a business process as a suite of Web services, generating the software components to implement the process, and then generating the adapters to connect to different EISs (Enterprise Information Systems) still requires application development in unconnected environments. The next wave of consolidation for business process modeling, presentation design, middle-tier software components, and legacy adapters will address this problem. Before XML became widely accepted as the glue to achieve this, such problems were insurmountable. However, with XML and related technologies, it is only a matter of time before we achieve the goal of a common development environment for enterprise business applications. ☒

AJIT @ SYS-CON.COM

AUTHOR BIO

Ajit Sagar is the founding editor and editor-in-chief of XML-J as well as the J2EE editor of Java Developer's Journal. A lead architect with a B2B software solutions firm based in Dallas, he is an expert in Java, Web, and XML technologies.

Sprint PCS

<http://developer.sprintpcs.com>

PUBLISHER, PRESIDENT, and CEO
Fuat A. Kircaali fuat@sys-con.com

BUSINESS DEVELOPMENT
VP, Business Development
Grisha Davida grisha@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
Advertising Account Manager
Megan Ring megan@sys-con.com
Associate Sales Managers
Carrie Gebert carrie@sys-con.com
Kristin Kuhnle kristin@sys-con.com
Alisa Catalano alisa@sys-con.com
Leah Hiltman leah@sys-con.com

EDITORIAL
Executive Editor
M'lou Pinkham mpinkham@sys-con.com
Managing Editor
Cheryl Van Sise cheryl@sys-con.com
Editor
Nancy Valentine nancy@sys-con.com
Associate Editors
Jamie Matusow jamie@sys-con.com
Gail Schultz gail@sys-con.com
Jean Cassidy jean@sys-con.com
Editorial Intern
Stephanie Williams stephanie@sys-con.com

PRODUCTION
VP, Production & Design
Jim Morgan jim@sys-con.com
Art Director
Alex Botero alex@sys-con.com
Associate Art Directors
Cathryn Burak cathyb@sys-con.com
Louis F. Cuffari louis@sys-con.com
Assistant Art Directors
Richard Silverberg richards@sys-con.com
Aarathi Venkataraman aarathi@sys-con.com
Tami Beatty tami@sys-con.com

SYS-CON EVENTS
VP, Events
Cathy Walters cathyw@sys-con.com
Conference Manager
Michael Lynch mike@sys-con.com
Sales Executives, Exhibits
Michael Pesick michael@sys-con.com
Richard Anderson richarda@sys-con.com

CUSTOMER RELATIONS / JDJ STORE
Manager, Customer Relations / JDJ Store
Anthony D. Spitzer tony@sys-con.com

WEB SERVICES
Webmaster
Robert Diamond robert@sys-con.com
Web Designers
Stephen Kilmurray stephen@sys-con.com
Christopher Croce chris@sys-con.com
William Darron bill@sys-con.com
Content Editor
Lin Goetz lin@sys-con.com

ACCOUNTING
Chief Financial Officer
Bruce Kanner bruce@sys-con.com
Assistant Controller
Judith Calnan judith@sys-con.com
Accounts Receivable
Jan Braidech jan@sys-con.com
Accounts Payable
Joan LaRose joan@sys-con.com
Accounting Clerk
Betty White betty@sys-con.com



WRITTEN BY ALEXANDER FALK

XML IDEs vs Classic IDEs: Competition or Synergy?

During the past few months I've seen several discussions on the validity of the term *XML developer*. Does this breed of developer exist in today's computing industry? Has XML matured to a stage that it warrants job descriptions for developers who specialize in XML programming? Is there an independent category of programming that relates primarily to XML?

For example, recent articles in *XML-Journal* (Vol. 3, issue 3) have sparked a discussion in the community that orbits around the needs of the XML developer – a new breed of developer whose scope of work encompasses a multitude of different XML-related technologies to a point where the actual programming language these folks work in is no longer relevant. This is much the same process that has occurred in other areas, and gave birth to the database developer some 34 years ago with the standardization of SQL. I'd like to address this issue here, and demonstrate that XML and the XML developer have definitely come of age.

In light of this development, the behavior of such a developer certainly bears investigation, particularly with respect to development environments. This will allow us to shed some light on the recent debate about whether there is a natural synergy between classic IDEs (such as Borland's JBuilder or Microsoft's Visual Studio) and XML IDEs, or whether these IDEs will compete in the same marketplace.

The XML developer typically works with any number of XML-related documents, such as XML files, DTDs, XML Schemas, XSLT stylesheets, XPath expressions, WSDL and WML files, SVG (Scalable Vector Graphics) files, XHTML documents. The list grows almost monthly as various standards organizations produce a never-ending stream of new proposals and recommendations. The work done with these documents depends largely on the specific application, but typically includes such tasks as creating XML use-case scenarios, developing DTDs or schemas, creating instance documents (either programmatically or by hand), validating instance documents against the defining schema or DTD, designing XSLT stylesheets, developing Web services, testing Web services for interoperability, and debugging XPath expressions. The list also grows as we discover more scenarios in which XML can be applied to solve issues in new domains.

It's certainly true that most classic IDEs have recently added XML-related features to their repertoire of classic edit-compile-link-debug functions, but their focus undeniably rests on the programmer, who is tasked with creating software that produces or consumes XML data – a narrow view that doesn't do justice to the scope of tasks previously discussed with respect to the needs of the XML developer.

Take the latest Visual Studio .NET release, for example: this adds a simple XML Schema editor, some XML support in the generic text-editor, and schema-only validation to the Visual Studio IDE, which is a rather typical modern programming-language IDE (supporting C++, C#, Java, Visual Basic, and other languages through third-party plug-ins). While these new XML capabilities of Visual Studio will certainly help those developers who only casually need to touch up a schema or briefly look at some XML data once in a while, there are also lots of people out there – true XML developers – who work with XML, Web services, XSLT stylesheets, and XML Schemas on a daily basis. These people need many more XML capabilities than the programming-language IDEs such as Visual Studio can provide.

This is where the XML IDEs come into play, in that they are totally focused on providing XML-centric features to fill the needs that a classic IDE does not address, such as:

- A graphical XPath Query analyzer to visualize the resulting node-set of an XPath expression
- A schema generator that can extract and create an XML Schema from a set of different use-case XML instance documents or derive it from an existing SQL database schema

Continued on page 8

AL @ ALTOVA.COM

AUTHOR BIO

Alexander Falk, cofounder, president, and CEO of Altova, Inc., has been actively involved with XML since the beginning and is a member of the W3C Advisory Committee and the W3C XML Schema Working Group. Author of the XML Schema processor and XML parser for XML Spy, Altova's XML software suite, he previously contributed to the ResEdit software at Apple Computer.

- A SOAP debugger that can intercept Web services transactions and set conditional breakpoints on the value of any payload element selected by an XPath
- And thousands of other little details that make life with XML easier

Examples of IDEs that support these types of capabilities are Altova's XML Spy and Tibco's TurboXML.

This also illustrates an important aspect of what makes a *true* XML developer – the term isn't necessarily restricted to people who write programs that emit and consume XML data. On the contrary, people who analyze requirements specifications and develop DTDs or XML Schemas, who develop stylesheets for transformation to diverse presentation form or interfaces for Web services – they all should be considered XML developers.

But the definition should be broader. Instead of restricting it to the classic developer (i.e., programmer) who just happens to work with XML, the *XML* developer is a person who develops XML-related materials – be it DTDs, schemas, stylesheets, or programs. This is analogous in many respects to the concept of a *database* developer or a *Web* developer – both terms describe much more than just the creation of programming language statements that deal with databases or Web pages!

As a result, we can now view XML-centric IDEs and classic programming-language IDEs from a new perspective: they aren't competing with each other, but instead are symbiotic in the sense that one provides what the other doesn't. These different paradigms are really complementary, and tools in those markets will continue to integrate well to provide developers in need of both sets of tools – programmers who are also XML developers – with a synergy of capabilities that make them truly productive.

To validate this perspective, let's take a look at an established market (>30 years) with a different powerful paradigm – SQL – and we'll see the same picture. Many programming-language IDEs provide a small set of SQL features to address the needs of the average developer. At the same time, there are several tools on the market for true SQL developers (or, more generically speaking, database developers) – for people who work with SQL daily and need more SQL capabilities than the run-of-the-mill IDEs can provide. Some of these SQL developer tools are so feature-rich that you

could call them *SQL* IDEs, because they address those needs perfectly.

To use a more recent analogy, consider the case of Web development tools: while you can certainly create Web sites using classic IDEs – and some of them do indeed provide some HTML features – most professional Web developers still favor a Web development tool, such as Macromedia Homesite, for their Web-related work and typically use it in conjunction with a classic IDE to get the best of both worlds.

It's therefore valid to assume that things are going to play out similarly in the XML market space – history *is* prone to repeat itself....

In addition, of course, there is the issue of XML editing beyond the need of the developer, which is again something that's typically addressed by other components in the product line of XML vendors: graphically designing XSLT stylesheets (which is more a task for Web designers than for developers) and editing XML content to be stored in content management systems (which is important for all areas of enterprise knowledge-management). These things aren't going to be of any interest in programming-language IDEs at all, and consequently there's another field of possible synergies out there to empower XML developers with the tools to create and integrate such content-management solutions within the entire IT infrastructure of enterprise customers.

In the end I think we've established beyond doubt that there is indeed a new breed of XML developer out there. They come from a diverse variety of backgrounds that certainly includes – but is not restricted to – programmers, who will need both a traditional programming-language IDE and an XML IDE. To limit the scope of the term *XML developer* to programmers only would certainly be a huge mistake and wouldn't do justice to the entire idea of XML – to be extensible beyond our current imagination. ☘

URLs for products mentioned in this article:

- Altova XML Spy: www.xmlspy.com/
- Borland JBuilder: www.borland.com/jbuilder/
- Macromedia Homesite: www.macromedia.com/software/homesite/
- Microsoft Visual Studio .NET: <http://msdn.microsoft.com/vstudio/>
- Tibco Turbo XML: www.tibco.com/products/extensibility/

Happy 4th Birthday, XML!

As part of XML-J's ongoing celebration of XML's 4th birthday (actual date: February 4, 1998, the date of the W3C's XML 1.0 specification), we asked: What would be the best birthday present anyone could give to XML?

From standards activist RICK JELLIFFE, CTO of Topologi, in Australia:

The best birthday present for XML would be for standards makers to stop treating XML as a carbundled data structure and instead to nurture it as a software engineering resource. In software engineering the importance of human factors is really coming into the limelight: we need to focus on XML as a tool for humans to comprehend, make, and use complex systems on the WWW.

One practical step forward would be for standards makers to only make "localized" specifications rather than "globalized" specs. You know that XML Schemas only allow one date format: if you send a date in American format, you cannot use an XML Schema to say "this is a date." That's an example of a globalized spec, and it encourages a certain division of processing between clients, servers, middleware, peers, and so on.

A localized specification for dates would be one in which the lexical space of the date datatype could be given along with a mapping to value space (i.e., what SGML calls *notations*).

Localization is what XML did with character encodings: you can

use any encoding you want, but it gets mapped to Unicode. The globalized approach would be merely to say "everyone must use UTF-8."

Another good present would be for XML to come down to the desktop. If I were one of the large players, I would hope to get everyone's attention on XML for its uses for centralized DBMS and Web services systems, so that people will forget its potential in publishing and peer-to-peer systems, where it may be disruptive.

—Rick Jelliffe

From CLAUDE L. (LEN) BULLARD, an analyst for the Intergraph Public Safety Corporation and an active member of the xml-dev list-serv community:

A raincoat and wellies are what XML needs. Any good idea can be promoted into absurd applications. That's when the wading gets deep and Wellington boots are just the thing. I chose them because the weather is changing fast and the Alberta Clipper is a-comin' South.

—Claude L. Bullard



XML Global Technologies, Inc.

www.xmlglobal.com/yourinformation

A real-life example,

IBM is one of the most dominant players in the push for Web services. It's therefore not surprising that much of the Web services work done by the company has been incorporated into their flagship product – the WebSphere Application Server.

As of version 4, support for Web services is incorporated into every level of WebSphere. This means that a few of the Web services libraries come with the WebSphere Application Server. Specifically, support for SOAP and UDDI4J are both an integral part of the WebSphere Application Server and used for implementing Web services, servicing calls made using SOAP, making calls to other Web services, and allowing applications deployed over WebSphere to interact with UDDI registries – both for discovering and publishing services.

The support provided for Web services within WebSphere focuses on the deployment aspects only. The focus on tools and developer productivity is part of WebSphere Studio and includes a WSDL generator. These tools were originally packaged as the Web Services Toolkit, (still) available through AlphaWorks. These tools can work with WebSphere 4 (actually, they also work with WebSphere 3.5) as well as with a number of UDDI repositories. The same tools have been improved and repackaged and are available as part of WebSphere Studio.

This article focuses on the deployment aspects and describes a full process in which I take a JAR file implementing an important business function and expose it as a Web service deployed on a WebSphere Application Server. For an overview of Web services in general and SOAP, WSDL, and UDDI specifically, please see my article in the November issue of *XML-Journal* (Vol. 2, issue 11).



DEPLOYING WEB SERVICES ON

using the tools and services provided

The Scheduling Service

The example I'll use as the Web service involves an important business function from the realm of workforce management (a specialty within CRM). I'm giving a brief description of the service because I feel it's important to show a real use of Web services in a real business context. I've been very disappointed and frustrated by the fact that almost every example I see of Web services is some permutation of the stock quote function. One could get the impression reading through the trade press that this is all Web services are good for, which of course isn't true.

The example is taken from a real application called the ViryanNet Service Hub, which publishes its core capabilities as XML-enabled APIs.

What I'm deploying as a Web service is a scheduling service. The input is a set of tasks that have to be performed within a day (the general problem isn't limited to a single day, but I'll use this simplification here). Each task has a location where the service needs to be provided – think of it as a place where something needs to be installed or fixed. Each task also has an appointment window – a start and end time that defines when the resource should be on-site. The other input to the system is a set of resources, usually field personnel. Each such resource has a home location and a set of availability constraints. These define when the resource can perform work.

The output of the service is a list of assignments, in which each task is assigned to a resource, as well as a sequencing that indicates the order in which the resources should perform the tasks assigned to them. The goal is to find an efficient assignment – one that minimizes travel time while conforming to all sorts of constraints that may be applicable. Different schedules imply very different costs to a business, and a good scheduling algorithm can generate huge savings.

The scheduling service is a good example of a function that should be deployed as a Web service. The problem at hand is hard. In fact, it's a highly complex version of the traveling salesman problem, which is known to be NP-complete. Therefore, any algorithm solving this problem will probably not provide the optimal solution in every case but can employ heuristics to

This article is derived from various chapters in *IBM WebSphere: The Complete Reference*, to be published later this year by Osborne/McGraw-Hill.

Written by Ron Ben-Natan
WEBSphere

get good solutions. The algorithm is thus of great value, hard to build, and worth much to a potential small business (which would often not have the resources to build such an algorithm in-house). What better way to provide value than to offer this as a service through a central site where customers submit their problem data and get a schedule they can then use to good effect? This would certainly be something worth paying for.

Building the EAR/WAR

Obviously I won't go into the details of the algorithm; it's very complex and highly proprietary. Assume that I've packaged the scheduling algorithm as a set of Java classes in a JAR file. The first thing I need to do in exposing this as a Web service in WebSphere is deploy it as an enterprise application and a Web module.

I start by opening the application assembly tool that's part of the WebSphere installation. Once started, I pick the application option from the entry window. In order to build the EAR I need to add the JAR file (shown in Figure 1) as well as set up the application's data.

The next step is to create a new Web module, which is mandatory in the current version of WebSphere. The module isn't directly related to the invocation of the Web service, but a true scheduling package requires more than just the service. It includes a user interface by which the assignments can be viewed and through which users can modify assignments, see outstanding tasks, view locations, and so on.

The assembly tool is also used to add the JSPs and servlets to the Web module. After the module is created and the servlets and JSPs required are added, I can import the WAR as a component of the enterprise application, as shown in Figure 2.

At this point the algorithm has been wrapped within an EAR (along with the user interface components) and I can proceed to create the SOAP wrapper and install everything on the WebSphere server for use as a Web service.

Creating the SOAP Deployment Descriptor

For the server to expose the application features as a Web service, I need to prepare the SOAP layer. The libraries installed as part of WebSphere and the tools that come as part of the WebSphere installation make this easy to do.

In this case I'm exposing a Java class and its methods. Therefore, I need to define which class and which method provide the service. I do this using the following deployment descriptor:

```
<isd:service
  xmlns:isd="http://xml.apache.org/
  xml-soap/deployment"
  id="urn:viryanet-advanced-scheduling">
  <isd:provider
    type="java"
    scope="application"
    methods="runAdvancedScheduling">
```



```
<isd:java class="com.rts.
  scheduling.AdvancedScheduling"/>
</isd:provider>
<isd:faultListener>
  org.apache.soap.server.
  DOMFaultListener
</isd:faultListener>
</isd:service>
```

The deployment descriptor is part of the Apache SOAP package. Deployment descriptors are XML documents that provide information to the SOAP runtime about the services available to client invocations. The exact content depends on the artifact that provides the service; for example, the deployment descriptor for an EJB providing the service will have `isd:option` elements in the `isd:provider` element. For a quick introduction to deployment descriptors please see <http://xml.apache.org/soap/docs/guide/deploy.html>.

Once the deployment descriptor is ready, I run the SoapEarEnabler tool, part of the WebSphere distribution that exists in `WAS_HOME/App-server/bin`. Running the tool for the EAR is simple enough and the console session is as follows:

```
IBM WebSphere Application Server Release 4
SOAP Enterprise Archive enabler tool.
Copyright IBM Corp.,1997-2001
```

```
Please enter the name of your ear file:
f:\Godzilla\appserv\classes\scheduling.ear
```

```
***Backing up EAR file to:
f:\Godzilla\appserv\classes\scheduling.ear~
```

```
How many services would you like your
application to contain (1...n)?1
```

```
Now prompting for info for service 1:
Please enter the file name of the SOAP
deployment descriptor xml file:
f:\Godzilla\appserv\classes\scheduling.xml
Is this service an EJB;(y =yes /n =no)?n
How many jarfiles are required for this
service (0...n)?1
Classpath requirement 1:Please choose a
file:([1 ]scheduling.war):1
Should this service be secured;(y =yes /n
=no)?n
```

```
Please enter a context root for your non-
secured services (e.g./soap):
/scheduling
```

```
Do you wish to install the administration
client?
Warning,you should not install this client
in a production ear unless you intend to
secure the URI to it.
```

```
Install the administration client;(y =yes
/n =no)?y
```

Installing the Service

Now that I have the package, I can deploy it onto WebSphere, a standard deployment of the package as an enterprise application. To install the EAR I use the administration console and go to the enterprise application folder. Clicking the

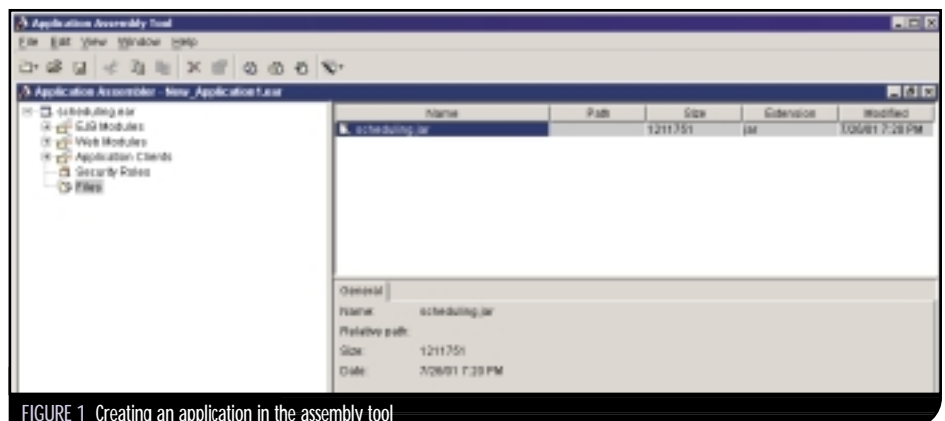


FIGURE 1 Creating an application in the assembly tool

iWay Software

www.iwaysoftware.com

install button brings up the enterprise application installation wizard. I enter the ear file name, select the host name, and confirm the application details. The newly installed application is then started. Finally, the Web server plug-in is regenerated, allowing for access to the application using the correct URL structure. The WebSphere instance is now ready to provide scheduling services using the advanced scheduling algorithms.

Creating the WSDL

Although I'm all set to go now, since the service has been installed on the server, it can't real-

ly be used unless you know exactly where the service resides and how to invoke it. To complete the deployment I need to create a description of the service (i.e., generate a WSDL file) and then publish the service (along with the describing metadata) to a UDDI repository.

In the example I'll use the Web Services Toolkit and generate the WSDL file using the `wsdlgen` utility. Note that you should have the right classpath set before you attempt this, because the utility uses introspection to get metadata from the class definition so as to create the appropriate typing information.

Starting the utility brings up the first screen of the generator. I fill in the class name and make changes to the URN and service name. The utility then brings up the available methods in the class. The tool can wrap the available public methods and the metadata created for these methods in the form of a WSDL file. If the method signatures have only simple types, then wrapping requires no work on our behalf. In such a case I can be done in no time.

Unfortunately, real business functions don't have just one int and one string as their arguments. Complex services typically require complex data structures. In the scheduling service, for example, the input to the service includes complex and recursive data types. In this case the method will be marked up with a red dot on the left of the method. If I select this method, I'm shown the complex types for which an additional mapping is required. I can then select any (and eventually all) of them and generate wrappers. Figure 3 shows some of the tool's capabilities as described.

At the end of the process I have a full WSDL file (which is very lengthy) without having to write a single line of it.

Interacting with a UDDI Registry

WebSphere doesn't provide a private UDDI registry, which disappointed me a little at first. What WebSphere has in terms of support for UDDI is a Java library called UDDI4J that allows Java programs to interact with a UDDI registry both for publishing and for discovering services. This means I can put together

a simple Java program to publish the service in a private or public registry.

While this may be enough, I hope (and believe) that future versions of WebSphere will include a UDDI registry. I'm currently left with (1) using a public registry, (2) using a third-party registry that I can install and manage, or (3) using a preview release of the IBM UDDI registry that can be downloaded from www7b.boulder.ibm.com/wsdd/downloads/UDDIRegistry.html. Let's proceed with the third option.

The WebSphere UDDI Registry Preview runs on WebSphere Application Server Version 4.0 (hence my belief that it will be included in a future release of WebSphere). It supports the 20 SOAP-based APIs defined by version 1 of the UDDI specifications and provides persistence for published entities using DB2. The download of the entire package is massive (the download of DB2 itself is almost 300MB in size) and it takes quite a bit of time to install.

The package includes a Web-based graphical user interface that supports publishing and querying of businesses, services, and other UDDI-compliant entities without programming. This is what I'm interested in right now.

After logging into the tool I go to publish the service. The first thing is to define a new business entity that will service the scheduling capabilities. As part of the business entity I define contact and address information and the like. The business entity isn't the most important part.

The next step is to define the service endpoint. This not only specifies ways to discover and find the service through service descriptors, it also defines the service access point, that is, where this service can be found and how it can be invoked. This defines the method for creating a request to the WebSphere server on which I installed the package. Some of the service details are shown in Figure 4.

Summary

Web services are the wave of the future, and WebSphere is right up there, along with other dominant platforms like .NET, providing full support for them. This article examined the tools and services provided by WebSphere through a real example, from conceptualizing the service as a business function that's useful for remote and decoupled usage, through packaging the implementation as an enterprise application, all the way to enabling it through Apache SOAP, creating the WSDL, and publishing it in a UDDI registry. ☺

AUTHOR BIO

Ron Ben-Natan, chief technology officer at ViryaNet Inc., previously worked for Intel, AT&T Bell Laboratories, Merrill Lynch, and J.P. Morgan. He holds a PhD in computer science in the field of distributed computing and has been architecting and developing distributed applications for almost 20 years. The author of CORBA, Objects on the Web, and CORBA on the Web, Ron coauthored IBM San Francisco Developer's Guide and IBM WebSphere Starter Kit.

RON.BEN-NATAN @ VIRYANET.COM

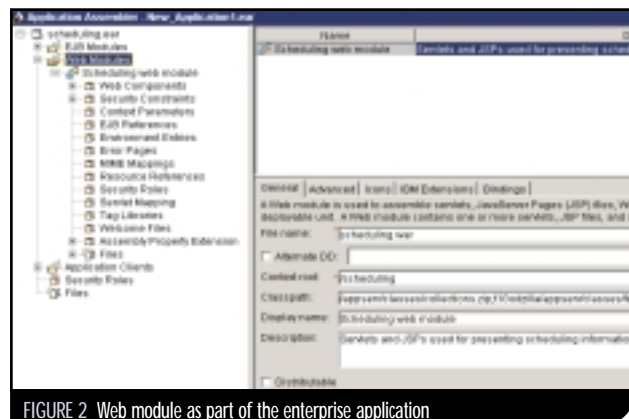


FIGURE 2 Web module as part of the enterprise application

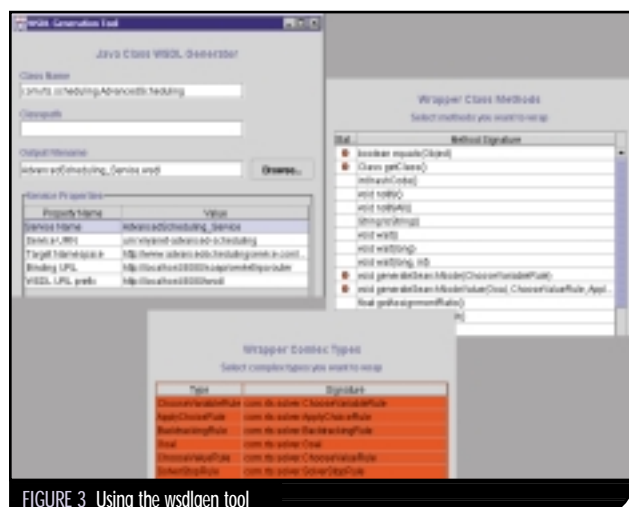


FIGURE 3 Using the wsdlgen tool

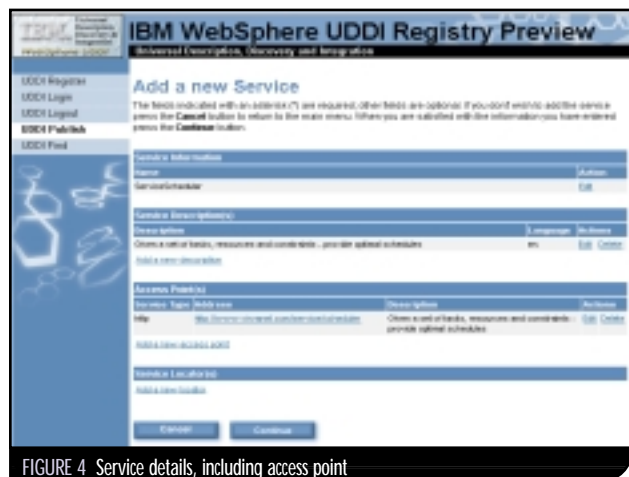


FIGURE 4 Service details, including access point

SilverStream Software, Inc.

www.silverstream.com



Web services...middleware for the masses...XML miracle tools
...instant integration: just add SOAP

.NET Web Services: The 'Three I' Monster

As readers of a magazine focused on XML, it may amuse many of you that the world has suddenly discovered what we've known for almost six years: XML is a massively powerful metalanguage. Nowhere is that proven better than in the sudden explosion of Web services in the marketplace.

Microsoft embraces Web services as an important component of their .NET framework, but rather than take it on as an afterthought, Microsoft went out of their way to make Web services both simple and pervasive. Indeed, XML itself is the lifeblood of .NET, with major components easily communicating back and forth, using XML when needed. Data persistence is also gracefully handled in the back end of .NET applications in XML, with a minimum of programming. What makes XML in the .NET framework so pleasant is that all of these features are easy to use and powerful. Going a step further, Microsoft's Visual Studio tool makes designing XML schemas a snap. What this means is that XML has finally been promoted out of buzzword status and into the real world, and the pack leader is Web services.

For good or bad, the term *Web services* is going to stick. I find that unfortunate, because the only thing that's really "Web" about them is that they can communicate over HTTP, the common transport protocol of the Web. As time progresses, we'll see more efficient protocols replacing HTTP for Internet communication, but most likely they'll still be called "Web services." In reality, Web services are *integration components* (a concept analogous to the hardware IC – a "software chip," if you will). Web services can do any combination of three things: *Inform*, *Inquire*, and *Invoke*. (Want a really good mnemonic? Remember "IC the 3 I Monster"!)

So how can we make sense of the Web services acronyms then: UDDI, WSDL, and SOAP? Let's do it the way I like, using the *All-you-need-to-know-about-Web-services-acronyms-in-one-paragraph* method.

Are You Ready?

Let's use the process of faxing as an analogy. A Web service *informs* a centralized directory that it exists, like putting a fax number in a phone book. Another Web service might *inquire* about how to use another Web service, like looking up a number in a phone book. A Web service then *invokes* another service, like dialing a fax number. The fax is finally transmitted, and the client receives a "special package offer" for a vacation in the Bahamas. Now that you have the fax analogy in your head, here's how the Web services acronyms fit together: UDDI = phone book, WSDL = fax number, SOAP = a faxed page, and HTTP = telephone-carrier signaling. There. All done! For those that want some more details, the next few paragraphs should help a little.

UDDI – Universal Description, Discovery, and Integration – is a mechanism to store and publish Web services. This is the cornerstone of Web services, because we can't learn how to integrate with other systems without searching for and finding the service

es we want first. I've noticed that the thought of UDDI servers "publishing" corporate Web services makes some executives feel uneasy. The fact is, of course, a UDDI server can run anywhere, and can be exclusively used within a corporate intranet. You're still publishing Web services, but only within the corporate infrastructure. In that context you might even call them *enterprise services*.

WSDL, the Web Services Description Language, is the lingua franca of Web services. Web services will store WSDL data on a UDDI server, which tells other Web services *how* they can use a Web service, and *what* information it needs and returns (and whether it gives you error messages).

And, finally, there's **SOAP**. SOAP is how data, in both textual and binary forms, gets bundled up into an XML format and shipped between Web services. The acronym means *Simple Object Access Protocol*, although I've heard others call it (perhaps more appropriately) *Simple Open Access Protocol*. SOAP is optional, of course – you can pass parameters to a Web service using HTTP "GET" or "POST" parameters, but you're limited in many ways as to what you can pass, and there's very little validation that the system can do *before* the Web service is invoked. It's also important to know that SOAP doesn't require HTTP as its carrier protocol. It could also use SMTP (e-mail) or FTP.

Okay, let's get to what you really want to know: How are Web services useful to a business? More important, what's the right way to use them?

Three Important Points

First, we have to remember that Web services are not a miracle tool. They don't solve all integration problems, and they darn sure don't increase transactional efficiency (I'll cover "overhead" issues later in the article). They do,



AUTHOR BIO

David Weller is a principal managing consultant at Valtech Technologies, Inc., an international consulting firm specializing in .NET/J2EE/Unified Process development, skills transfer, and training. He holds a computer science degree from the University of Houston at Clear Lake. David currently resides in Dallas.

THE INSIDER INTELLIGENCE YOU NEED...

TO KEEP AHEAD OF THE CURVE
FREE E-Newsletters
SIGN UP TODAY!

Go to www.SYS-CON.com

The most innovative products, new releases, interviews, industry developments, and plenty of solid *i*-technology news can be found in SYS-CON Media's Industry Newsletters. Targeted to meet your professional needs, each e-mail is informative, insightful, and to the point. They're free, and your subscription is just a mouse-click away at www.sys-con.com.

SELECT THE INDUSTRY NEWSLETTERS THAT MATCH YOUR NEEDS!
CHOOSE ONE – OR TRY THEM ALL!



Don't Delay!
Subscribe
for **FREE!**

at www.sys-con.com

Exclusively from the World's Leading *i*-Technology Publisher



however, dramatically simplify getting applications to communicate reliably with one another. This can be particularly beneficial to companies that have “language camps,” where you have entrenched developers working in their-favorite-language-that-everybody-else-should-use-if-they-had-half-a-brain. And, of course, people working in distinctly different platforms – some obvious examples being Java applications running inside the Java Virtual Machine, C++/COBOL applications using CORBA, and .NET applications running inside the Common Language Runtime (CLR). Web services, in this case, are genuinely a miracle tool. There, I just contradicted the first sentence in this paragraph!

Second, finding good examples of Web services on the Internet isn't really easy to do. At the moment we'll probably find only simple “trinket” Web services that are publicly available (there's a nice list on www.xmethods.com that's growing daily). The power of Web services, however, is there. In many ways I'd venture to guess that Web services right now is where the phone system was when Alexander Graham Bell invented the world's first busy signal. But I'm guessing that by the end of 2003, Web services will have caught up to “The Age of Telemarketers.”

Third, Web services are simple to create and use. That's why they're effective. Nowhere is this better demonstrated than with Microsoft's .NET Web services. All is not lost for Java-oriented developers, though. The GLUE product

from The Mind Electric (www.themind-electric.com) is an excellent tool in that arena – surpassing what you might get from IBM or BEA at the moment. Anyone who's tried to build interoperable applications using CORBA will greatly appreciate what Web services bring to the table. To paraphrase the late Douglas Adams, “Web services provide language-, platform-, and vendor-independent integration, exactly the same way that CORBA doesn't.”

Meaningfulness of Web Services

Let's ponder how Web services are meaningful to businesses now. The effectiveness of a Web service will be measured in terms of the functionality it delivers. Businesses aren't going to invest money in developing Web services just to, say, publish a service that gives consumers an on-time stock ticker quote. Web service effectiveness will come at a much higher level of granularity, primarily in the form of encapsulating important business processes that are meaningful to a client: payment processing, order fulfillment, and claim processing, to name a few. These have been traditional domains of Electronic Data Interchange, but Web services can do it far easier, primarily because it's significantly more flexible and maintainable. I'm predicting that EDI will be largely displaced in large and small businesses within the coming decade, and will most likely be reduced to a niche market within the next five years.

Businesses will also offer smaller, end-consumer-style services designed to build affinity and customer confidence. The example in this article details a couple of ways a lending company can do that. Once I've covered the example, I'm going to describe a little more about the “Big 3” that companies insist on before making a large-scale assault on Web services, plus I'll go over some of the overhead issues associated with using Web services. Since I'm a strong proponent of hands-on learning, let's not waste any more time and get into the example.

Learning by Example

Let's take a hypothetical company, The Elmo P. Foobar Bank and Trust (let's call it “EPF” for short). They want to start sticking their toes into the Web services waters because they know that next year they'll have to integrate with all their partners using Web services. They've decided to offer some simple mortgage calculations to the end consumer, by means of Web services. They want to offer five different mortgage calculation services:

1. Compute a monthly payment given the principal (the amount loaned), the annual interest rate, and the period of the loan (in months).
2. Compute the annual interest rate given the principal, period, and monthly payment.
3. Compute the period (number of months) given the principal, monthly payment, and interest rate.

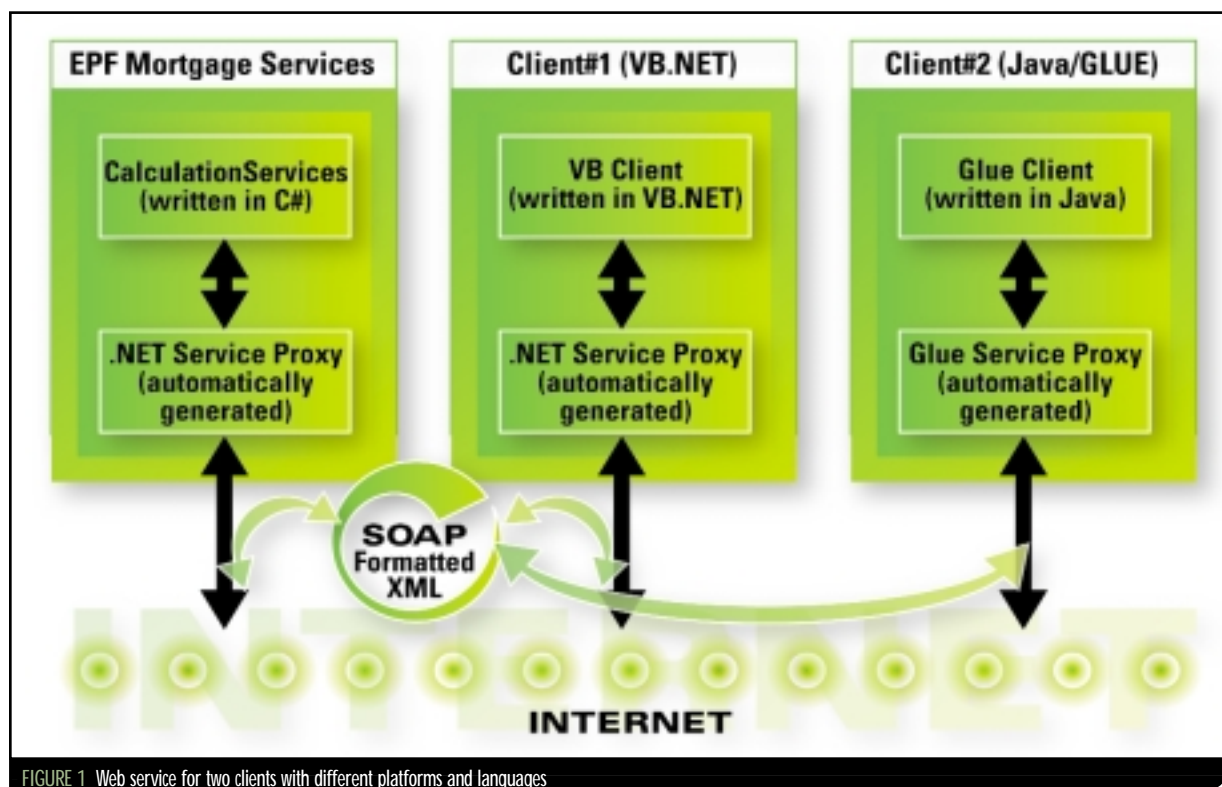


FIGURE 1 Web service for two clients with different platforms and languages

Missed an issue?

We've got 'em all for you on CD!

JAVA DEVELOPERS' JOURNAL

The most complete library of exclusive **JDJ** articles on one CD!

Check out over 500 articles covering topics such as...

Java Fundamentals, Advanced Java, Object Orientation, Java Applets, AWT, Swing, Threads, JavaBeans, Java & Databases, Security, Client/Server, Java Servlets, Server Side, Enterprise Java, Java Native Interface, CORBA, Libraries, Embedded Java, XML, Wireless, IDEs, and much more!

JDJ The Complete Works
Reg \$119.99

Buy Online
Only **\$71.99**

XML JOURNAL

The most complete library of exclusive **XML-J** articles on one CD!

Check out over 150 articles covering topics such as...

XML in Transit, XML B2B, Java & XML, The XML Files, XML & WML, Voice XML, SYS-CON Radio, XML & XSLT, XML & XSL, XML & XHTML, 2B or Not 2B, XML Industry Insider, XML Script, <e-BizML>, XML & Business, XML Demystified, XML & E-Commerce, XML Middleware, and much more!

XML-J The Complete Works
Reg \$59.99

Buy Online
Only **\$53.99**

COLDFUSION Developer's Journal

The most complete library of exclusive **CFDJ** articles!

Check out over 250 articles covering topics such as...

Custom Tags, ColdFusion and Java, Finding a Web Host, Conference Reports, Server Stability, Site Performance, SYS-CON Radio, ColdFusion Tips and Techniques, Using XML and XSLT with ColdFusion, Fusebox, Building E-Business Apps, Application Frameworks, Error Handling, and more!

CFDJ The Complete Works
Reg \$79.99

Buy Online
Only **\$71.99**

CF Advisor

The most complete library of exclusive **CFA** articles!

Check out over 200 articles covering topics such as...

E-Commerce, Interviews, Custom Tags, Fusebox, Editorials, Databases, News, CF & Java, CFBasics, Reviews, Scalability, Enterprise CF, CF Applications, CF Tips & Techniques, Programming Techniques, Forms, Object-Oriented CF, WDDX, Upgrading CF, Programming Tips, Wireless, Verity, Source Code, and more!

CFA The Complete Works
Reg \$79.99

Buy Online
Only **\$71.99**

SPECIAL OFFER:
Buy CFDJ & CFA
The Complete Works
For Only **\$129.99**



JDJStore.com

Order Online and Save 10% or More!

WWW.**JDJSTORE**.com

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

COLLECT
FOR ALL
ONLY \$**229.99**
JDJ, XML-J
CFDJ & CFA **4**

4. Compute the principal given the period, monthly payment, and annual interest rate.
5. Return a payment schedule given the principal, annual interest rate, and period.

Initially, the EPF developers thought that creating five separate services would be fine, but one of the junior developers comments that the first four differ from each other only by a missing parameter, and that since Web services should aim for higher granularity, they really need to publish only two separate Web services. All the developers agree that this is a wise thing, and thus focus on two Web services: Calculate and Schedule. (As a side note, the junior developer was shortly thereafter promoted into the ranks of management and was last seen babbling incoherently about quality circles and employee empowerment.)

The application behind the Web service will be written in C#. However, two different service consumers will be using different platforms and languages. One company is a Visual Basic shop, but recently moved to the .NET Framework, so the client app is written in VB.NET. The other company has been a longtime UNIX shop, and all their new applications are written in Java.

The first thing you should notice in the simple diagram in Figure 1 is that each client uses a proxy to communicate with the Web service. The second thing is that the proxies are automatically generated for you. This allows you to focus on the details of writing your application without needing to know the sticky details of SOAP or WSDL. In fact, I don't advise trying to build WSDL by hand at all; it's much too frustrating. For GLUE clients, you simply invoke `wsdl2java` to create your proxy. For .NET clients, you call the `wsdl` command and specify the language you want to generate. After that, you can call the Web service as if it were simply another method in your system. It's that easy!

Build and Test

It's time for a little commonsense advice now: you should build and test your class before exposing it as a Web service. Nothing is more frustrating to service consumers than discovering the interface or behavior of a Web service has changed. Technically speaking, of course, the UDDI registration process prevents mismatched interfaces from

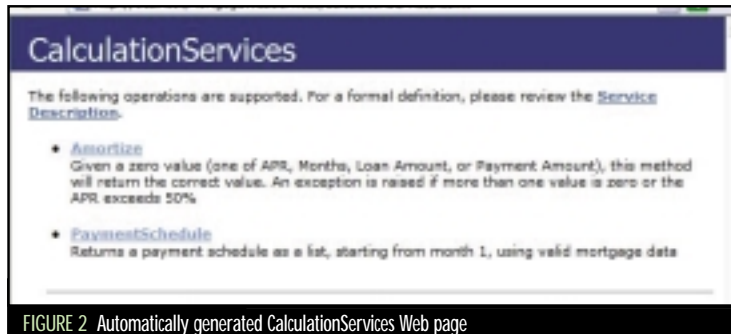


FIGURE 2 Automatically generated CalculationServices Web page

being an issue, but changing the behavior of a previously registered Web service can be infuriating to service consumers.

Assuming you have a complete and tested class, you can now expose selected methods as a Web service. There are really only three basic steps you must follow when building a .NET Web service:

1. Include the `System.Web.Services` assembly in your project.
 2. Have your class inherit from `System.Web.Services.WebService` (this is actually optional, but gives a lot of convenience features through ASP.NET).
 3. Put a `WebMethod` attribute before each method you want to make available.
- Note:* The specific syntax for attributes varies with the language, but the attribute name is always `WebMethod`.

In our service we're exposing `Amortize` and `MonthlyAmortization` to our clients (see Listings 1 and 2 – all source code referenced in this article can be found at www.sys-con.com/xml/sourcec.cfm).

Once you've written your class (and assuming you inherited from `System.Web.Services.WebService`), you can verify the service is available by pointing your browser to the Web service URL. Figure 2 shows what your page will look like (this page is also automatically generated for you).

Clicking on either of the services will show you the complete SOAP specification. Adding the URL parameter `"?wsdl"` to the URL will reveal the WSDL associated with that class. Two "helper" classes are also part of the service (see Listings 3 and 4). These classes separate the structure of the information from the Web service class. These classes could just as well reside in the same file as the `CalculationServices` class.

That's all there is to it on the server side. On the client side it's just as simple. Assuming that you've already created your proxy (remember, it's the `wsdl` command for .NET and the `wsdl2java` command for GLUE), all you need to do is write a simple user interface and call the Web services exactly as if they were methods. I used the same syntax for both the

Java (`GlueClient.java`) and VB.NET (`VbClient.vb`) invocations: `Calc.-Amortize` and `Calc.PaymentSchedule` (see Listings 5 and 6).

I've left off some important parts in the client code, and that has to do with erroneous input and handling SOAP-Exceptions. I decided to leave that as an exercise

for the reader...

In Brief, What's Been Left Out

Some important factors have been left out of these examples. No matter how effective or handy a Web service is, a business can't successfully use Web services without the "Big 3": security, transactions, and orchestration. The details of these are beyond the scope of this article, but let's touch on each one.

Security

A Web service can be found and used only by authorized and authenticated service consumers, and the contents of the service can be received and understood only by the service consumers. Fortunately, Web services have two security mechanisms: (1) they can work common authentication features like HTTPS or signed certificates, and (2) SOAP messages themselves can have strong encryption of all or part of the message. These solutions are secure, but in many cases lack flexibility. Current efforts are under way to enhance encryption features in Web services, but existing mechanisms are perfectly secure, should you choose to use them.

Transactions

A Web service can deliver contents in a time-consistent manner, that is, the exchange of information between the service provider and the service consumer adheres to the "ACID" test (Atomic, Consistent, Isolated, and Durable – these concepts are well known in the database world, and I refer you to introductory texts on database concepts for more details). This focuses on the contents of the service, not the invocation process.

The .NET framework contains some mechanisms to support Web services in a transactional environment, but they aren't very advanced. In order to address that, Microsoft has proposed the XFLOW language, which includes transactional processing support. Unfortunately, other "flow languages," like WSFL and XAML, are competing in the same space. Hopefully this matter will be resolved this year.

Orchestration

Somewhat related to the transactional issue, this issue focuses on successful orchestration of multiple distributed Web services in a way that ensures all participants receive the invocations they're supposed to, and in the correct order. This mechanism focuses on the invocation process and ordering, not necessarily on the contents. Microsoft's newly released BizTalk 2002 provides an orchestration layer that can easily coordinate Web services. Developers that will be deeply entrenched in business processes using Web services will find BizTalk a welcome product.

All that aside, the time to build Web services is indeed "now." There are already good security and transactional mechanisms in place, and very shortly the orchestration issue will be addressed.

With all this good advice, what are the "gotchas"? Well, not many, but there are a few...

- **Data transformations:** Keep in mind that information undergoes several transformations before being sent over the wire. It gets converted, broken apart, mashed, and diced up before it heads out your door (not unlike your luggage at the airport these days). Using Web services adds in the steps of converting into SOAP format when it's

sent, and converting back again if the service responds with results. On the brighter side, most of this process is handled within your local system. This is good, because converting/moving data within the same application is about as fast as visiting your next-door neighbor, compared to the time it takes data to get converted, sent over the wire, and reconverted on the receiving end, which on the same time scale is like making a round trip to Pluto.

- **Decoupling classes:** Hard-core "object-oriented" people, of course, would get a bit uncomfortable with this, but "industrial strength" Web services aren't created by simply slapping a special tag in front of a class method. You need to take the time to create a service-oriented interface in front of your system (a facade) that exposes the services and data structures separately. This ensures that your Web service offerings are decoupled from the long-term class maintenance issues. Speaking of maintenance...
- **Versioning:** Let's face it; publishing a Web service is like pouring concrete. Once your users start to use a Web service, changing that service is going to invoke pain across corpo-

rate boundaries – a sure way to upset your users is to introduce "churn" in your services. Therefore, make a point of *not* changing the behavior of a Web service, so that clients can invoke any version of a service you provide (unless you have compelling business reasons to remove a version or "turn off" a service).

• • •

Wow. I hope your head doesn't hurt as much as my hands do from typing this up. I know I've covered a lot of stuff in a short space, but hopefully you've learned some key points. Let's recap before I close:

- Web services, physically expressed as "interface components," have 3 I's: *Inform. Inquire. Invoke.*
- Making a Web service is easy. Making a really *good* Web service takes time and thought.
- Things the corporations want before making a big plunge into Web services: security, transactions, and orchestration.

I'll write some more details about Web services in a different article. Right now, I need to sign off...it appears that my Web services have started sending faxes to clients about vacations in the Bahamas....



DAVID.WELLER @ VALTECH.COM

EnginData Presents

2002 Developer Market Survey Reports



engindata.com

Our comprehensive reports offer insight and strategy to guide your most critical business decisions in today's fastest growing technologies...

- ✓ Establish your product and marketing strategy
- ✓ Understand your customer's needs
- ✓ Evaluate Technology & Trends

engindata
RESEARCH

*Persistence for Web services*

XML—BeyondTransport

The computing world is now positioned to deliver on its unmet promise of lowering the cost of doing business and providing the enterprise with easy access to additional markets. The computer industry finds itself in the third of three significant waves in the deployment of new software infrastructure that enables this promise – the adoption of the Internet, the expansion of mobile services, and now the move to Web services.

The first two waves were required to open access to and for the enterprise and its suppliers, vendors, and customers. The third is a novel approach to execution of new business models and offerings over a new and distributed means of access. This new model enables the consolidation and view of business information, opening a new vista for collaborative commercial exploitation.

The result is a convergence of software infrastructure and application development from point products to broader platform solutions – a move from tightly coupled synchronous architectures to loosely coupled asynchronous architectures. This enables collaborative applications that can manage information derived from unknown heterogeneous sources, respond in kind, and support disparate applications and business processes on common data views. Web services, unlike previous attempts at distributed computing, are designed to take full advantage of the architecture and standards behind the Internet, a requirement for broad adoption. Furthermore, Web services can be built using a number of underlying technologies, platforms, and programming languages, and can expose existing IT functionality, assuring nearly ubiquitous deployment over the next decade. XML is a key enabling technology necessary to achieve this third wave.

XML Changes Everything

Few technologies and standards have exploded through the marketplace as XML has over the last few years. XML is technically not a language itself, but a metalanguage, a means to create other markup languages. Industry groups have created hundreds of industry- and

domain-specific markup languages in an effort to standardize the means of communication within and beyond corporate walls. The use of XML as a means of standards-based transport between and within corporations is growing rapidly. However, XML's true power has yet to unfold.

For the first time in computing history, XML provides a widely adopted means to express "information," not just "data." Information is data with context. Since an XML document carries complete context with data, it is self-describing. Unlike traditional data representation models, such as relational or object models, in which the structure must be predefined, an XML document is complete. The structure can be derived from the tags. Furthermore, XML is multidimensional. Information is carried within the tags, within the tag hierarchy, within properly applied attributes, and within the data elements themselves. Unfortunately, to date, most work being done with XML is not fully leveraging these capabilities. Many XML schemas are poorly designed. While suitable for transport between systems, they miss out on the rich information modeling supported by the underlying metalanguage.

As more systems begin transporting XML, and Web services becomes the lead driver, an impedance mismatch between the transport and persistence layer occurs. While some XML is transient, generally it is stored and retrieved to and from back-office systems, often within an RDMBS. This mapping of hierarchical XML to and from relational tables is tedious and inefficient. Furthermore, DBMS's used in a relational model can't accommodate the key value propositions inherent in XML: heterogeneity, flexibility, and extensibility. While storing XML documents as CLOBs

(Character Large Objects) can provide some of these features, it can do so only at the document level and at the expense of having to round-trip the entire document. A new breed of information management system is needed, and is now available, to support the large-scale XML interchange provided by Web services.

Where Do Corporations' Information Assets Reside?

The problem is compounded because approximately 70% of all corporate information is unstructured and unmanaged, sitting within individual files, spreadsheets, documents, and e-mail systems. Much of the remaining data is stored in nonrelational mainframe systems, or it is nonrelational/hierarchical or object data forced into a relational model (see Figure 1).

While relational databases are extremely robust and efficient at managing well-defined relational schemas, they're ill-suited for managing most corporate information assets. Content-management systems try to address the unstructured part, but full-text retrieval without intelligent tagging is limiting, and does nothing about merging the structured and unstructured worlds.

Enter XML: XML is capable of describing "all" information. We need a natural mechanism for efficiently managing XML. While various dominant vendors try to layer XML capabilities on top of old technology, and others try to reignite the object databases to support XML, new technologies and new approaches designed with XML in mind are needed. It's time to enter the age of "information processing" and leave behind the world of "data processing."

We need self-constructing, heterogeneous, flexible, and standards-based databases that provide persistence in the "cloud" and at the "edge" of the network. Attempting to connect Web services to

AUTHOR BIO

Kevin Huck, chief architect, NeoCore Inc., has been involved in software engineering for nearly 20 years. Most recently, he has been involved with the development of a high-performance native XML database that extends the J2EE platform by offering XML and object-persistence mechanisms. A speaker at numerous small user groups, engineering organizations, and teams on system architecture, XML, and J2EE technologies, Kevin has a bachelor's degree in business administration from Baker University and an MBA in entrepreneurship from the University of Missouri - Kansas City.

Now in More than 5,000 bookstores worldwide

subscribe **Now!**

FOR FAST
DELIVERY

Go
Online
and
Subscribe
Today!

The World's Leading Independent WebLogic Developer Resource

FOR WLS DEVELOPERS BY WLS DEVELOPERS

WebLogic
DEVELOPER'S JOURNAL

WebLogicDevelopersJournal.com

SYS-CON Media, the world's leading publisher of *i*-technology magazines for developers, software architects, and e-commerce professionals, brings you the most comprehensive coverage of WebLogic. *Only \$149 for 1 year (12 issues) regular price \$180.

SYS-CON
MEDIA

Helping
you enable
intercompany
collaboration
on a global scale

- Product Reviews
- Case Studies
- Tips, Tricks
and more!

SPECIAL
INTRODUCTORY OFFER
SAVE \$31*
HURRY, DON'T DELAY! OFFER EXPIRES JUNE 30, 2002

SAVE UP TO \$100 ON MULTIPLE SUBSCRIPTIONS

Special
online offers

Pick **4** or **5** and Subscribe
for one **special low** price

**RECEIVE YOUR
DIGITAL EDITION
ACCESS CODE
INSTANTLY
WITH YOUR PAID
SUBSCRIPTION**



Wireless Business & Technology • Java Developer's Journal

Web Services Journal • WebLogic Developer's Journal

XML-Journal • WebSphere Developer's Journal

ColdFusion Developer's Journal • PowerBuilder Developer's Journal

SYS-CON
MEDIA

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

WWW.SYS-CON.COM/SUBOFFER.CFM

back-end legacy systems is fraught with peril, from significant performance and scaling problems to major security breaches, inflexibility, and inability to adapt to ever-changing business requirements. Most of these back-end systems weren't meant to be "Web-enabled" and simply can't handle the unknown, real-time demand load created by Web service interactions. Web services are all about integration, and integration usually involves various forms of middleware. Within a department or enterprise, traditional middleware solutions – while complex and expensive – can work well when properly designed. However, exposing functionality to outside vendors, suppliers, and customers creates a new set of problems: middleware and Web services need a "memory." They need an XML-based "Operational Information Store" (OIS) that can be used as a staging ground for XML business documents, workflows, business process management, Web service interaction logging, and microbilling. The XML OIS can also serve as the database, or information base of record for new applications, whether they be new Greenfield developments or derived from other Web services (see Figure 2).

A New Approach – Information and Patterns

To fully leverage XML, dynamic management of metadata, or tags, must take on the same importance as the management of data elements. This allows the structure of the information to change as well as the content. This is difficult, if not impossible, using traditional technologies and approaches, particularly in large-scale implementations. We need an approach that can represent metadata, data, or any combination of the two. This "information-centric" approach to XML persistence eliminates the impedance mismatch and inherent inefficiencies that exist with relational, object, or document-centric models. Furthermore, an information-centric approach allows for transactions with variable granularity, from the element level to the fragment, document, or cross-document level, regardless of the type of operation – be it query, update, or

the insertion or deletion of structure or content – without the performance degradation or resource footprint explosion common with traditional approaches. In order to achieve this level of granularity and take full advantage of XML's inherent capabilities, information must be viewed as arbitrary patterns, and the boundaries between metadata and data must be dynamic, determined by the use of the information at the time it is consumed, not at design time.

What's Missing for Web services?

A critical element of Web services is the need for more robust security, access control, and customization. One company accessing a Web service for catalog management may need to see a completely different view and set of controls than another company. However, the creator of the services can't afford to write customized services for each subscriber. By leveraging an information-centric approach, variable views and context- as well as content-based access control can be provided without code modifications. Simple XML-based business rules allow for the appropriate level of customization and control.

The self-constructing nature of the Operational Information Store allows it to support dozens of Web services without requiring the creation of additional "silos" of information. Aggregating legacy information into a common XML-based view reduces new application development efforts significantly, while lowering deployment and operational support costs. This allows corporate information assets to be leveraged across not only the enterprise, but also the enterprise's trading partners. As reliance on the older legacy systems is reduced through rewrites of older applications, and functionality is exposed via Web services, the legacy systems can slowly be "rusted out" without affecting new systems or applications that rely on the Web services, providing an incremental, nonstop migration strategy while reducing the fragility and risk of large-scale integration projects.

The OIS is a core infrastructure component in today's evolving IT model and should be easily accessible via current Internet protocols and communications mechanisms used by J2EE and .NET. In addition to supporting HTTP, RMI, EJB, and JMS, the OIS should itself be exposed as a Web service, providing XML persistence and management not as a monolithic application-specific solution, but as a service available across the entire IT organization.

Since XML is inherently heterogeneous, extensible, and flexible, the OIS can easily adapt to changing business require-

ments, new applications and services, and mass customization. This allows an enterprise to provide competitive advantage through improved customer care and service. The 70% of unstructured, unmanaged information assets can incrementally be rolled into the OIS and its information model, allowing the enterprise to leverage these assets in a low-risk, constantly evolving cycle, bringing the concept of the "Semantic Web" to the enterprise. In fact, the concept of the Semantic Web – the ability to intelligently tag and search what is now unstructured information, as currently found on the Web – will grow from the enterprise outward, and will also embrace current structured information. Standards efforts such as the Resource Description Framework (RDF) will help address the management of this combined informational view. The huge wall between current Web content and back-office corporate data will slowly be torn down, and a new, common model for managing all information assets will emerge.

For those who don't believe this, keep in mind that the Web didn't exist 10 years ago, but now contains over 17 trillion bytes of information. Imagine if this information, along with the hundreds of terabytes of corporate data, were effectively managed and available using a common mechanism, and exposed as Web services, available to anyone with the appropriate credentials? New technology is currently available that can extract and intelligently mark up much of this unstructured data, converting it into context- and content-searchable assets and unlocking its true potential. When this information is placed into an easily accessible, self-constructing Operation Information Store that leverages XML's flexibility, extensibility, and heterogeneity, a corporation gains competitive advantage, improves its decision making, and increases the quality of service it offers its customers, suppliers, vendors, and employees.

Next-Generation Apps Must Embrace XML's Full Potential

As the next generation of business applications – customer relations management, enterprise resource planning, supply chain management, and knowledge management – are developed and the applications are exposed as Web services, they will have to support the flexibility, extensibility, and heterogeneity inherent in XML. To understand why, let's take a closer look at each capability.

Flexibility allows data elements to be of any length and type. Even if a given business domain or problem defines an XML Schema to limit the length of a field, or its type, a different domain may have

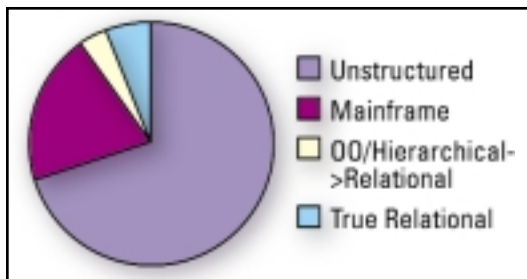


FIGURE 1 Where data is stored

THE LARGEST INDEPENDENT

JAVA

DEVELOPER CONFERENCE IN THE WORLD!

**WIN A
\$35,000
LUXURY CAR!**



ATTENDEES WILL BE INVITED TO TAKE A GOLF SWING
TO WIN AND RIDE OFF IN A \$35,000 LUXURY CAR!

Focus on Java

Java, now mainstream, is the dominant back-end technology upon which next-generation technologies are evolving.

Hear from the leading minds in Java how this essential technology offers robust solutions to technology professionals and senior IT/IS strategy decision makers.

The Java Fast Paths on June 24, a Java-focused CEO Keynote Panel, and comprehensive conference sessions will focus you on Java every information-packed day!



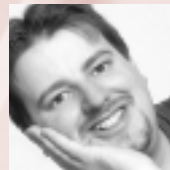
JAVA IN JUNE ESPECIALLY IN NEW YORK REGISTER ONLINE TODAY

FOR LOWEST CONFERENCE RATES
EARLY SELL-OUT GUARANTEED!

VISIT WWW.SYS-CON.COM

A Sampling of Java-Focused Sessions

- JAVA 1.4: WHAT'S NEW?
- BUILDING TRULY PORTABLE J2EE APPLICATIONS
- JAVA TOOLS FOR EXTREME PROGRAMMING
- BUILDING ASYNCHRONOUS APPLICATIONS USING JAVA MESSAGING
- .NET VS. J2EE
- J2EE: SETTING UP THE DEVELOPMENT ENVIRONMENT
- BUILDING WEB SERVICES WITH J2EE
- DETECTING, DIAGNOSING, AND OVERCOMING THE FIVE MOST COMMON J2EE APPLICATION PERFORMANCE OBSTACLES



ALAN WILLIAMSON
JAVA CHAIR • EDITOR-IN-CHIEF
JAVA DEVELOPER'S JOURNAL

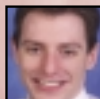
Featuring...

- UNMATCHED KEYNOTES AND FACULTY
- THE LARGEST INDEPENDENT JAVA, WEB SERVICES, AND XML EXPOS
- AN UNPARALLELED OPPORTUNITY TO NETWORK WITH OVER 5,000 I-TECHNOLOGY PROFESSIONALS

Who Should Attend...

- DEVELOPERS, PROGRAMMERS, ENGINEERS
- I-TECHNOLOGY PROFESSIONALS
- SENIOR BUSINESS MANAGEMENT
- SENIOR IT/IS MANAGEMENT/C LEVEL EXECUTIVES
- ANALYSTS, CONSULTANTS

Hear these thought leaders in interactive, cutting-edge keynote addresses and panels...



TYLER JEWELL
PRINCIPAL TECH.
EVANGELIST • BEA



GREGG KIESSLING
CEO
SITRKA



DAVID LITWACK
CEO
SILVERSTREAM



BARRY MORRIS
CEO
IONA



GREGG O'CONNOR
PRESIDENT
SONIC SOFTWARE



RICK ROSS
FOUNDER
JAVALOBBY



JAMES DUNCAN
DAVIDSON
FATHER OF ANT

For Exhibit information

CONTACT: MICHAEL PESICK
135 CHESTNUT RIDGE RD.
MONTVALE, NJ 07645
201 802-3057
MICHAEL@SYS-CON.COM

JDJEDGE
conference & expo

JUNE 24-27

JACOB JAVITS
CONVENTION CENTER
NEW YORK, NY

OCTOBER 1-3

SAN JOSE
CONVENTION CENTER
SAN JOSE, CA

SPONSORED BY:



MEDIA SPONSORS



OWNED AND PRODUCED BY



JAVA AND JAVA-BASED MARKS ARE TRADEMARKS OR REGISTERED TRADEMARKS OF SUN MICROSYSTEMS, INC., IN THE UNITED STATES AND OTHER COUNTRIES. SYS-CON PUBLICATIONS, INC., IS INDEPENDENT OF SUN MICROSYSTEMS, INC. ALL BRAND AND PRODUCT NAMES USED ON THESE PAGES ARE TRADE NAMES, SERVICE MARKS, OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES.

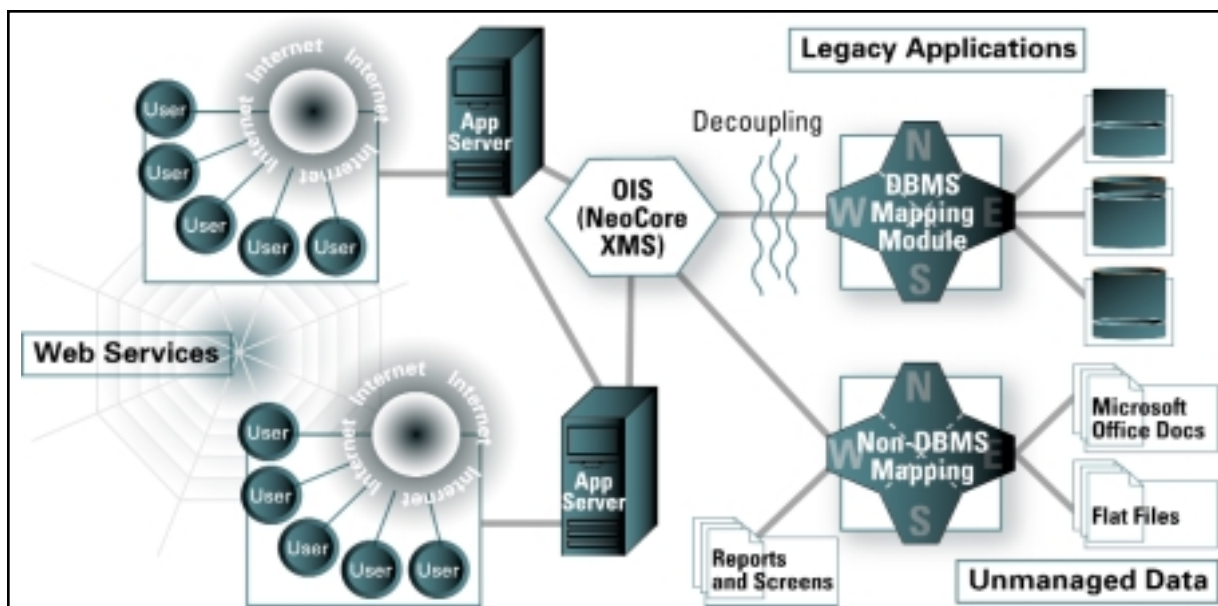


FIGURE 2 XML-based operational data store: aggregate view of structured and unstructured information

different needs, yet share the same underlying information. This is often handled today by creating multiple silos and replicating essentially the same information, a costly and error-prone practice. If the underlying technology can support flexibility but the applications still need constraints, the same information can be shared, eliminating silos, replication, and errors. This also allows for quicker application design and development, as a priori knowledge at every level of detail is no longer needed, and change occurs naturally and quickly, driven by the underlying business requirements.

Extensibility allows for schema evolution. Business needs change and with change comes maintenance costs. Web services, XML, and appropriate information management technology, when properly applied, allow an organization to embrace change, not avoid it, thereby improving return on investment and customer service. Business objectives are achieved, not constrained by the technology.

For example, company A supports a class of XML documents representing customer shipments that conform to version 1.0 of an industry-accepted XML Schema. Company A currently receives these shipment documents from companies B and C. C, however, is moving to an updated system that conforms to version 2.0 of the schema, which adds several new fields. Several approaches can be taken. First, the documents can be versioned and the system can accept and support both versions simultaneously (see "heterogeneity" below). Alternatively, all current version 1.0 documents could be structurally modified in place, migrating them to version 2.0 without bringing the

system offline and performing database schema changes and data migration. Consumers of version 1.0 documents could still see these as 1.0 documents through a simple, customized view.

Finally, *heterogeneity* enables the necessary customization to support a multitude of applications, vendors, suppliers, and customers. There are several layers to heterogeneity, and this can be challenging to those steeped in traditional technology. However, heterogeneity can have significant benefits. The first layer, as mentioned earlier, involves the hundreds of XML Schemas created by various industry groups – in some cases creating overlap and compounding the problem they intended to solve; how will corporations deal with the heterogeneity that all of the schemas bring without new approaches and technology?

Next, within a given application – purchasing for instance – different customers have different requirements, and there's no reason that the purchase orders for one customer have to be forced into a one-size-fits-all predefined schema. If a customer's purchase order requires an alternative shipping address or contact, then the purchase orders created for that customer, and only that customer, can contain that information. This flexibility can even be supported on an order-by-order basis without difficulty.

Finally, purchase orders for a given class of items may not need to look the same as for a different class of item. For instance, purchasing electronic components may be slightly different than purchasing plumbing supplies. Again, there's no need to force-fit everything into one model. Having underlying services that

can support this goes a long way toward lowering development and maintenance costs and embracing ever-changing and differing customer needs.

Flexibility, extensibility, and heterogeneity can also go a long way toward easing the implementation of Web service components such as UDDI, ebXML, UBL (Universal Business Language), DSML (Directory Services Markup Language), and a dozen other initiatives. The versions, schemas, document formats, and business objects behind these initiatives are constantly evolving, making implementation using traditional rigidly structured technology difficult, error-prone, slow, and expensive. A more flexible approach allows various development groups to simultaneously and iteratively develop these initiatives without constant database design/-redesign, rewriting of mappings or mapping code, migrating data between versions and iterations, and tuning physical database layouts to achieve acceptable performance, only to find that the next round of changes places them back in square one.

If Web services are to be the revolution the industry talks about, there must be fundamental changes in the way we manage information assets, build, deploy, and operate these services. Issues around security, performance, customization, usage tracking and billing, and rapid development and deployment must all be addressed. If we fully leverage XML's capabilities through the use of new technologies and methods, these issues are no longer insurmountable and the true power of the next-generation Web will indeed unfold. ☼

THE LARGEST INTERNATIONAL

WEB SERVICES CONFERENCE & EXPO IN THE WORLD!

**WIN A
\$35,000
LUXURY CAR!**



ATTENDEES WILL BE INVITED TO TAKE A GOLF SWING
TO WIN AND RIDE OFF IN A \$35,000 LUXURY CAR!

WEB SERVICES SKILLS, STRATEGY, AND VISION

REGISTER ONLINE TODAY

FOR LOWEST CONFERENCE RATES
EARLY SELL-OUT GUARANTEED!

VISIT WWW.SYS-CON.COM

Focus on Web Services

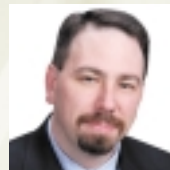
Web services, the next-generation technology that will enable the Internet to work for you and your business, and finally provide that ROI you have been after, will be put under a microscope at Web Services Edge East 2002.

Information-packed sessions, exhibits, and tutorials will examine Web services from every angle and will provide cutting-edge solutions and a glimpse at current and future implementations. Hear from the innovators and thought leaders in Web services. Enjoy a highly interactive CEO Keynote panel that will debate and discuss the realities and promise of Web services.



A Sampling of Web Services-Focused Sessions

- PRACTICAL EXPERIENCES WITH WEB SERVICES AND J2EE
- STATE OF THE WEB SERVICES INDUSTRY
- THE ODD COUPLE: MAKING .NET AND J2EE WORK TOGETHER
- EXPLORING THE .NET MY SERVICES INITIATIVE
- STANDARDS WATCH
- GUARDING THE CASTLE: SECURITY & WEB SERVICES



SEAN RHODY
CONFERENCE TECH CHAIR
WEB SERVICES TRACK CHAIR
EDITOR-IN-CHIEF
WEB SERVICES JOURNAL

Featuring...

- UNMATCHED KEYNOTES AND FACULTY
- THE LARGEST INDEPENDENT JAVA, WEB SERVICES, AND XML EXPOS
- AN UNPARALLELED OPPORTUNITY TO NETWORK WITH OVER 5,000 I-TECHNOLOGY PROFESSIONALS

Who Should Attend...

- DEVELOPERS, PROGRAMMERS, ENGINEERS
- I-TECHNOLOGY PROFESSIONALS
- SENIOR BUSINESS MANAGEMENT
- SENIOR IT/IS MANAGEMENT/C LEVEL EXECUTIVES
- ANALYSTS, CONSULTANTS

Hear these thought leaders in interactive, cutting-edge keynote addresses and panels...



JEAN-FRANCOIS ABRAMATIC
SENIOR VP R&D, ILOG
FORMER CHAIRMAN, W3C



DAVE CHAPPELL
VP CHIEF TECHNOLOGY EVANGELIST
SONIC SOFTWARE



GREGG KIESSLING
CEO
SITRAKA



ANNE THOMAS MANES
CTO
SYSTINET



BARRY MORRIS
CEO
IONA



ERIC RUDDER
SENIOR VP OF DEVELOPER
AND PLATFORM EVANGELISM
MICROSOFT



PATRICIA SEYBOLD
FOUNDER & CEO
SEYBOLD

For Exhibit information

CONTACT: MICHAEL PESICK
135 CHESTNUT RIDGE RD.
MONTVALE, NJ 07645
201 802-3057
MICHAEL@SYS-CON.COM

web services **EDGE**
conference & expo

JUNE 24-27
JACOB JAVITS
CONVENTION CENTER
NEW YORK, NY

OCTOBER 1-3
SAN JOSE
CONVENTION CENTER
SAN JOSE, CA

SPONSORED BY:



MEDIA SPONSORS



JAVA AND JAVA-BASED MARKS ARE TRADEMARKS OR REGISTERED TRADEMARKS OF SUN MICROSYSTEMS, INC., IN THE UNITED STATES AND OTHER COUNTRIES. SYS-CON PUBLICATIONS, INC. IS INDEPENDENT OF SUN MICROSYSTEMS, INC. ALL BRAND AND PRODUCT NAMES USED ON THESE PAGES ARE TRADE NAMES, SERVICE MARKS, OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES.

OWNED AND PRODUCED BY





The middle tiers in the XDBMS architecture can be confusing

Middle-TierDataManagement

XML databases are different from traditional databases, and they require a new set of features and metrics for evaluating them. In my last column (*XML-J*, Vol. 3, issue 2) I talked about native XML database management systems (XDBMS), and I'd like to follow up with how they differ from traditional databases and why this is significant.

AUTHOR BIO

Coco Jaenicke was, until recently, the XML evangelist and director of product marketing for eXcelon Corporation. She played a key role in the successful development and introduction of eXcelon, the industry's first application development environment for building and deploying e-business applications. Coco is currently an independent consultant.

When databases were first introduced to the industry eons ago, corporations had piles upon piles of data waiting to be poured into some form of digital storage and management. Not so with the introduction of XML – the biggest bottlenecks today are the piles upon piles of applications trying to effectively use all the available digital information.

Addressing this need is something that is square in the middle of XML's sweet spot. XML is flexible enough to manage any information (structured and unstructured) and extensible enough to support new and changing applications, and it is an Internet standard to cross IT boundaries. For all these reasons, XML database management systems do not usually act as the data-

base of record – although they certainly can – but more often are used for middle-tier data management.

What Is Middle-Tier Data Management?

The typical architecture within an enterprise consists of four layers: back-end systems, business logic, presentation and delivery, and the client application (see Figure 1). Most will agree that XML is the language of choice for delivering information to clients, and few are rushing to convert back-end systems to XML, but the two tiers in the middle present confusion.

Much has to happen to information in the middle tier(s). Most infrastructure is required to:

1. Convert diverse information (struc-

tured and unstructured) into XML.

2. Extract the relevant information (a custom view) for each application.
3. Deliver data in real time.
4. Connect to systems not under local control (i.e., systems from another division or a partner).
5. Support changes in applications, partners, and back-end systems.
6. Prevent localized changes from rippling throughout the extended enterprise.

These aren't easy tasks, but this is exactly what middle-tier data management addresses: it works with app servers and Web servers to make sure that the right data gets to the right place, in real time – and XML is the format of choice for the job.

How Is Middle-Tier Different from Back-End Data Management

The two are very different. Consider these distinctions:

RECEIVE \$150
DISCOUNT OFF FULL CONFERENCE
WEB SERVICES EDGE REGISTRATION

web services **EDGE**
world tour 2002

**Learn How to Develop
SOAP Web Services NOW!**
at a One-Day Seminar...Coming to a City Near You!

- **Primary purpose:** The primary role of a back-end data management system is to be the database of record, providing permanent and definitive storage regardless of how the data is eventually used. Middle-tier data management is the opposite. Its purpose is to serve the needs of the applications.

- **Data model:** The data model in a back-end database has a fixed schema and is usually highly typed and normalized. And it should be. If it's to serve as a permanent database of record, it should be predictable and solid as a rock. Again, middle-tier data management is the reverse – inflexibility is a liability here. The data is constantly changing to support new business initiatives, and an inextensible data model forces systems and applications to be hard-wired together, creating the ripple effect described above.

- **Duration of the data:** In back-end databases the data is usually stored forever. Invoices, customer profiles, and part numbers are maintained in the database long after the information stops being actively used. With middle-tier data management the data typically persists only for the duration of a process. While the process may last for several years, as in the case of a supply chain or an e-commerce site, durable data can be siphoned off and stored in a back-end system.

- **Development effort:** – Because a back-end database is designed to be permanent, significant development



FIGURE 1 Typical enterprise architecture

effort is invested upfront to anticipate all future needs. Beyond that, the development is mostly maintenance (connecting to new sources, schema evolution, etc.), and in general the less the better. This isn't necessarily the case at the northern end of the architecture diagram. Development is constant, and occurs every time the organization moves to become more competitive. Whether there's a new partner, a new system, or an additional feature, the data management system requires nearly continuous new development.

Is Middle-Tier Data Management Necessary?

That depends on how much the infrastructure is being stretched to deliver on the six requirements listed above. If data is strictly structured, if batch files

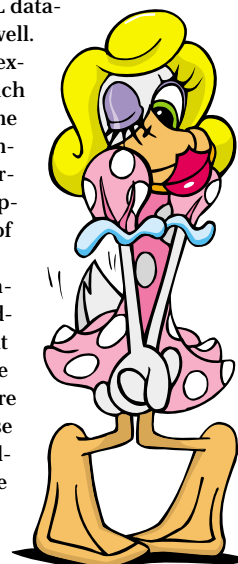
suffice, or if agility is not an issue, then it may not be necessary. The alternate ways of delivering XML to clients may be more appropriate, such as flat files or on-the-fly adapters. Flat files are flexible, and of course cheap, but everything is left as an exercise for the reader and there are no enterprise-class guarantees included. Adapters can turn non-XML data into an XML document, but they don't address any of the issues above.

If It Doesn't Walk Like a Duck, It Probably Won't Look Like a Duck

Because the usage models of middle-tier and back-end data management systems are quickly diverging, the feature

sets of relational and XML databases are diverging as well. XDBMSes depend on an extensible data model, which is considered heresy to the traditional DBA. Also, simple and standard interfaces, and connectivity options, jump to the top of the priority list.

It should be mentioned, however, that middle-tier data management is not the only way to use an XDBMS. XDBMSes are often used as the database of record or as an embedded database, but those are topics for another day. ☛

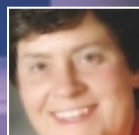


CJAENICKE @ ATTBI.COM

Jump-start your Web Services knowledge. Get ready for Web Services Edge East and West!

AIMED AT THE JAVA DEVELOPER COMMUNITY AND DESIGNED TO EQUIP ATTENDEES WITH ALL THE TOOLS AND INFORMATION TO BEGIN IMMEDIATELY CREATING, DEPLOYING, AND USING WEB SERVICES.

EXPERT PRACTITIONERS TAKING AN APPLIED APPROACH WILL PRESENT TOPICS INCLUDING BASE TECHNOLOGIES SUCH AS SOAP, WSDL, UDDI, AND XML, AND MORE ADVANCED ISSUES SUCH AS SECURITY, EXPOSING LEGACY SYSTEMS, AND REMOTE REFERENCES.



PRESENTERS...

Anne Thomas Manes, Systinet CTO, is a widely recognized industry expert who has published extensively on Web Services and service-based computing. She is a participant on standards development efforts at JCP, W3C, and UDDI, and was recently listed among the Power 100 IT Leaders by Enterprise Systems, which praised her "uncanny ability to apply technology to create new solutions."



Zdenek Svoboda is a Lead Architect for Systinet's WASP Web Services platform and has worked for various companies designing and developing Java and XML-based products.

EXCLUSIVELY SPONSORED BY



Register at www.sys-con.com or Call 201 802-3069

BOSTON, MA (Boston Marriott Newton) **SOLD OUT!** JANUARY 29
WASHINGTON, DC (Tysons Corner Marriott) **SOLD OUT!** FEBRUARY 26
NEW YORK, NY (Doubletree Guest Suites) **SOLD OUT!** MARCH 19
SAN FRANCISCO, CA (Marriott San Francisco)APRIL 19

REGISTER WITH A COLLEAGUE AND SAVE 15% OFF THE \$495 REGISTRATION FEE.



ADO.NET

What is ADO.NET?

It's the successor
to the Active Data Object
technology used in Classic ASP.

What is ASP.NET?

It's the successor
to Active Server Pages,
Microsoft's popular architecture
for writing server-side
Web applications.

ASP.NET applications run on servers that meet the following requirements:

- Windows NT 4.0 SP6a, Windows 2000, or Windows XP
- Internet Information Server 4.0 or later
- Microsoft .NET Framework 1.0 or later (downloadable from <http://msdn.microsoft.com/net/>)

ASP.NET applications are typically written in either of two equally powerful, object-oriented languages: VB.NET (the successor to Visual Basic) or C# (the successor to C++). In this article we'll use ASP.NET to write our applications.

ADO.NET implements several improvements:

- **Multitable data sets rather than single-table record sets:** In ADO.NET a data set can contain multiple tables of data, just like a real database. You can even execute SQL queries against the data in these data sets! In contrast, ADO record sets can store only a single table of information and work with it one row at a time.
- **Disconnected data sets versus connected record sets:** ADO record sets remained connected to the database, with the

ASP application moving a cursor to work with records one at a time. ADO.NET's disconnected data set approach retrieves the data once, eliminating the overhead of constantly paging data from the database.

- **Integrated support for XML:** ADO.NET data sets can be populated with XML data and can be structured according to an XML Schema. In addition, any data set can have its schema and data exported as XML.

This article introduces the basics of building XML applications using ASP.NET and ADO.NET. Specifically, we'll focus on:

- Producing XML and XML Schemas from data in a database
- Reading XML data into a data set and binding it to Web server controls

AUTHOR BIO

Steve Heckler has been an instructor and technologist at WestLake Internet Training (www.westlake.com) since 1995. WestLake teaches instructor-led, hands-on courses in .NET, XML, and other Web development technologies. Steve holds bachelor's and master's degrees from Stanford University.

- Reading XML data into a data set and inserting it into a database
- Other ASP.NET/ADO.NET XML capabilities

About the Database...

In this article we'll use a Microsoft Access database storing programs, viewers, and orders for the ".NET Network," a fictitious pay-per-view network aimed at programmers. The relationship among the three tables in the database is shown in Figure 1.

For production applications you'll want to use a "real" client/server database such as SQL Server or Oracle. For demonstration purposes Microsoft Access is entirely adequate.

Producing XML from Database Data via ADO.NET

ADO.NET makes it remarkably easy to produce XML files and schemas from data sets. As an example, see the application in Listing 1. The application produces XML and schema files from the data in the programs table and displays the page shown in Figure 2.

The ASP.NET application has retrieved the data in the programs table and has written it out to disk as Programs.xml (see Listing 1). As shown in the code, the XML has been written according to the following rules:

- The root node is NewDataSet.

Written by Steven Heckler

...How to use them to build XML applications

- The immediate child nodes of the root node match table names in the database. In our file, for example, all the immediate child nodes are instances of Programs. This node has the same name as the Programs table in the database.
- The child nodes of each table node (in this case the children of Programs) must bear names that match the corresponding column names in the database. Here our child nodes have the names programid, name, length, rating, price, and description.

In addition to writing the XML to disk, our application has also written out a schema for the data as Programs-Schema.xsd (see Listing 2).

As shown, the schema is very loose, with every element optional. If you use ADO.NET to generate a schema automatically, you may want to edit the schema to make it more restrictive.

The application that generated the schema, ProgramsAsXml.aspx, is shown in Listing 3. In the body of the page two hyperlinks, xmllink and schemalink, will point to the XML and schema files. Their URLs are set in the Page_Load() subroutine, which is executed after the ASP.NET application has been loaded into memory and all objects in the body of the page have been instantiated.

Specifying the Table Name and SQL Statement

Within Page_Load() the first two statements assign the table name and the SQL statement used to retrieve the programs to variables:

```
Dim TableName as String = "Programs"

Dim strSQL as String = "SELECT * FROM " + TableName
```

Creating the Connection String and OleDbConnection Object

The next two statements create a connection string, then use it to create an instance of OleDbConnection (a connection to our data source):

```
Dim strConnection as String =
"Provider=Microsoft.Jet.OLEDB.4.0;"
strConnection += "Data Source=" +
Server.MapPath("../payperview.mdb")

Dim objConnection as New OleDbConnection(strConnection)
```

Creating a DataAdapter and DataSet

In the next three statements:

```
Dim objAdapter as New OleDbDataAdapter(strSQL,
strConnection)

Dim objDataSet as New DataSet()

objAdapter.Fill(objDataSet, tableName)
```

the first statement creates a new instance of OleDbDataAdapter and assigns it to objAdapter. The data adapter is used in ADO.NET to retrieve the results of a specific query (here, strSQL) from a specific connection (here, strConnection), then populate a data set.

The second and third statements instantiate an empty data set, then fill its Programs table with data from the adapter.

Writing the Data to Disk as XML

Once the data is loaded in a data set, we just call the data set's WriteXml() method to write the XML to disk:

```
Dim BaseFile as String = TableName + ".xml"
Dim FileName as String = Server.MapPath(BaseFile)
objDataSet.WriteXML(FileName)
xmllink.NavigateUrl = BaseFile
```

ASP.NET...

The last statement sets the URL of the hyperlink to the file that was just written to disk.

Writing the XML Schema to Disk

Writing the XML Schema to disk is equally easy: just call the data set's WriteXmlSchema() method:

```
BaseFile = TableName + "-Schema.xsd"
FileName = Server.MapPath(BaseFile)
objDataSet.WriteXmlSchema(FileName)
schemalink.NavigateUrl = BaseFile
```

Reading XML Data into a Data Set

The ReadXml() method of DataSet reads an XML file into a data set. Once the file has been read into the data set, it can be used to populate Web server controls or create records in the database.

For an example, please browse demos > BindXML.aspx. The page shown in Figure 3 is displayed. The source of the application is given in Listing 4.

objDataSet.ReadXml() reads the XML file at a given path into the data set. Once this is done, the data set can be used just as you would any other data set. In this case we're binding the data in the data set to XMLOutputGrid, an instance of the asp:datagrid server control, an ASP.NET component that displays data in an HTML table.

Importing XML into the Database

XML data can easily be read into a data set, then imported into a database. Although conformance with the standard Microsoft format is

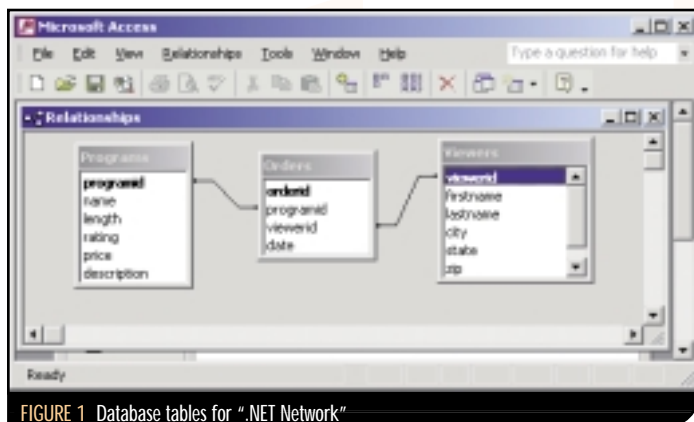


FIGURE 1 Database tables for ".NET Network"

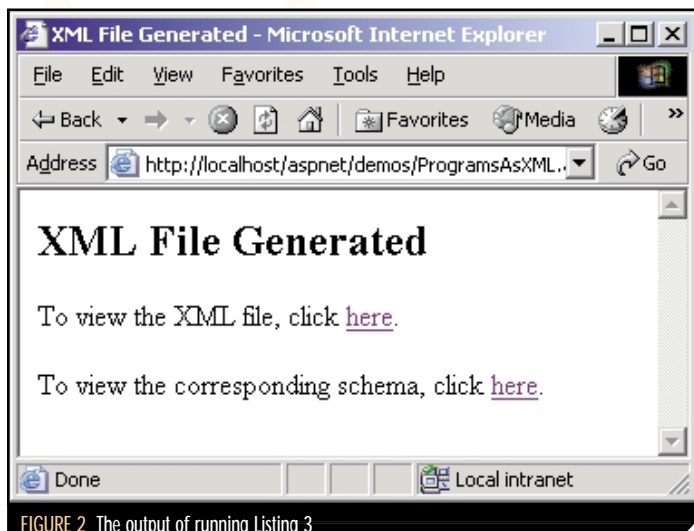


FIGURE 2 The output of running Listing 3

helpful, it isn't required. The data set will do its best to infer the schema of the data based on the XML it receives.

For an example, please examine the application in Figure 4. The programs are read from an XML file and inserted into the database (note the presence now of duplicate records indicating that the records have been imported). The source of the application is given in Listing 5.

After reading the XML file into the data set objXMLDataSet:

```
objXMLDataSet.ReadXml(FilePath)
```

we load the current records into the OleDbDataAdapter objAdapter:

```
Dim strSQL as String = "SELECT * FROM " + TableName
```

```
Dim strConnection as String = "Provider=Microsoft.Jet.OLEDB.4.0;"
strConnection += "Data Source=" +
Server.MapPath("../payperview.mdb")
```

```
Dim objConnection as New OleDbConnection(strConnection)
Dim objAdapter as New OleDbDataAdapter(strSQL, strConnection)
```

then insert the data from objXMLDataSet at the end of those records:

```
Dim objBuilder as New OleDbCommandBuilder(objAdapter)
objAdapter.InsertCommand = objBuilder.GetInsertCommand()
objAdapter.Update(objXMLDataSet, TableName)
```

The first statement creates an instance of OleDbCommandBuilder, which enables insert, update, and delete commands to be generated automatically. The second statement retrieves an appropriate Insert command and assigns it to the data adapter's InsertCommand property. If data in the data set were used to update or delete records in the database, we would have needed to include these statements too:

```
objAdapter.UpdateCommand = objBuilder.GetUpdateCommand()
objAdapter.DeleteCommand = objBuilder.GetDeleteCommand()
```

Once these properties are set, calling the data adapter's Update() method runs the specified inserts, updates, and deletes. In this case we've specified that records are only to be inserted.

Once this is done, just to confirm that the records have been inserted, we retrieve all records in the table and bind them to ProgramsGrid, an instance of the asp:datagrid server control:

```
Dim strSQLSelect as String = "SELECT * FROM " + TableName + "
ORDER BY programid"
Dim objDataSet as New DataSet()
```

```
Dim objAdapter2 as New OleDbDataAdapter(strSQLSelect,
strConnection)
objAdapter2.Fill(objDataSet, TableName)
ProgramsGrid.DataSource = objDataSet.Tables(TableName).DefaultView
ProgramsGrid.DataBind()
```

Specifying an XML Schema for the Data Set

Specifying an XML schema for the data set in the preceding example would have been easy:

```
objXMLDataSet.ReadXmlSchema("your_schema_here.xsd")
```

```
objXMLDataSet.ReadXml(FilePath)
```

Once the schema is read in, any XML files you read into the data set will be presumed to conform to the schema.

If you wish to perform detailed validation of the XML against a schema, you should use an instance of the XmlValidatingReader class.

THE LARGEST INTERNATIONAL

XML CONFERENCE & EXPO IN THE WORLD!

**WIN A
\$35,000
LUXURY CAR!**



ATTENDEES WILL BE INVITED TO TAKE A GOLF SWING
TO WIN AND RIDE OFF IN A \$35,000 LUXURY CAR!

XML-NEXT G OF ENTERPRISE DEPLOYMENT REGISTER ONLINE TODAY

FOR LOWEST CONFERENCE RATES
EARLY SELL-OUT GUARANTEED!

VISIT WWW.SYS-CON.COM

Focus on XML

XML is today's essential technology to develop and deploy Web services. Here's your chance to learn from expert practitioners how XML is making the next phase of the Web a reality.

Focus on standards, interoperability, content management, and today's new quest for more efficient and cost-effective Internet and intranet-enabled business process integration.

Focus on XML during information-packed days while you enjoy an XML/Web Services Keynote panel, comprehensive conference sessions, and an unparalleled opportunity to exchange ideas with peers and industry leaders.



A Sampling of XML-Focused Sessions

- XML STANDARDS - AN OVERVIEW
- OASIS STANDARDS UPDATE
- UBL - A UNIVERSAL BUSINESS LANGUAGE
- BRINGING XML TO PKI
- LINK MANAGEMENT WITH XLINK
- PRACTICAL XSLT AND XPATH
- ENTERPRISE CONTENT MANAGEMENT WITH XML
- XML IN THE ENTERPRISE AND INTER-ENTERPRISE WORLD



NORBERT MIKULA
XML CHAIR
BOARD OF DIRECTORS, OASIS
INDUSTRY EDITOR
WEB SERVICES JOURNAL

Featuring...

- UNMATCHED KEYNOTES AND FACULTY
- THE LARGEST INDEPENDENT JAVA, WEB SERVICES, AND XML EXPOS
- AN UNPARALLELED OPPORTUNITY TO NETWORK WITH OVER 5,000 I-TECHNOLOGY PROFESSIONALS

Who Should Attend...

- DEVELOPERS, PROGRAMMERS, ENGINEERS
- I-TECHNOLOGY PROFESSIONALS
- SENIOR BUSINESS MANAGEMENT
- SENIOR IT/IS MANAGEMENT/C LEVEL EXECUTIVES
- ANALYSTS, CONSULTANTS

Hear these thought leaders in interactive, cutting-edge keynote addresses and panels...



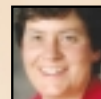
JEAN-FRANCOIS ABRAMATIC
SENIOR VP R&D, ILOG
FORMER CHAIRMAN, W3C



TYLER JEWELL
PRINCIPAL TECH.
EVANGELIST • BEA



DAVID LITWACK
CEO
SILVERSTREAM



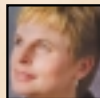
ANNE THOMAS MANES
CTO
SYSTINET



BARRY MORRIS
CEO
IONA



RICK ROSS
FOUNDER
JAVALOBBY



PATRICIA SEYBOLD
FOUNDER & CEO
SEYBOLD

For Exhibit information

CONTACT: RICHARD ANDERSON
135 CHESTNUT RIDGE RD.
MONTVALE, NJ 07645
201 802-3056
RICHARD@SYS-CON.COM

XMLEDGE
conference & expo

JUNE 24-27

JACOB JAVITS
CONVENTION CENTER
NEW YORK, NY

OCTOBER 1-3

SAN JOSE
CONVENTION CENTER
SAN JOSE, CA

SPONSORED BY:



MEDIA SPONSORS



OWNED AND PRODUCED BY



JAVA AND JAVA-BASED MARKS ARE TRADEMARKS OR REGISTERED TRADEMARKS OF SUN MICROSYSTEMS, INC., IN THE UNITED STATES AND OTHER COUNTRIES. SYS-CON PUBLICATIONS, INC. IS INDEPENDENT OF SUN MICROSYSTEMS, INC. ALL BRAND AND PRODUCT NAMES USED ON THESE PAGES ARE TRADE NAMES, SERVICE MARKS, OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES.

programid	name	length	rating	price	description
1	Backyard Boys Concert	60	G	6.95	Hot young group performs live from Shoreline Amphitheater.
2	Dot Gone	115	R	7.95	Five twenty-somethings straight out of college attempt to start a company with \$20 million in venture capital and no experience.
3	Who Moved my Beach?	90	PG	4.95	After a surprise melting of the polar ice caps, Las Vegas blackjack dealer Vanessa Kim discovers her condo is now beachfront property.
4	Fort Pierced	120	PG13	4.95	Heavy metal bikers invade a small Florida town.
5	Ernest Goes to College	105	G	5.95	Ernest the Squirrel, after years of happily eating acorns on the campus of Haverford College, develops higher aspirations.
6	George W. Bush: The First 100 Days	120	NR	10.95	An inside look at the first 100 days of the Bush presidency.
7	The Steve WestLake Show	60	NR	0	Late-night talkshow features tech industry luminaries and their lighter sides.
8	America's Funniest Homebrew Software	60	NR	0	Programmers from throughout North America demonstrate their most humorous software creations.

FIGURE 3 HTML table generated by binding XML data to a DataGrid

programid	name	length	rating	price	description
1	Backyard Boys Concert	60	G	6.95	Hot young group performs live from Shoreline Amphitheater.
2	Dot Gone	115	R	7.95	Five twenty-somethings straight out of college attempt to start a company with \$20 million in venture capital and no experience.
3	Who Moved my Beach?	90	PG	4.95	After a surprise melting of the polar ice caps, Las Vegas blackjack dealer Vanessa Kim discovers her condo is now beachfront property.
4	Fort Pierced	120	PG13	4.95	Heavy metal bikers invade a small Florida town.
5	Ernest Goes to College	105	G	5.95	Ernest the Squirrel, after years of happily eating acorns on the campus of Haverford College, develops higher aspirations.
6	George W. Bush: The First 100 Days	120	NR	10.95	An inside look at the first 100 days of the Bush presidency.
7	The Steve WestLake Show	60	NR	0	Late-night talkshow features tech industry luminaries and their lighter sides.
8	America's Funniest Homebrew Software	60	NR	0	Programmers from throughout North America demonstrate their most humorous software creations.

FIGURE 4 Programs read from XML file into database

LISTING 1

```
<?xml version="1.0" standalone="yes"?>
<NewDataSet>
  <Programs>
    <programid>1</programid>
    <name>Backyard Boys Concert</name>
    <length>60</length>
    <rating>G</rating>
    <price>6.95</price>
    <description>Hot young group performs live from
      Shoreline Amphitheater.</description>
  </Programs>
  <Programs>
    <programid>2</programid>
    <name>Dot Gone</name>
    <length>115</length>
    <rating>R</rating>
    <price>7.95</price>
    <description>Five twenty-somethings straight out of
      college attempt to start a company with $20 million
      in venture capital and no experience.</description>
  </Programs>
  <Programs>
    <programid>3</programid>
    <name>Who Moved my Beach?</name>
    <length>90</length>
    <rating>PG</rating>
    <price>4.95</price>
    <description>After a surprise melting of the polar ice
      caps, Las Vegas blackjack dealer Vanessa Kim discovers
      her condo is now beachfront property.</description>
  </Programs>
  <Programs>
    <programid>4</programid>
    <name>Fort Pierced</name>
    <length>120</length>
    <rating>PG13</rating>
```

Other Uses of ASP.NET and ADO.NET with XML

ASP.NET and ADO.NET can work together to provide a wide array of additional capabilities to your programs. For example, ADO.NET data sets are serializable and can be passed readily to and from Web services. Web services written in ASP.NET can receive a data set from a calling application, process the data, and even return a data set with updated records.

ASP.NET's System.Xml.Xsl namespace contains classes that enable you to transform XML data via an XSL stylesheet. These XSL transformations can be useful for converting XML data from one schema to another, or to HTML for output to a browser.

The XmlDataDocument class is a subclass of XmlDocument, the ASP.NET class that provides DOM (Document Object Model) access to an XML document. XmlDataDocument provides powerful methods for manipulating XML data via XPath statements, then synchronizing the data with a DataSet.

Suggestions for Further Learning

Magazines

- **XML-Journal** and **Web Services Journal** are both great ways to keep up with the latest technologies for building XML applications and Web services in ASP.NET and other languages. Visit www.sys-con.com.


Books

Wrox Press (www.wrox.com) publishes several excellent books on ASP.NET and ADO.NET that include coverage of XML applications:

For beginners:

- Sussman, D., et al. (2001). *Beginning ASP.NET Using C# and Beginning ASP.NET Using VB.NET*.

For more advanced ASP.NET developers:

- Hasan, J., et al. (2001). *ADO.NET Programmer's Reference*.
- Homer, A., et al. (2001). *Professional ASP.NET*.
- Reynolds, M. (2001). *Professional ASP.NET Web Services*. 

SHECKLER @ WESTLAKE.COM

```
<price>6.95</price>
<description>Heavy metal bikers invade a small Florida
  town.</description>
</Programs>
<Programs>
  <programid>5</programid>
  <name>Ernest Goes to College</name>
  <length>105</length>
  <rating>G</rating>
  <price>5.95</price>
  <description>Ernest the Squirrel, after years of happily
    eating acorns on the campus of Haverford College,
    develops higher aspirations.</description>
</Programs>
<Programs>
  <programid>6</programid>
  <name>George W. Bush: The First 100 Days</name>
  <length>120</length>
  <rating>NR</rating>
  <price>10.95</price>
  <description>An inside look at the first 100 days of
    the Bush presidency.</description>
</Programs>
<Programs>
  <programid>7</programid>
  <name>The Steve WestLake Show</name>
  <length>60</length>
  <rating>NR</rating>
  <price>0</price>
  <description>Late-night talkshow features tech industry
    luminaries and their lighter sides.</description>
</Programs>
</NewDataSet>
```

LISTING 2

```
<?xml version="1.0" standalone="yes"?>
<xs:schema id="NewDataSet" xmlns="" xmlns:xs=
```

```

"http://www.w3.org/2001/XMLSchema"
xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
  <xs:element name="NewDataSet" msdata:IsDataSet="true">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded">
        <xs:element name="Programs">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="programid" type="xs:int"
                minOccurs="0" />
              <xs:element name="name" type="xs:string"
                minOccurs="0" />
              <xs:element name="length" type="xs:short"
                minOccurs="0" />
              <xs:element name="rating" type="xs:string"
                minOccurs="0" />
              <xs:element name="price" type="xs:decimal"
                minOccurs="0" />
              <xs:element name="description" type="
                xs:string" minOccurs="0" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:choice>
    </xs:complexType>
  </xs:element>
</xs:schema>

```

LISTING 3

```

<%@ import Namespace="System.Data" %>
<%@ import Namespace="System.Data.OleDb" %>

<script language="vb" runat="server">
sub Page_Load()

  Dim TableName as String = "Programs"

  Dim strSQL as String = "SELECT * FROM " + TableName

  Dim strConnection as String =
    "Provider=Microsoft.Jet.OLEDB.4.0;"
  strConnection += "Data Source=" + Server.MapPath
    ("../payperview.mdb")

  Dim objConnection as New OleDbConnection(strConnection)
  Dim objAdapter as New OleDbDataAdapter
    (strSQL, strConnection)

  Dim objDataSet as New DataSet()

  objAdapter.Fill(objDataSet, tableName)

  'Write the XML file and set the corresponding link
  Dim BaseFile as String = TableName + ".xml"
  Dim FileName as String = Server.MapPath(BaseFile)
  objDataSet.WriteXML(FileName)
  xmllink.NavigateUrl = BaseFile

  'Write the schema file and set the corresponding link
  BaseFile = TableName + "-Schema.xsd"
  FileName = Server.MapPath(BaseFile)
  objDataSet.WriteXMLSchema(FileName)
  schemalink.NavigateUrl = BaseFile

end sub
</script>
<html>
<head><title>XML File Generated</title></head>
<body>
<h2>XML File Generated</h2>
<p>To view the XML file, click <asp:hyperlink id="xmllink"
text="here"
runat="server" />.</p>
<p>To view the corresponding schema, click <asp:hyperlink id=
"schemalink" text="here" runat="server" />.</p>

</body>
</html>

```

LISTING 4

```

<%@ import Namespace="System.Data" %>
<%@ import Namespace="System.Data.OleDb" %>

```

```

<script language="vb" runat="server">
sub Page_Load()
  Dim TableName as String = "Programs"
  Dim BaseFile as String = TableName + ".xml"
  filename.text = BaseFile

  Dim FilePath as String = Server.MapPath(BaseFile)
  Dim objDataSet as New DataSet()

  objDataSet.readXml(FilePath)
  XmlOutputGrid.DataSource = objDataSet.Tables
    (TableName).DefaultView
  XmlOutputGrid.DataBind()
end sub
</script>
<html>
<head><title>XML File Generated</title></head>
<body>
<h2>Contents of <asp:label id="filename" runat=
  "server" /></h2>
<asp:datagrid id="XmlOutputGrid" runat="server">
<HeaderStyle font-name="arial black,arial" />
</asp:datagrid>
</body>
</html>

```

LISTING 5

```

Namespace="System.Data" %>
<%@ import Namespace="System.Data.OleDb" %>

<script language="vb" runat="server">
sub Page_Load()
  Dim TableName as String = "Programs"
  Dim BaseFile as String = TableName + ".xml"

  Dim FilePath as String = Server.MapPath(BaseFile)
  Dim objXMLDataSet as New DataSet()

  objXMLDataSet.readXml(FilePath)

  Dim strSQL as String = "SELECT * FROM " + TableName

  Dim strConnection as String =
    "Provider=Microsoft.Jet.OLEDB.4.0;"
  strConnection += "Data Source=" + Server.MapPath
    ("../payperview.mdb")

  Dim objConnection as New OleDbConnection(strConnection)
  Dim objAdapter as New OleDbDataAdapter
    (strSQL, strConnection)

  Dim objBuilder as New OleDbCommandBuilder(objAdapter)
  objAdapter.InsertCommand = objBuilder.GetInsertCommand()
  objAdapter.Update(objXMLDataSet, TableName)

  Dim strSQLSelect as String = "SELECT * FROM
  " + TableName + " ORDER BY programid"
  Dim objDataSet as New DataSet()

  Dim objAdapter2 as New OleDbDataAdapter
    (strSQLSelect, strConnection)
  objAdapter2.Fill(objDataSet, TableName)
  ProgramsGrid.DataSource = objDataSet.Tables
    (TableName).DefaultView
  ProgramsGrid.DataBind()

  dbtablename.text = TableName
end sub
</script>
<html>
<head><title>Data Imported</title></head>
<body>
<h2>Contents of <asp:label id="dbtablename" runat=
  "server" /></h2>
<asp:datagrid id="ProgramsGrid" runat="server">
<HeaderStyle font-name="arial black,arial" />
</asp:datagrid>
</body>
</html>

```




Is this a second chance for the industry?

A Pragmatic Convergence of ebXML & Web Services

The 18-month-old Electronic Business XML project ended officially in Vienna this past May 11, 2001. ebXML was an ambitious and high-profile initiative of two neutral industry groups: OASIS and the United Nations' Centre for Trade Facilitation and Electronic Business (UN/CEFACT).

What was the mission of ebXML? According to the ebXML Web FAQ: "ebXML is a modular suite of specifications that enables enterprises of any size and in any geographical location to conduct business over the Internet. Using ebXML, companies now have a standard method to exchange business messages, conduct trading relationships, communicate data in common terms, and define and register business processes."

As this is clearly a worthwhile goal, scores of technology vendors participated in the ebXML project in one way or another. ebXML had many parallel working teams (e.g., Requirements, Registry/-Repository, Transport/Routing, Security, Business Processes, Proof-of-Concept). Some companies led the effort to write specifications; my own company, Kildara, focused on the live Proof-of-Concept interoperability demos.

I won't pretend to be an expert in any of these areas. However, 20 years of building software products has made me an expert on implementing standards, and I'm afraid that ebXML has not succeeded in defining a standard in all of these areas. A standard is concise, tightly structured, implementable. Conformance to a standard can be tested and certified. Some parts of ebXML (e.g., Transport/Routing) approach this, but other parts don't.

The publication of a simple, concrete set of XML-based business messaging standards would have helped the industry immeasurably. Until this standard emerges, we won't see the wildfire build-out of commercial XML networks that we all want.

The Fax Analogy

Before studying engineering, I studied history. Whenever I'm confused, I instinctively revert to a longer-term perspective. Historically, was there an electronics standard comparable to ebXML? I found one that seemed a very good fit: the fax machine. Like ebXML, the goal of the fax was to enable organizations to communicate faster, better, and cheaper. We all take fax machines for granted – they can talk to each other globally, cheaply, and effortlessly. But the evolution of the underlying fax machine standards is deeply instructive:

1. **May 1865:** Twenty European countries agree to resolve telegraph technical issues through the International Telegraph Union (ITU).
2. **1968:** ITU (CCITT) publishes Group 1 facsimile. Six minutes to transmit one page.
3. **1976:** The Group 2 standard halved the time of transmission to three minutes.
4. **1980:** Group 3 took less than one minute per page. Effective compatibility between all vendors.

I was surprised to find that the emergence of the fax machine took so long. In the technology business we tend to dwell on the pleasures of the future and ignore the pains of the past. If we take Group 3 fax as the first widespread interoperable fax standard, it took 12 years to achieve – not 18 months. And the fax standard – which simply sends an IMAGE of a document – is much simpler than sending a structured, normalized electronic document in XML.

Fax machines made the world a better place just by improving the speed of delivering business messages; Web services will match this and also improve our ability to comprehend and manipulate the messages mechanically (i.e., in software).

Hubris

It's appropriate at this point to introduce the ancient Greek word *hubris*. Hubris is defined as exaggerated pride, often attracting vengeance from the gods. As the founder of tech companies that have attracted venture funding, I am familiar with this vice. Nobody will invest in a company that doesn't have hubris.

However, I am guilt-free because nobody believes a startup when they make extravagant claims. However, industry groups are held to a higher standard. ebXML has done the industry a disservice by promising a set of comprehensive standards but delivering a set of academic working papers or simple rubber-stamping of more useful vendor-sponsored standards.

The "tech wreck" of 2000 has shown us the brutal reality of what happens when we overpromise technology. When all the dust settles, the real villain in this piece is ourselves: smart technologists who have told society at large that we can rebuild the world in 18 months instead of 12 years. We promise to change the world quickly, dramatically, before the media and the investors get bored. As an industry, we have just begun to pay for these excesses.

Avoiding Hubris in Web Services

Web services is still young as a paradigm. We still have a few months to correct our mistakes, rein in the media, and realign what we are doing with reality. This is a real second chance for the industry.

AUTHOR BIO

John Ogilvie, founder/CTO of Kildara, has founded three companies in his 20-year career as an engineer, and was a developer of one of the first commercial Web browsers, Pythia. John is a frequent speaker and panelist at industry events and leading XML and Web services conferences.

JDJSTORE.COM GUARANTEED! LOWEST PRICES!

Guaranteed Best Prices

JDJ Store Guarantees the Best Prices. If you see any of our products listed anywhere at a lower price, we'll match that price and still bring you the same quality service.

Terms of offer:

- Offer good through June 30, 2002
- Only applicable to pricing on current versions of software
- Offer does not apply toward errors in competitors' printed prices
- Subject to same terms and conditions

Prices subject to change.
Not responsible for typographical errors.

Attention Software Vendors:

To include your product in JDJStore.com, please contact tony@sys-con.com

MICROSOFT MicroSoft Visual Studio .NET 2002 Professional



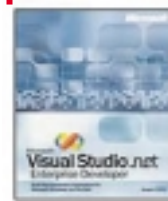
JDJStore.com **\$999⁰⁰**



\$79⁰⁰

SYS-CON MEDIA
WebServices Journal -
XML-Journal Resource CD

MICROSOFT MicroSoft Visual Studio .NET 2002 Enterprise Developer



JDJStore.com **\$1629⁰⁰**

MACROMEDIA

ColdFusion Server 5 Professional Edition

Macromedia® ColdFusion® 5 empowers Web developers with a highly productive solution for building and delivering the next generation of Web applications. ColdFusion 5 offers major enhancements in development, administration, and performance.



ColdFusion Server 5. **\$1219⁹⁹**

INSTALLSHIELD

InstallShield Professional 6.3

InstallShield Professional 6.3 is the de-facto industry standard for creating sophisticated Windows installations. InstallShield Professional 6.3 offers scripting and visual tools in an integrated environment that gives users complete flexibility and control while allowing unparalleled productivity.



InstallShield Professional 6.3 **\$999⁰⁰**

SYS-CON MEDIA

XML-Journal: The Complete Library

THE MOST COMPLETE LIBRARY OF EXCLUSIVE XML-J ARTICLES IN ONE CD
Features & Articles, Product Reviews
Case Studies, Tips & Tricks, Interviews,
Editorials, IMHOs & more!



XML-Journal: The Complete Library **\$53⁹⁹**

WEBGAIN

Visual Café Enterprise Suite

WebGain's VisualCafé Enterprise Edition delivers more power to enterprise Java developers with new technology that simplifies the development, verification, debugging and deployment of Enterprise Java Beans (EJB) plus new support for the EJB 1.1 specification including XML-based deployment descriptors.



FREE- The Complete Works CD
(Your choice: JDJ, XML-J, CFDJ or CFAdvisor)
Visual Café Enterprise Suite **\$2749⁰⁰**

BORLAND

Borland JBuilder Enterprise V6.0

JBuilder is the leading cross-platform environment for building enterprise Java applications. JBuilder 6 Enterprise makes EJB 2.0 development easy with two-way visual designers and rapid deployment to leading J2EE platform application servers. Enhance developer productivity with UML code visualization, refactoring, unit testing, documentation tools, and support for multiple version control systems.



FREE- The Complete Works CD
(Your choice: JDJ, XML-J, CFDJ or CFAdvisor)
Borland JBuilder Enterprise V6.0 **\$2749⁹⁹**

eHELP

RoboHELP Office 2002

Develop Help for your Java Apps with RoboHELP Office 2002. Just point and click to develop the Help content, structure and design no coding. Easily deploy these Help formats:

- WebHelp - Web, server and desktop-based cross-browser, cross-platform Help
- Oracle Help for Java 100% pure Java solution, not Oracle dependent
- JavaHelp - 100% pure Java solution



RoboHELP Office 2002 **\$898⁹⁹**

ORDER TODAY!

888-303-JAVA



It's better to have every vendor implementing the same imperfect standard than to have the industry in limbo waiting for the upcoming perfected standard



try. Web services is actually a horrific misnomer, since this is all very post-Web. The Web was a way for humans to access huge amounts of information from far-flung organizations. The information was crudely formatted (HTML), so it required the sophistication of a human reader to interpret it. The technologies we call Web services (XML, SOAP, UDDI, WSDL) are really about structuring this information so tightly that even the dumbest software can interpret it.

This is really a Very Good Thing. A seamless web of thousands of smart machines transmitting millions of routine business messages over the Internet is a foundation that we must build. This will improve our economy and society in ways that we barely imagine.

But this time around, we need to build it on stone, not sand.

- We need to admit publicly that it's going to take at least five years, not a few months.
- We need to advance incrementally, based on real-world demonstrations and interoperability testing.
- We need to be aggressively shallow, solving today's small problems (security, transaction reliability) before tackling tomorrow's larger problems (business processes).
- We need to recognize that the successful standards that we use today (HTTP, XML, HTML) came from the work of an individual or a small team, not a committee.
- We should be shameless in taking the best work coming out of the smartest corporations (Sun, Microsoft, IBM) and compelling them to sand off the rough, proprietary edges.

A Pragmatic Melding of Web Services and ebXML

The Web as we know it was developed (incredibly) by independent researchers like Tim Berners-Lee, not by corporations. That won't happen this time – there is too much at stake. While Tim is now focusing on designing the Semantic Web, commercial software vendors are already coming to market with Web services products. Reining in these commercial interests will be challenging enough without the distractions of projects like ebXML.

Since the formal end of the ebXML effort, industry leaders have been solic-

iting support for new initiatives to complete the work ebXML started, especially in the area of standardized content. With all due respect, these attempts to define a universally useful XML “Esperanto” are impractical and unnecessary. There are other areas higher up the protocol stack that are simply not ready for standardization anytime soon, period.

For example, the introduction of fax technology didn't impose universally standardized business forms, formalized business processes, or common nomenclature/vocabulary (e.g., product codes). We have a “good enough” fax network that we built simply by standardizing only the lowest-level transmission.

So let's be pragmatic. There is the de facto consensus view of how to build out basic Web services (see Table 1). The ebXML architects pointed the way in this by abandoning their own attempts to define new transport/routing protocols, instead endorsing SOAP.

Building a Solid Foundation for Web Services

- Standards-writing bodies should take a couple of years off. We have more than enough overlapping XML standards. Like the existing HTML/HTTP standards we depend on today, the existing XML standards are imperfect but workable. It's better to have every vendor implementing the same imperfect standard than to have the industry in limbo waiting for the upcoming perfected standard.
- Standards-supporting bodies such as the National Institute for Standards and Technology (NIST) and OASIS have a pivotal role to play. These bodies should continue their historical

role and build conformance test-beds on the Internet for the de facto Web services standards discussed above.

- Industry groups such as OAGI, RosettaNet, and HL7 would do well to focus their efforts on defining solid, simple XML payload for their respective industries.
- Vendors should support this basic Web services framework in their products. It's okay if they sneer at this basic framework as “lowest-common-denominator” and up-sell customers on their advanced extensions. If there are enough customers who want these advanced extensions, they will end up self-organizing into higher-performance subnetworks.
- Customers should insist on vendors providing clean support for this “basic” Web services model. Unless your implementation of Web services is interoperable with others, you're not going to have access to the full universe of available services.

Report from the Real World

There is evidence that this pragmatism has occurred to others as well. On January 3, 2002, the B2B auto exchange Covisint made a firm public commitment to ebXML – but only for the TRP (SOAP) and payload (OAGI) portions. Three weeks later, in an independent effort, a commercial consortium including EDS, Sun, and Killdara demonstrated an ebXML-based network for the retail side of this same industry, “Dealersphere.” Six software applications communicated using HTTP, ebXML TRP (extended SOAP), and XML payloads derived from the work of the STAR/OAGI standards effort. The security and directory aspects of ebXML weren't implemented, but will be in the production version (assuming the relevant standards are firmed up in 2002).

The formation of the Web Services Interoperability (WS-I) consortium in February is a recognition of this problem. We can hope that WS-I will be a pragmatic project, keeping a sharp focus on hard-core interoperability specs and testing.

JOGILVIE @ KILLDARA.COM

LAYER	DE FACTO STANDARD
Transport/Routing	EbXML's TRP (extended SOAP)
XML Payload	Depends on the industry. Pick one payload standard for each sector: RosettaNet (electronics), OAGI (manufacturing), HL7 (healthcare), ACORD (insurance), STAR (auto retail)
Privacy	SSL
Authentication	XML-Digital Signatures (PKI)
Directory	UDDI & WSDL (with minor upgrades inspired by ebXML Registry/Repository)

TABLE 1 How to build out basic Web services

Simplex Knowledge

www.skc.com

PART 2 XML

WRITTEN BY Mike Jasnowski

Asynchronous messaging and JAXM JSP tags

Part 1 of this article looked at the JAXM API and some of the components you can use to send messages synchronously. But the JAXM API offers more features, such as the ability to send and receive messages asynchronously. To make the JAXM API easier to use in your applications, there is also a set of custom JSP tags.

Communications in enterprise applications are rarely only synchronous in nature. Some applications send a request and don't expect an immediate response. Applications that send a request and receive a response at a later time are said to be *asynchronous* in nature. This ability to "fire-and-forget" is a common practice in distributed applications.

An asynchronous JAXM client uses a messaging provider to send messages to instead of sending them directly to the recipient. A messaging provider is somewhat similar to a JMS queue. The difference is that, with a JMS queue, messages may not be delivered automatically to the recipient, depending on the type of queue and messaging model used. With a SOAP messaging provider, the messages are forwarded to the recipient automatically. There are provisions for handling retries should a message recipient be unavailable. This enables some rudimentary form of reliable messaging, as messages can be persisted to disk in a directory until they can be delivered. This could take several hours to several days.

When you send messages using a messaging provider, you also create messages and connections differently from when you send messages directly to a recipient. When you want to create a message, you still use the `MessageFactory` class, but you get a reference to the `MessageFactory` class from an object called a `ProviderConnection`.

The ProviderConnection Object

To get a `ProviderConnection` object, you first have to obtain a reference to a `ProviderConnectionFactory`, which is responsible for creating `ProviderConnection` objects that are retrieved using the JNDI (Java Naming and Directory Interface). The `ProviderConnectionFactory` is found at the URL where it has been registered. The following snippet illustrates how you can retrieve an instance of a `ProviderConnection` object.

```
Context ctx = new InitialContext();
ProviderConnectionFactory pcf =
    (ProviderConnectionFactory)ctx.lookup
```

```
("http://www.yourprovider.com/provider");
ProviderConnection pc = pcf.createConnection();
```

The first line of code, which retrieves a `Context` object, is used by applications that will use JNDI operations such as the `lookup()` method in the next line of code. This method is used to obtain a reference to the `ProviderConnectionFactory` object that you'll use to create a connection. The URI you provide to the `lookup()` method must be bound by some other means before you attempt to use it. Usually JNDI names are bound by the objects that need to be located by the JNDI name in question. If you've written code that uses RMI or EJBs, the preceding code may look familiar. Remote objects are looked up and accessed in a similar way.

Once you have an instance of the `ProviderConnection` object, you're ready to create the `MessageFactory` from which you'll create SOAP-Message objects. To create a `MessageFactory` using the `ProviderConnection` object you'll use code that looks like the following:

```
MessageFactory messageFactory = pc.createMessageFactory("XMLJournal")
```

The `createMessageFactory()` method is called to create a `MessageFactory` instance, unlike previous examples where you used the `newInstance()` method. You pass in a single argument representing the name of a profile for a `MessageFactory` that creates a certain type of message.

Once you've created the desired message type, you can go about populating the message. Depending on the message profile retrieved, you set different parts of the message to meet your needs. For instance, if you were creating a basic SOAP message, you might execute code that looks like:

```
SOAPMessage soapMessage = messageFactory.createMessage();
SOAPPart soapPart = soapMessage.getSOAPPart();
SOAPEnvelope soapEnvelope = soapPart.getEnvelope(true);

SOAPBody soapBody = soapEnvelope.getBody();
soapBody.addTextNode("XML Journal");
```

You could also subclass the `SOAPMessage` class and create message classes that are unique to your own message. The JAXM reference implementation, for example, includes an `ebXML` message class. You would of course need to have the implementation of the class on the sending and receiving ends.

MESSAGING JAXM

WITH

A large satellite dish antenna structure, possibly part of a space station or a large ground-based facility, is shown against a cloudy sky. The dish is composed of many green panels and a complex white metal framework. The structure is angled upwards towards the top right of the frame.

AUTHOR BIO

Mike Jasnowski, a senior software engineer with eXcelon Corporation in Burlington, MA, has over 17 years of experience, including more than 4 with Java. Mike is a Sun-certified Java programmer.



Having set the content of the message the way you want it, you can now send it. Sending a message using a messaging provider looks like:

```
pc.send(s soapMessage);  
pc.close();
```

The `send()` method takes an instance of a `SOAPMessage` class. After the `send()` method returns, you should call the `close()` method to close the connection with the provider. Listing 1 shows the source code for a JAXM client application that sends a message asynchronously.

Unlike a message sent directly to a recipient, a message sent to a recipient via a provider involves knowing only the name of the profile. This hides many details of the recipient and can be somewhat confusing if you know nothing about the recipient, but it's also good in that you don't have to know the details. The messaging provider is responsible for delivering the message and knowing about the address of the recipient. The JAXM specification defines a construct known as a message profile.

Message Profiles

Profiles are used with JAXM clients that use a messaging provider, unlike JAXM clients that talk directly to the client using a `SOAPConnection`. Profiles are used to describe information about a particular message type – in the case of the JAXM API, a sample XML-Journal message profile is provided. You could also create your own custom profile to describe other types of messages.

A list of supported profiles can be retrieved from a messaging provider by using the `getSupportedProfiles()` method of the `ProviderMetaData` object. You can retrieve an instance of the `ProviderMetaData` object from the `ProviderConnection` object using the `getMetaData()` method of the `ProviderConnection` object. To do this you'd use code that looks like this:

```
ProviderMetaData metaData = pc.getMetaData();  
String[] supportedProfiles = metaData.getSupportedProfiles();
```

The `getSupportedProfiles()` method returns a `String` array of the profiles it knows about. You can then look through them to find which profiles it supports. Message profiles are created at runtime by loading an XML file containing information about the profile. The JAXM specification defines a DTD that indicates what the profile should look like. A message profile consists of, for example:

- Where the messaging provider should store messages persistently
- How many times the message provider should attempt to forward a message
- What URI a client should use to contact the provider
- What transport protocol you should use to contact the provider
- How many messages are stored persistently in one file

The message profile is contained in an XML document that's loaded at runtime by a messaging provider. The document contains one or more profiles that describe the previously mentioned information as well as some other pieces of information about the profile. A message profile might look the one in Listing 2.

This profile defines the XML-Journal message profile. It can be found at www.xmljournal.com/sender using HTTP. The provider will store messages in a directory called `xmljournal` and place up to 10 messages in each message log file. These messages represent the messages that will be sent to the recipient.

JAXM clients that use a messaging provider must also be configured with a similar profile about the messaging provider. The JAXM specification also defines a DTD for these clients. The client configuration is XML-based as well, of course, and provides the client with the following information:

- An endpoint identifying the client
- A URL that the provider can send pending messages to
- A URL that the client can use to contact the provider

Reading in and utilizing profiles isn't covered in the current reference implementation, but at least one of the examples that ships with the reference implementation uses profiles.

One thing you should note is that clients that use messaging providers should be able to process messages asynchronously. A provider can deliver messages to a client at any time, which is in sharp contrast to the JAXM client you saw earlier, which sent a message and received a response synchronously. The client wouldn't be able to send a message, close the connection, and then later receive a message from a provider on a different connection. You have clients that send messages to other clients that are later delivered by the provider.

The JAXM API defines the `JAXMServlet` class as a convenience class for writing JAXM clients that run in a servlet container. This enables a JAXM client to make use of the facilities of the servlet container. You don't need any additional infrastructure such as a Web server or HTTP message-handling classes.

The API also defines two interfaces; `AsyncListener` and `SyncListener`, that are generally used by JAXM clients written as servlets. Both interfaces and the `JAXMServlet` class reside in the `javax.xml.messaging` package.

Both the `SyncListener` and `AsyncListener` define an `onMessage()` method that takes a `SOAPMessage` object as the single argument. The `SyncListener` implementation of `onMessage()` also returns an instance of a `SOAPMessage` object as the response. Listing 3 shows the source for a Java servlet that can act as an asynchronous SOAP message listener.

Servlets that implement the `AsyncListener` interface operate similarly to message-driven beans in J2EE applications. The EJB container handles forwarding the message to the MDB in the same way that the messaging provider handles forwarding the HTTP request to the JAXM client receiving the message. I wouldn't be surprised to see JAXM become a standard component of J2EE.

Writing lots of Java code to use JAXM isn't a requirement of using JAXM. In fact, if you primarily use JSPs in parts of your application, you can then make use of JAXM without writing a lot of the code you've seen previously.

JAXM JSP Custom Tags

Writing JAXM applications can also be done using a custom JSP tag library that's included with the API. The tag library defines a number of custom tags that encapsulate the functionality you've seen in Part 1 and the first part of this article. These tags enable you to send messages, access the contents of SOAP messages, and receive a message asynchronously. These tags take care of the necessary details of transforming an HTTP POST request into a SOAP message so you can work with it. It also enables you to formulate a SOAP message request or response and send it to a recipient.

If you're unfamiliar with custom tags, they encapsulate program logic into an XML-like element. The custom tag is used on a Java Server Page;

TAG NAME	DESCRIPTION
<jaxm:context>	Initializes the JAXM runtime; sets up variables <code>jaxmConnection</code> , <code>jaxmMessageFactory</code> , and <code>jaxmPageURI</code> you can use on your page.
<jaxm:onMessage>	Use to receive a SOAP message and return a response; response, if required, can be constructed in the body of the tag.
<jaxm:call>	Use to send a SOAP message to another client; content of the message can be defined in the body of the tag.
<jaxm:soapBody>	Use to encapsulate body of message you send with <jaxm:call> or receive with <jaxm:soapBody>.
<jaxm:dump>	Use to dump contents of a SOAP message.

TABLE 1 Custom tags used on JSPs



when the page is run, the tag is executed and performs some action, possibly inserting new content dynamically into the output of the JSP where the tag was defined. Table 1 shows all the tags and a brief description of each.

Before discussing each tag, let's examine how to install the tag library these tags are a part of. What follows assumes the use of the Tomcat server, but any J2EE-compliant application server or servlet container should work just as well.

To use each tag requires that your JSP can find the tag. To this end, you need to specify the tag library using a declaration that looks like this:

```
<%@ taglib uri="http://java.sun.com/jaxm" prefix="jaxm" %>
```

Deploying the JAXM WAR

The JAXM custom tags and library are packaged as a Web Application Archive (WAR) named `jaxmtags.war`. This file can be found in the `\jaxm\samples` directory of the JAXM distribution. To deploy the tag library in Tomcat, copy the `jaxmtags.war` file into the `\tomcat\webapps` directory and then restart Tomcat. Once restarted, the tag library, along with several examples, will be deployed. You can then access a JAXM custom tag example page by visiting this URL with your Web browser:

```
http://localhost:8080/jaxmtags
```

The port listed in the command line above may need to be changed, depending on how you've configured the Tomcat Web server. The example page has a few examples you can experiment with to get an idea of what's possible with the tag library.

Now that you have Tomcat up and running and can access the JAXM example page, we'll look at each tag, starting with the `<jaxm:context>` tag.

A Look at Each Tag

The `<jaxm:context>` tag is used to initialize the JAXM runtime, set up

any message factories, and construct variables that you can use to access the context elsewhere in the page. This tag creates the variables `jaxmConnection` and `jaxmMessageFactory`. It also defines a variable named `jaxmPageURI` that represents the URI of the Web application this JSP is a part of. The `<jaxm:context>` tag is used at the top of the JSP before any other JAXM-related custom tags.

The `<jaxm:onMessage>` tag allows you to write JSPs that receive SOAP. If you recall, the `onMessage()` method is called when an HTTP POST is sent. The SOAP message is in the body of the HTTP message and is extracted and made into an instance of the `javax.xml.soap.SOAPMessage` class and passed to the recipient. To extract the incoming SOAP message, we need to look at the attributes of the `<jaxm:onMessage>` tag.

The tag defines three attributes: `msgId`, which will hold the incoming SOAP message; `resId`, which will hold the response SOAP message; and `useBody`, which is used to indicate whether the response SOAP message will be constructed using a scriptlet or the body of the tag. The following snippet illustrates the use of the `<jaxm:onMessage>` tag.

```
<%@ taglib uri="http://java.sun.com/jaxm" prefix="jaxm" %>
<jaxm:context/>

<jaxm:onMessage msgId="mId">
  <jaxm:soapBody>
    <response> Message Received </response>
  </jaxm:soapBody>
</jaxm:onMessage>
```

I haven't specified the `useBody` attribute in the above example because the default value of this attribute is true, which means you use the body of the tag to define the response.

When you build the response message, you're specifying only the body of the SOAP message, not the entire message, such as the envelope

SYS-CON Media, the world's leading publisher of i-technology magazines for developers, software architects, and e-commerce professionals, brings you the most comprehensive coverage of WebSphere.



WebSphereDevelopersJournal.com



**Introductory
Charter Subscription**
**SUBSCRIBE NOW AND SAVE \$31.00
OFF THE ANNUAL NEWSSTAND RATE**
ONLY \$149 FOR 1 YEAR (12 ISSUES) REGULAR RATE \$180
OFFER SUBJECT TO CHANGE WITHOUT NOTICE

**Do You Have
Access to the
Internet?**

The
World's
Leading
Independent
WebSphere
Developer
Resource

**Then
Subscribe
Online and
Save \$31!**
It's that easy



or headers. If I were to dump the SOAP message created with the body of <response> Message Received </response>, it would look like this:

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<Body>
<response>Message Received</response>
</Body>
</Envelope>
```

The custom tags take care of building the SOAP <Envelope> and <Body> elements for you.

The <jaxm:call> tag is used to send a SOAP message to an endpoint defined by its endpoint attribute. If you recall from Part 1, the call() method was used by JAXM clients to send messages directly to a recipient. The <jaxm:call> tag defines three attributes: msgId, which is used to hold the request SOAP message; resId, which is used to hold the response to the message; and endPoint, which is used to indicate the URL of the endpoint you're sending the SOAP message to.

The code snippet in Listing 4 illustrates the use of the <jaxm:call> tag to send a message to a JSP named jaxmreceiver.jsp. When using this tag, you can retrieve the response SOAP message by accessing the variable created by the resId attribute. In the previous example the variable sres can be used to access the contents of the response programmatically using the JAXM SOAP classes. Listing 5 shows the

source of a JSP that uses the SOAP classes to programmatically access the body of the returned SOAP message.

The section of code that appears within the scriptlet tags <% and %> extracts the SOAPMessage object representing the response by casting the sres variable to a SOAPMessage class. I then proceed to access the various parts of the SOAP message such as the envelope and the body. Once I have the body of the message, I use the getChildElements() method to retrieve an iterator over the child elements in the body of the SOAP message.

The <jaxm:soapBody> tag is used to define the contents of a SOAP message when using the body of the tag instead of a scriptlet to define the content. You use this tag in conjunction with the <jaxm:call> tag to define the contents of the SOAP message you're sending. You also use the tag in conjunction with the <jaxm:onMessage> tag to define the body of the SOAP message you're sending as a response. Following is an example of using the <jaxm:soapBody> tag to define the contents of a response to a SOAP message:

```
<%@ taglib uri="http://java.sun.com/jaxm" prefix="jaxm" %>
<jaxm:context jndiLookup="Sender"/>

<jaxm:onMessage msgId="mId">
  <jaxm:soapBody>
    <response> Message Received </response>
  </jaxm:soapBody>
</jaxm:onMessage>
```

LISTING 1

```
<%@ taglib uri="http://java.sun.com/jaxm" prefix="jaxm" %>
<%@ page import="javax.xml.soap.*" %>
<%@ page import="java.util.*" %>
<jaxm:context/>
<HTML>
<BODY>

<jaxm:call endpoint='<%= jaxmPageURI + "/jaxmcontext.jsp" %>'
resId="sres" msgId="sreq" >
  <jaxm:soapBody>
    <title>XML Journal</title>
  </jaxm:soapBody>
</jaxm:call>
```

RESPONSE DIAGNOSTIC INFO:

```
<%

SOAPMessage soapMessage = (SOAPMessage)sres;
SOAPPart soapPart = soapMessage.getSOAPPart();
SOAPEnvelope soapEnvelope = soapPart.getEnvelope();
SOAPBody soapBody = soapEnvelope.getBody();
Iterator iter = soapBody.getChildElements();
while (iter.hasNext()) {
    SOAPElement elem = (SOAPElement)iter.next();
    out.println("Element Name: " +
elem.getElementName().getLocalName()+"<br/>");
}

%>

<br/>
<b>Here is the Message I sent:</b> <br/>
<jaxm:dump msgId="sreq"/>
<br/>

<b>Here is the response I received:</b> <br/>
<jaxm:dump msgId="sres"/>

</BODY>
</HTML>
```

LISTING 2

```
import javax.xml.soap.*;
import javax.xml.messaging.*;
import java.io.*;
import java.net.*;
import javax.activation.*;
import javax.naming.*;
import java.util.*;
```

```
public class AsyncClient {

    public static void main(String args[])
        throws
        SOAPException, IOException, NamingException{

        Context ctx = new InitialContext();
        System.out.println(ctx.INITIAL_CONTEXT_FACTORY);
        ProviderConnectionFactory pcf =
            (ProviderConnectionFactory)ctx.lookup
            ("http://java.sun.com/xml/jaxm/provider");
        ProviderConnection pc =
            pcf.createConnection();

        /* Create the MessageFactory */
        MessageFactory messageFactory =
            pc.createMessageFactory("XMLJournal");

        /* Create an empty SOAP Message */
        SOAPMessage soapMessage =
            messageFactory.createMessage();
        SOAPPart soapPart =
            soapMessage.getSOAPPart();
        SOAPEnvelope soapEnvelope =
            soapPart.getEnvelope(true);

        SOAPBody soapBody = soapEnvelope.getBody();
        soapBody.addTextNode("XML Journal");

        pc.send(soapMessage);
        pc.close();

    }
}
```

LISTING 3

```
<?xml version="1.0" encoding="ISO-8859-1"?>

<!DOCTYPE ProviderConfig
PUBLIC "-//Sun Microsystems, Inc.//DTD JAXM Provider//EN"
"http://java.sun.com/xml/dtds/jaxm_provider_0_8.dtd">

<ProviderConfig>
  <Profile profileId="XMLJournal">
    <Transport>
      <Protocol>http</Protocol>
      <Endpoint>
        <Map>
          <URI>
            http://www.xmljournal.com/sender
```




The `<jaxm:dump>` tag will display the raw SOAP message, including the envelope, when given a message ID previously defined on another tag with an `msgId` attribute. Because you can easily dump the contents of a SOAP message, this tag is useful for debugging. Listing 6 is an example of the `<jaxm:dump>` tag used to dump a message you've sent and the response you received.

In this example I used the `<jaxm:dump>` tag to dump the contents of the SOAP request message defined in the `sreq` variable as well as the response to the message held in the `sres` variable. The output from using the `<jaxm:dump>` tag to dump a SOAP message could result in output that looks like this:

```
<Envelope xmlns="http://schemas.xmlsoap.org/soap/envelope/" xmlns:
  SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
<Body>
<magazines>
<magazine>XML Journal</magazine>
<magazine> Java Developers Journal </magazine>
<magazine> Web Services Journal </magazine>
</magazines>
</Body>
</Envelope>
```

The output of using the `<jaxm:dump>` tag to dump the contents of the response would look nearly identical to that of the request.

Summary

The JAXM API can be used to send messages asynchronously using a messaging provider that acts like a post office, forwarding the message to the recipient and possibly performing retries if the message can't be delivered. The custom tag library that's distributed with the JAXM API is used to incorporate sending SOAP messages using JSPs. There are tags for sending messages, receiving messages asynchronously, and viewing the raw contents of SOAP messages, just to name a few.

Links

For more information about SOAP, XML, and messaging in general, as well as locations for downloading the APIs discussed in this article:

- **Java API for XML messaging:** <http://java.sun.com/xml/jaxm>
- **Simple Object Access Protocol 1.1 (SOAP w/Attachments):** www.w3.org/TR/SOAP-Attachments
- **Simple Object Access Protocol 1.1 (SOAP):** www.w3.org/TR/SOAP
- **Java Message Service:** <http://java.sun.com/products/jms/index.html>
- **Hypertext Transfer Protocol 1.1:** www.w3.org/Protocols/rfc2616/rfc2616.html
- **Java Server Page Tag Libraries:** http://java.sun.com/j2ee/tutorial/1_3-fcs/doc/JSPTags.html
- **Message-driven beans:** http://java.sun.com/j2ee/tutorial/1_3-fcs/doc/MDB.html

MIKEJ @ EXCELONCORP.COM

```
</URI>
<URL>
  http://127.0.0.1:8080/hmi/receiver
</URL>
</Map>
</Endpoint>
<Persistence>
  <Directory>
    xmljournal/
  </Directory>
  <RecordsPerFile>
    10
  </RecordsPerFile>
</Persistence>
</Transport>
</Profile>
</ProviderConfig>
```

LISTING 4

```
<%@ taglib uri="http://java.sun.com/jaxm" prefix="jaxm" %>
<jaxm:context jndiLookup="Sender"/>
<HTML>
<BODY>

<jaxm:call endpoint='<%= jaxmPageURI + "/jaxmreceiver.jsp"
%>' resId="sres" msgId="sreq" >
  <jaxm:soapBody>
    <magazines>
      <magazine>XML Journal</magazine>
      <magazine> Java Developers Journal </magazine>
      <magazine> Web Services Journal </magazine>
    </magazines>
  </jaxm:soapBody>
</jaxm:call>
```

 Message Sent

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

LISTING 5

```
import javax.xml.messaging.*;
import javax.xml.soap.*;
import javax.servlet.*;
import javax.xml.transform.*;
import javax.xml.transform.*;
import javax.xml.transform.*;
import java.io.*;
```

```
public class MsgReceiveServlet extends JAXMServlet implements
  AsyncListener {

  public void init(ServletConfig servletConfig) throws
    ServletException {
    super.init(servletConfig);
  }

  public void onMessage(SOAPMessage message) {

    try {
      message.saveChanges();
      message.writeTo(System.out);
    } catch (Exception e) {
      System.out.println(e);
    }
  }
}
```

LISTING 6

```
<%@ taglib uri="http://java.sun.com/jaxm" prefix="jaxm" %>
<jaxm:context jndiLookup="Sender"/>
<HTML>
<BODY>

<jaxm:call endpoint='<%= jaxmPageURI + "/jaxmcontext.jsp"
%>' resId="sres" msgId="sreq" >
  <jaxm:soapBody>
    <magazines>
      <magazine>XML Journal</magazine>
      <magazine> Java Developers Journal </magazine>
      <magazine> Web Services Journal </magazine>
    </magazines>
  </jaxm:soapBody>
</jaxm:call>
```

Here is the Message I sent:

<jaxm:dump msgId="sreq"/>

Here is the response I received:

<jaxm:dump msgId="sres"/>

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

WebServices
JOURNAL

XML JOURNAL

THE FIRST & ONLY
WEB SERVICES
RESOURCE CD

WEB SERVICES RESOURCE CD

THE SECRETS OF THE WEB SERVICES MASTERS

INCLUDES EXCLUSIVE .NET ARTICLES

MORE THAN

400
EXCLUSIVE

WEB SERVICES
& XML
ARTICLES



EDITED BY
SEAN RHODY

\$119
CD
VALUE

FROM
WEB SERVICES
JOURNAL

Web services **EDGE** \$100
conference & expo coupon inside

EVERY ISSUE OF WSJ & XML-J EVER PUBLISHED

THE MOST COMPLETE LIBRARY OF EXCLUSIVE WSJ & XML-J ARTICLES ON ONE CD!

"The Secrets of the Web Services Masters"

CD is edited by well-known WSJ Editor-in-Chief
Sean Rhody and organized into more than 40 chapters
containing more than 400 exclusive WSJ & XML-J articles.

Easy-to-navigate HTML format!

Bonus:

Full .PDF versions of every WSJ & XML-J published
since the first issue

XML in Transit	XML Industry	XML &	UML
XML B2B	Insider	Databases	Integration
Java & XML	<e-BizML>	Electronic Data	WSDL
The XML Files	XML & Business	Interchange	Beginning
XML & WML	XML Demystified	Ubiquitous	Web Services
Voice XML	XML &	Computing	Web Services
SYS-CON Radio	E-Commerce	Information	Tips & Techniques
XML & XSLT	XML Middleware	Management	Frameworks
XML & XSL	XML Modeling	Objects & XML	Management
XML & XHTML	CORBA & XML	XML Pros & Cons	Security
2B or Not 2B	Data Transition	Java Servlets	UDDI
XML Script	XML @ Work	XML Filter	.NET

Special Limited Time Price

Now
Shipping

\$79

+ S&H

ONLINE
ORDER AT
JDJSTORE.COM
SAVE
\$40

WWW.JDJSTORE.com

OFFER EXPIRES JUNE 30, 2002

3
YEARS
25
ISSUES
400
ARTICLES
ONE CD





XML – Relevant to both the wireless and the wired worlds

XMLWithoutWires

Part 2 of 2

Last month we focused on the need for compression when XML is transmitted over a wireless network (*XML-J*, Vol. 3, issue 3). We also looked at the use of XSLT to tailor a single XML document for display on multiple wireless devices, each of which might have different display capabilities.

This article continues the discussion of wireless XML with a look at wireless XML initiatives such as VoiceXML and WAP/WML. And we'll see how two of the hottest areas in computing – information security and Web services – apply to wireless XML.

Five years ago HTML's ease-of-use enabled the World Wide Web to grow at a dramatic pace. Anybody could create a Web page and publish it for the whole world to see. The goal of the VoiceXML initiative is to bring this same ease of use to the development of voice applications, which include interactive voice response (IVR) and automated speech recognition (ASR) systems. Up until now, developing for these systems meant learning an arcane programming or scripting language.

The VoiceXML 2.0 specification was issued by the World Wide Web Consortium as a working draft on October 23, 2001 (see www.w3.org/TR/2001/WD-voicexml20-20011023/). Real-world uses of VoiceXML include the delivery of road traffic information over telephones, and automated helpdesk systems. The developer doesn't need to worry about the underlying speech-synthesis and speech-recording technology. The VoiceXML engine takes care of that in the same way that a Web browser takes care of the rendering of HTML.

VoiceXML describes how to make audio "forms," which can be thought of as analogous to visual HTML-based forms. These audio forms play audio questions and record the user's voice responses. The interaction between the user and the VoiceXML-enabled application can proceed sequentially by following a predefined order of questions. However, VoiceXML provides a rich framework for flow control and scripting that enables "intelli-

gent" applications to be developed. For an example, see Listing 1.

The "field" tag indicates that the user must speak an input value (in this case, a type of coffee) before control passes to the next element in the form.

The following is a sample interaction using this form:

C:(computer): Would you like a latte, mocha, or espresso?

H:(human): A beer, please.

C:I'm sorry; I did not understand what you said. [This is a platform-specific message used when the user's voice input doesn't match an entry in the coffee.grxml grammar.]

C:Would you like a latte, mocha, or espresso?

H:Latte.

C:[Control passes to the script "deliver-coffee.pl"]

As you can see, the items "latte," "mocha," and "espresso" are contained in a separate grammar file, with a ".grxml" file ending. This file contains words that the user is permitted to speak as a response to the question. An example of a grammar is shown below:

```
<grammar mode="voice">
  <rule id="root" scope="public">
    <one-of>
      <item> latte </item>
      <item> mocha </item>
      <item> espresso </item>
    </one-of>
  </rule>
</grammar>
```

The grammar shown above has a mode of type "voice." This contrasts with grammars of type "dtmf" (DTMF stands for Dual-Tone Multi-Frequency). DTMF tones are the noises that phone buttons

make when pressed. Somewhat interestingly, each DTMF tone is the sum of two tones played together (hence "dual-tone"). By using a DTMF grammar, our coffee-selection example application can be adapted to take key-presses in place of spoken input. An example of such a DTMF grammar, in which the user presses key 1, 2, or 3 on a phone to choose a coffee type, is shown in the next example:

```
<grammar mode="dtmf">
  <rule id="root" scope="public">
    <one-of>
      <item> 1 </item>
      <item> 2 </item>
      <item> 3 </item>
    </one-of>
  </rule>
</grammar>
```

Wireless Markup Language was one of the first commercial applications of XML. WML was developed by the WAP Forum, a consortium of providers of wireless hardware and software products. WAP began with HDML (Handheld Device Markup Language) as its basis, and aimed to use the model of the World Wide Web and its protocols but in a more efficient language, more suited to the limited size, memory, and processing power of the wireless world.

The first Wireless Access Protocol specification drew flak because of a perception that the WAP Forum was needlessly re-creating technologies that were already present in the nonwireless world. Why bother using WML and WTLS (Wireless Transport Layer Security), the argument went, when XHTML and TLS are available already, and are proven in the field.

When the successful Japanese i-mode system bypassed WML and instead used cHTML, a subset of HTML, the writing seemed to be on the wall for WML. However, the next generation of WAP – WAP 2.0 – supports the W3C

AUTHOR BIO

Mark O'Neill, CTO of Vordel, oversees the development of Vordel's technical strategy and product development in the areas of XML and public key cryptography. Mark designed and implemented Internet EDI solutions for Ireland's largest EDI value-added network, Eirtrade Services Ltd. He holds a double-honors degree in mathematics and psychology from Trinity College Dublin and studied neural network modeling at Oxford.

XHTML and CSS recommendations, as well as TLS, thus reducing the need for Web developers to learn new languages and technologies. Other new features in the 2.0 release include synchronization features – implemented using SyncML – to access calendaring and addressing information from remote devices.

With the release of WAP 2.0, WML continues to be relevant. So take a look at Listing 2, an example WML page, reusing our coffee choice example from before.

Reading this code listing from top to bottom, we can see that it begins with the familiar XML directive, which identifies the document as XML. The next line indicates that the document must conform to the WML DTD. After the `<wml>` tag, we see the first “card”; WML uses the analogy of “cards” to describe each single screen of text to be shown on a cell phone display, and a group of these cards is sometimes called a “deck.” The `<do>` and `<go>` tags create a hyperlink in the form of a button, which a user can scroll to and choose, using the navigation bar on his or her phone. The “id” attributes of the card elements are used to target the hyperlinks, so that users who choose a mocha coffee, for example, are sent to the card called “MochaCard”.

How this WML application is displayed depends on the cell phone or PDA that’s used for viewing. For instance, in Nokia 6150, the “accept” attribute on a “do” element gives users an Options “button” so they can click Next.

Security is a hot topic at the moment, not just for wireless systems but for computing in general. In the area of XML, initiatives such as XML Signature and XML Encryption are vitally important. XML Signature provides for digital signatures that are XML-aware, enabling portions of an XML document to be signed, and enabling information about the digital signature itself to be expressed in XML format.

XML Encryption is at a much earlier stage of development than XML Signature, though its philosophy is somewhat similar. XML Encryption provides for XML-aware encryption, meaning that only a portion of an XML document may be encrypted and the information about the encryption is then expressed in XML format. XML Encryption and XML Signature bring the advantages of the XML world to the arcane and complicated world of information security, and also bring much-needed security capabilities to the XML world.

When XML is sent wirelessly, XML Signature and XML Encryption are just as important as when the medium is fixed-wire. Related initiatives such as SAML

(Security Assertions Markup Language) and Microsoft’s .NET MyServices (formerly code-named HailStorm) will also need to be addressed by wireless devices. Public key cryptography is typically used to generate digital signatures and to enable the exchange of encryption keys.

The problem for wireless devices is the lack of battery power, bandwidth, and processor speed necessary to implement public key cryptography. One solution to this problem is to use the relatively new Elliptical Curve Cryptography technology. ECC uses less processing power than other types of public key cryptography – for example, the famous RSA public and private key algorithms, which are based on prime numbers arithmetic.

Web services are a very hot topic right now, and provide a framework that promises to make XML ubiquitous as the means of connecting software services across networks. Web services make particular sense for wireless devices because they allow the device to act as a Requestor to Web services located on remote machines. In this scenario it isn’t necessary for the wireless device to run the service itself; it just needs the ability to create an XML-based request, wrap it as a SOAP message, and invoke the remote Web service. To make the outgoing SOAP message, the wireless device may use a full-blown SOAP stack, complete with a Document Object Model. However, this may be out of the question for many small-footprint devices, due to the size of DOM implementations. The alternative is to use a pared-down DOM, using only the features needed for SOAP, or to simply not use a DOM at all, and to craft the outgoing XML using string-manipulation code.

Microsoft’s .NET framework for Web services includes a technology called Smart Devices Extensions (SDE), which allows .NET applications to run on mobile devices. By *.NET applications* I mean applications that use Microsoft’s Common Language Runtime (CLR), which provides many useful functions such as TCP/IP networking and string processing.

The most common example – for now at least – of a .NET-enabled mobile device is a Pocket PC. Examples of Pocket PC devices include the popular Compaq iPaq PDA. In order to create a Web service request from a Pocket PC, a SOAP implementation is needed (see for example the Pocket SOAP toolkit from Simon Fell at www.pocketsoap.com). In the Palm world the analogous way to enable a Palm device to consume Web services is to install a Java 2 Platform, Micro Edition (J2ME) virtual machine and a SOAP implementation (e.g., the SOAP package from Enhydra (kSOAPenhydra.org).

• • •

In summary, we’ve rounded out our look at wireless XML by examining VoiceXML and WML, security, and how the hot new area of Web services applies to mobile devices. The defining theme of these two articles on wireless XML has been the limitations of the platform in terms of screen size, memory, bandwidth, and processing power. Although wireless devices will improve in performance, and wireless bandwidth will become more plentiful, the wireless world will always lag the fixed-wire world in terms of capability. Wireless XML technologies address this fact, and mean that the great benefits of XML are as relevant to the wireless world as to the wired world. ☎

MARK ONEILL @ VORDEL.COM

LISTING 1 A simple VoiceXML interaction

```
<?xml version="1.0"?>
<vxml version="2.0">
  <form>
    <field name="Coffee">
      <prompt>Would you like a latte, mocha,
        or espresso?</prompt>
      <grammar src="coffee.grxml" type=
        "application/grammar+xml"/>
    </field>
  </form>
  <submit next="http://www.coffee-example.com/
    deliver-coffee.pl"/>
</vxml>
```

LISTING 2 WML example

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML
  2.0//EN" "http://www.wapforum.org/DTD/wml20.dtd">
<wml>
  <card id="CoffeeChoiceCard" title=
    "Choose Coffee">
    <p>
      Please choose a type of coffee...
    </p>
    <do type="accept" label="Latte">
      <go href="#LatteCard"/>
    </do>
    <do type="accept" label="Mocha">
      <go href="#MochaCard"/>
    </do>
    <do type="accept" label="Espresso">
      <go href="#EspressoCard"/>
    </do>
  </card>
  <card id="LatteCard" title="Latte">
    <p>
      Enjoy your latte!
    </p>
  </card>
  <card id="MochaCard" title="Mocha">
    <p>
      Enjoy your mocha!
    </p>
  </card>
  <card id="EspressoCard" title="Espresso">
    <p>
      Enjoy your espresso!
    </p>
  </card>
</wml>
```

DOWNLOAD THE CODE @
www.sys-con.com/xml



How to transform XML to PDF using FOP

GotXSLT?

Part 5 of 5

As a substitute for handcrafting, most word processing and desktop publishing tools lack the capability to produce the detailed design and sophistication that normally accompany high-end page layouts. And companies, of course, are always looking for the most cost-effective way to deliver documents of consistently top-quality appearance.

What's needed is a powerful software that automates the process entirely and possesses capabilities far beyond those found in today's tools. Unfortunately, a document that's transformed using XSLT may be presentable, but it's not really ready for the type of report or presentation we're aiming for here. To be specific, printed reports often require properties or instructions that aren't included with the XML document and aren't provided by XSLT. At a minimum, an advanced formatting engine or processor must provide:

- Support for header/footer specifications and page number generation/ placement in the page layout
- Support for page layout descriptions, including different left, right, and blank page layouts
- Support for multiple columns on a page
- Flexibility to specify min/max/optimum values for page margins
- Fonts, highlighting (foreground/background colors, underscoring, all capitals)
- Support for justification and page breaks
- Support for tables, lists, and external graphics linking, including autoscaling

Introducing XSL-FO

XSL Formatting Objects is an XML-based markup language that specifies formatting semantics for high-quality documents. XSL-FO is basically a style specification designed to provide detailed layout-control devices and format-independent requirements.

The World Wide Web Consortium's specification for XSL comes in two parts (see Figure 1):

- **XSLT**: A language for transforming XML documents to other formats such as XML, HTML, and WML

- **XSL-FO**: A language for specifying format semantics

Thus XSL = XSLT + a vocabulary of FOs and properties.

It's only fair to warn you that XSL-FO is more complex and bigger than XSLT, and because of that, it isn't widely used or supported by your browser or other software tools at this time. In fact, as far as I know, no implementation of XSL-FO is near the complete standard.

FOP (the Apache FO-to-PDF project) is one such implementation of the XSL-FO specification. FOP is the world's first print- and output-independent formatter (as claimed by the Apache XML project) that's driven by XSL-FO and compliant to at least the basic conformance level described in the W3C Candidate Recommendation of November 21, 2000, and that complies with the March 11, 1999, Portable Document Format Specification (version 1.3) from Adobe Systems.

FOP is a Java application or library that converts an XSL-FO document into paginated output. FOP supports several output formats such as printer formats (HP PCL, PostScript), PDF, print (hard copy output printed directly), and SVG (I'll explain that later). The main target, however, is PDF. FOP can be run from the command line or embedded in your Java application.

As we've seen in previous articles, the most common use of XSLT is to transform XML into HTML; the most common use of XSL-FO is to convert XML into PDF. Given that XSL-FO is part of XSL, the normal way to produce XSL-FO documents is by transforming XML documents using XSLT. In fact, XSL-FO documents are almost always created using XSLT! Figure 2 depicts the XSL-FO-to-PDF transformation process.

Our Goal

Our goal is to produce a high-quality presentation (PDF) of our original XML document. This is a two-step process in which we'll use the XSLT engine to transform XML to XSL-FO and then use XSL Formatter (FOP) to transform XSL-FO to the PDF format. Figure 3 shows the resultant PDF output.

Understanding Our XSL-FO Document

XSL-FO documents typically end with .fob, .fo, or .xml. Since the XSL-FO specification is quite large and complex, we'll look only at the most frequently used elements. The following is a list of the most common formatting objects and their use:

- **Page-sequence**: A major part (such as front or body) in which the basic page layout may differ from other parts
- **Flow**: A chapter- or section-like division within a page-sequence
- **Block**: A paragraph (or title or block quote, for instance)
- **Inline**: A font change within a paragraph, for example
- **List FOs**: List-block, list-item, list-item-label, list-item-body
- **Graphic**: References an external graphic object
- **Table FOs**: Mostly analogous to the standard HTML table models

Step 1: FO file skeleton

Like all XML documents, an XSL-FO document requires a namespace and root node, as follows:

```
<?xml version="1.0"?>
<fo:root xmlns:fo=
"http://www.w3.org/1999/XSL/Format">
...
</fo:root>
```

XSL-FO borrows its page formatting from DSSSL, the Document Style Semantics and Specification Language that

AUTHOR BIO

Shouki Sourì, a lead software engineer at PanAmSat Corporation, is experienced in Java, CORBA, XML/XSL, C/C++, and other technologies. Shouki holds a bachelor's degree in electrical engineering and a master's in computer science.

is meant to work with SGML, the Standard Generalized Markup Language. DSSSL is divided into many different parts: style language, flow objects, transformation language, the document model, and query language. There are several implementations of DSSSL. Jade was the first DSSSL processor released. It is a fast C++-based processor that is available for Win32 and Unix platforms.

For more information about DSSSL see the Resources section.

Step 2: Setting up page templates

It's important to mention DSSSL here because it uses the desktop publishing concept of "master pages," which are basically templates for how different types of pages should be laid out. Odd-numbered pages, for example, have a "gutter" (an area set aside for the stitching that holds a book together) on the left, while even-numbered pages have a gutter on the right. (The odd-numbered pages do have a similar format though, as do the even-numbered pages.)

At the start of an XSL-FO document we describe the master pages used in the document. The description must occur within a layout-master-set tag. In the example here I'll just use a single type of master page called "first." The formatting object <fo:simple-page-master> helps us do just that, that is, have a 1.5-cm margin on the right and left, 1 cm on the top, and 2 cm on the bottom. Margins can also be specified in inches, if you prefer – as in margin-right="1 in" – or in points – as in margin-right="75pt". We also specified page size, as in page-width and page-height, in centimeters in our example. And, finally, we use the formatting object <fo:region> to define

the space of each page region. A page has five regions: region-body, region-before, region-after, region-start, and region-end.

```
<fo:layout-master-set>
  <fo:simple-page-master
    margin-right="1.5cm"
    margin-left="1.5cm"
    margin-bottom="2cm"
    margin-top="1cm"
    page-width="21cm"
    page-height="29.7cm"
    master-name="first">
    <fo:region-before extent="1cm"/>
    <fo:region-body margin-top="1cm"/>
    <fo:region-after extent="1.5cm"/>
  </fo:simple-page-master>
</fo:layout-master-set>
```

...

Figure 4 shows all the regions in a page.

The main region, region-body, defines the space for the body of the document that our formatting objects will end up in.

Step 3: Laying out the content

Once the master pages are defined, you can put text on these pages. To lay out text, you first have to define which master page should be used by defining a page-sequence. A page-sequence defines the content of the output document. Since we're only going to use one page layout, there's no need for the sequence-specification element that normally accompanies the page-sequence element.

Next we define the <fo:static-content> element in the region-after area, whose content remains the same for all pages except for the page number. Thus you should specify <fo:static-content> or <fo:static-text> to hold data that's repeated on many pages and is retrieved in a static way (the content itself isn't necessarily static, as the page numbers change from page to page, but the method of retrieving the content is static), such as page headers and chapter titles rather than a variable <fo:flow> to fill the region.

SUBSCRIBE AND SAVE

XML JOURNAL

Offer subject to change without notice

ANNUAL NEWSSTAND RATE	
\$83.88	
YOU PAY	
\$77.99	
YOU SAVE	
\$5.89	Off the Newsstand Rate

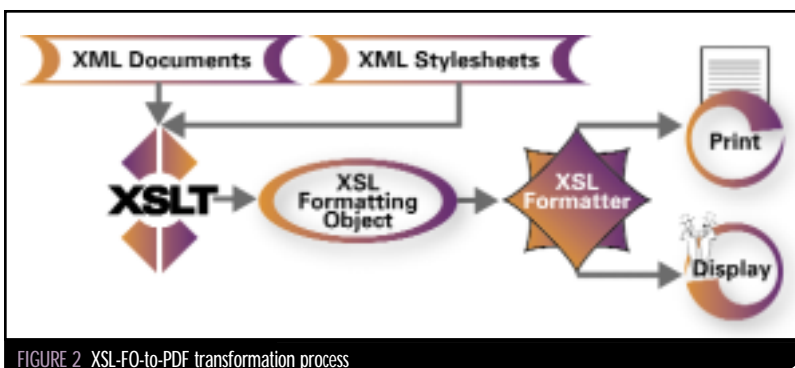
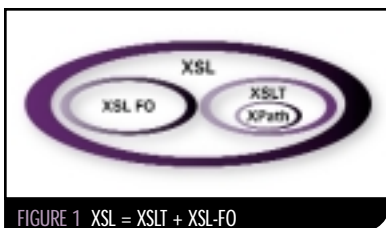
DON'T MISS AN ISSUE!

Receive 12 issues of **XML-Journal** for only **\$77.99!** That's a savings of **\$5.89** off the annual newsstand rate.

Sign up online at www.sys-con.com or call 1 800 513-7111 and subscribe today!

Here's what you'll find in every issue of XML-Journal:

- Exclusive feature articles
- Interviews with the hottest names in XML
- Latest XML product reviews
- Industry watch



SAVE 30% off the annual newsstand rate

JAVA DEVELOPER'S JOURNAL

Offer subject to change without notice

ANNUAL NEWSSTAND RATE

~~\$71.88~~

YOU PAY

\$49.99

YOU SAVE

30% Off the Newsstand Rate

DON'T MISS AN ISSUE!

Receive 12 issues of *Java Developer's Journal* for only \$49.99! That's a savings of \$21.89 off the cover price. Sign up online at www.sys-con.com or call 1 800 513-7111 and subscribe today!

In April *JDJ*:

Pervasive Computing: The Next Generation of Consumer Applications

Now that broadband connections are becoming commonplace and wireless technology is becoming a reality, developers should begin positioning their applications to leverage the new capabilities.

J2EE Application, Module and Component Packaging

The different types of J2EE modules and how they fit into the J2EE architecture.

JDiff – What Really Changed?

JDiff is an open source Java tool, based on Javadoc and developed by the author, that produces HTML documentation describing the precise API changes between two versions of a product.

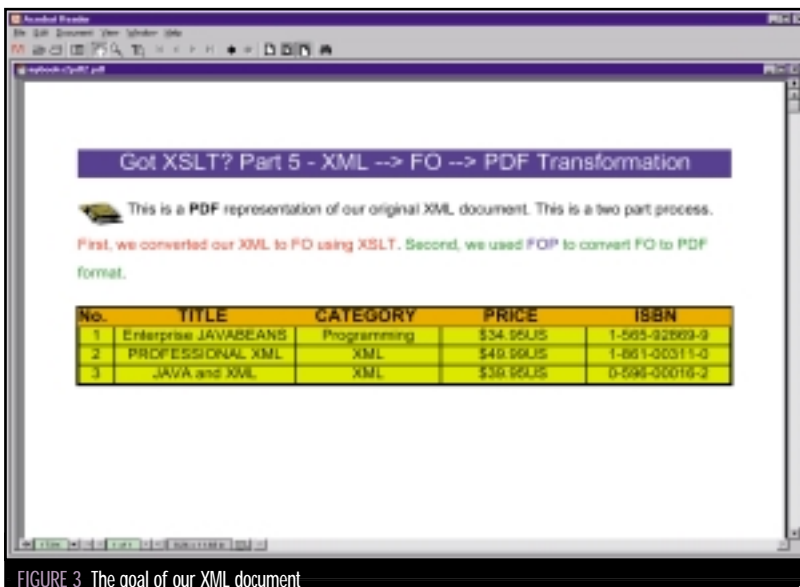


FIGURE 3 The goal of our XML document

We now define a flow element that indicates that the entire formatting object inside, such as text, should flow together as a unit. This text is then divided into individual paragraphs or *blocks*, in XSL-FO terminology. Each block results in a new paragraph. Blocks can be further subdivided into inline regions for individual words, sentences, or other arbitrary sections within the block. The text in flows, blocks, and inlines can have text properties attached to it, such as font size or font type. By default, these text properties are inherited by all text within those regions, but they can be overridden by giving new properties to subregions.

```
<fo:page-sequence master-name="first">
  <fo:static-content flow-name="xsl-region-after">
    <fo:block line-height="14pt"
      font-size="10pt" text-align="center">
      XML to PDF Transformation - p.
    </fo:block>
  </fo:static-content>
  <fo:flow flow-name="xsl-region-body">
    <fo:block text-align="center"
      color="white" background-color="blue"
      space-after.optimum="15pt"
      line-height="24pt" font-family="sans-serif"
      font-size="18pt">
      >Got XSLT? Part 5 - XML -> FO ->
      PDF Transformation</fo:block>
    <fo:block space-after.optimum="15pt"
      line-height="24pt" font-size="12pt">
      <fo:external-graphic height="17px"
        width="35px" src="file:BOOKCATALOG.jpg"/>
    </fo:block>
  </fo:flow>
</fo:page-sequence>
```

This is an **PDF** representation

of our original XML document. This is a two-part process.

```
<fo:inline color="red">
```

First, we convert our XML to FO using XSLT</fo:inline>.

```
<fo:inline color="green">
```

Second, we use

```
<fo:basic-link color="blue" text-decoration="underline"
  external-destination="url('http://www.xml.apache.org/fop/
  index.html/')">FOP</fo:basic-link>
```

to convert FO to PDF format</fo:inline>.

```
</fo:block>
```

```
...
```

```
</fo:flow>
```

```
</fo:page-sequence>
```

Note that we embed a small graphic in the document by using the formatting object <fo:external-graphic>. The source of this JPEG file could be located in a local file, as in this example, <fo:external-graphic src="file:BOOKCATALOG.jpg"/>, or, better yet, we could have XSLT retrieve a graphic in our XML document, the location of which is stored as an attribute named "bookphoto," as in the following:

```
<xslt:template match="book">
  <fo:block>
    <fo:external-graphic src="{@bookphoto}"/>
  </fo:block>
</xslt:template>
```

In addition, we use an external link `<fo:basic-link>` element with a source defined as a URL in our document to define a link to the FOP Web site. There are two types of links in XSL-FO, external and internal. Internal links are for links to locations within the document. Links in XSL-FO are objects that act like the HTML links we're all familiar with.

Step 4: Setting up the table

The fundamental table element in XSL is `<fo:table>`. The XSL-FO table model is quite close to HTML's table model. The `<fo:table>` corresponds to the HTML `<table>` tag, and the `<fo:table-body>` corresponds to `<tbody>`. Keep in mind, however, that FOP 0.20.1 has only limited table support. It doesn't support `<fo:table-caption>` and `<fo:table-and-caption>` elements, for example. In addition, FOP requires you to explicitly specify the column width using a `<fo:table-column>` element, since FOP doesn't automatically figure out how wide the table is.

Our table has been designed to have a first column of 1cm (for No.) and a second column of 5cm (for the title), and so forth.

```
<fo:table border-collapse="separate"
border-style="solid" border-width=
"0.5mm">
  <fo:table-column column-width=
    "1cm"/>
  <fo:table-column column-width=
    "5cm"/>
  <fo:table-column column-width=
    "4cm"/>
  <fo:table-column column-width=
    "4cm"/>
  <fo:table-column column-width=
    "4cm"/>
  <fo:table-body font-family=
    "sans-serif" line-height="16pt"
    font-size="12pt">
    <fo:table-row border-width=
      "0.5pt" border-style="solid"
      font-size="14pt" background-color=
      "Orange">
      <fo:table-cell font-weight=
        "bold">
        <fo:block text-
          align="justify">
          No.
        </fo:block>
      </fo:table-cell>
      <fo:table-cell font-weight=
        "bold">
        <fo:block text-
          align="center">
          TITLE
        </fo:block>
      </fo:table-cell>
      <fo:table-cell font-weight=
        "bold">
```

```
<fo:block text-
  align="center">
  CATEGORY
</fo:block>
```

Note that we defined each block under the `<fo:table-cell>` element with the `text-align` property. The "justify" value specifies that the content is to be expanded to fill the available width in the inline-progression-direction.

Next we define our data rows. The following code depicts five cells in this data row.

```
<fo:table-row font-size="12pt" back
ground-color="yellow">
  <fo:table-cell border-width=
    "0.3pt" border-style="solid">
    <fo:block text-align=
      "center">1</fo:block>
  </fo:table-cell>
  <fo:table-cell border-width=
    "0.3pt" border-style="solid">
    <fo:block text-align=
      "center">Enterprise
      JAVABEANS</fo:block>
  </fo:table-cell>
  <fo:table-cell border-width=
    "0.3pt" border-style="solid">
    <fo:block text-align=
      "center">Programming
    </fo:block>
  </fo:table-cell>
  <fo:table-cell border-width=
    "0.3pt" border-style="solid">
    <fo:block text-align=
      "center">$34.95US<
      /fo:block>
  </fo:table-cell>
  <fo:table-cell border-width=
    "0.3pt" border-style="solid">
    <fo:block text-align=
      "center">1-565-92869-9<
      /fo:block>
  </fo:table-cell>
</fo:table-row>
...
```

Transformations and PDF Files

There are two common ways to generate a PDF when using FOP: a two-stage process and a one-stage process.

In the former we use the Xalan XSLT engine or processor to transform our XML document to XSL-FO using the XSL stylesheet.

I have saved both my `mybooks.xml` (see Listing 1) and `mybook2fo.xsl` (see Listing 2) in the `gotxslt` directory.

To perform the translation type the following:

```
D:\>cd gotxslt
D:\gotxslt>java
```

SAVE 30% off the annual newsstand rate

BUSINESS & TECHNOLOGY
wireless

Offer subject to change without notice

ANNUAL NEWSSTAND RATE	
\$71.88	
YOU PAY	
\$49.99	
YOU SAVE	
30%	Off the Newsstand Rate

DON'T MISS AN ISSUE!

Receive 12 issues of **Wireless Business & Technology** for only \$49.99! That's a savings of 30% off the cover price. Sign up online at www.sys-con.com or call 1 800 513-7111 and subscribe today!

In April **WBT**:

More Than An Overnight Success

Federal Express developed the first wireless package-tracking system in the industry over 20 years ago.

It's a Small World

And getting smaller at Disney World, where wireless LANs are employed throughout the giant park to aid employees and enhance visitors' experiences.

Will Sprint's Strategy Succeed?

The number 4 wireless carrier seems to be rolling the dice with heavy bets on Java.

Wireless Java Is Coming

With the arrival of wireless Java, users are finally able to add exciting new applications to their devices.



SUBSCRIBE AND SAVE

WebServices JOURNAL

Offer subject to change without notice

ANNUAL NEWSSTAND RATE	
\$83.88	
YOU PAY	
\$69.99	
YOU SAVE	
\$13.89	Off the Newsstand Rate

DON'T MISS AN ISSUE!

Receive 12 issues of **Web Services Journal** for only **\$69.99!** That's a savings of **\$13.89** off the annual newsstand rate.

Sign up online at www.sys-con.com or call 1 800 513-7111 and subscribe today!

In April **WSJ**:

Using Web Services with J2EE

Moving into a technology-independent world

Asynchronous Web Services

Deployment based on JMS that extends SOAP-over-HTTP

Dynamically Converting Existing Java Code to a Web Service

A robust and comprehensive framework for delivering Web services

Web Services Over P2P Networks

The technology is ready for an interesting intersection

Knowing the Score – Web Services and Business Processes

The promise of a new era in e-commerce



```
org.apache.xalan.xslt.Process -in
mybooks.xml -xsl mybooks2fo.xsl -out
mybooks2fo.fo
```

If all goes well, the new output file `mybooks2fo.fo` will be the output from the XML-to-XSL-FO translation. The output file `mybooks2fo.fo` is a pure ASCII file, so you can open it and study it in any text editor.

FOP is now used to transform the resultant `.fo` file into a PDF (see Listing 3):

```
D:\gotxslt> fop -fo mybooks2fo.fo
-pdf mybooks2pdf.pdf
```

In the one-stage process FOP actually uses the Apache project Xalan directly to perform the transformation to a PDF.

```
D:\gotxslt> fop mybooks.xml
mybooks2fo.xsl mybooks2pdf.pdf
```

The SVG and XSL-FO Connection

The XSL-FO specification isn't very clear on how to handle different media. For example, do we create one XSL stylesheet for screen display and another for printing, or do we create one generic stylesheet for both?

This is where Scalable Vector Graphics (SVG) can help. In my previous article in this series (Vol. 3, issue 2), I showed how SVG technology brings to the Web screen the rich, high-resolution graphics we've all come to expect in printed catalogs and magazines. Thus SVG is (at least at this time) more suitable for screen rendering and PDF is more suitable for printing.

FOP has limited support for converting SVG into PDF. This can be compensated for by using `<fo:instream-foreign-object>` or, in a separate file, referenced with `<fo:external-graphic>`. The FO stylesheet determines whether the type is SVG from the file extension of the given file property. Otherwise it assumes it is an image.

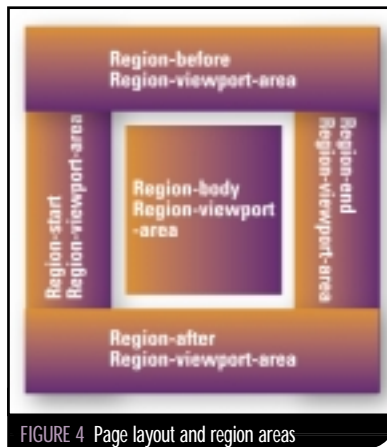


FIGURE 4 Page layout and region areas

The formatting object `<fo:instream-foreign-object>` is used to embed SVG, as in this example:

```
<fo:block>
  <fo:instream-foreign-object>
    <svg:svg xmlns:svg=
      "http://www.w3.org/2000/svg"
      width = "542px" height=
      "505px">
      <svg:title>Embedding SVG
    </svg:title>
    . . .
  </svg:svg>
</fo:instream-foreign-object>
</fo:block>
```

Notice that the SVG is defined using the namespace `svg` to distinguish it from `fo`.

It's possible to convert a stand-alone SVG document directly into a simple PDF document through the use of Batik's (SVG library) transcoder mechanism.

You can, of course, use XSLT to produce SVG documents, and to produce XSL-FO. And, with the correct use of namespaces and the development of suitable tools, you can convert an XML document to a combination of XSL-FO and SVG that a future multicapable XML browser will be able to display in all kinds of impressive ways.

Summary

This article has shown how to use XML and XSLT to create an XSL-FO file, and then, by using FOP, to convert our XSL-FO into a PDF. We also showed that, in principle, if you're using XML-based data, it could be transformed and styled using XSLT and XSL-FO for output on the Web (screen) and on paper (printing).

Given the complexity of the XSLT-FO specification, however, I don't expect much implementation of it in the area of multiformats other than PDF. And certainly not in any Web-enabled mobile phone anytime soon (but I could be wrong...).

Resources

- *The Extensible Stylesheet Language (XSL)*: www.w3.org/Style/XSL/
- *XML Bible, Second Edition*, Chapter 18, "XSL Formatting Objects": www.ibiblio.org/xml/books/bible2/chapters/ch18.html
- *XSL FO reference*: www.zvon.org/xxl/xslfoReference/Output/
- *Introduction to DSSSL*: www.prescod.net/dsssl/

SSOURI @ PANAMSAT.COM

LISTING 1 Our XML document

```
<?xml version="1.0" ?>
<BOOKS>
  <BOOK>
    <ISBN>1-565-92869-9</ISBN>
    <CATEGORY>Programming</CATEGORY>
    <RELEASE_DATE>2000-03-21</RELEASE_DATE>
    <TITLE>Enterprise JAVABEANS</TITLE>
    <PRICE currency="US">34.95</PRICE>
  </BOOK>
  <BOOK>
    <ISBN>1-861-00311-0</ISBN>
    <CATEGORY>XML</CATEGORY>
    <RELEASE_DATE>2000-03-10</RELEASE_DATE>
    <TITLE>PROFESSIONAL XML</TITLE>
    <PRICE currency="US">49.99</PRICE>
  </BOOK>
  <BOOK>
    <ISBN>0-596-00016-2</ISBN>
    <CATEGORY>XML</CATEGORY>
    <RELEASE_DATE>2000-07-11</RELEASE_DATE>
    <TITLE>JAVA and XML</TITLE>
    <PRICE currency="US">39.95</PRICE>
  </BOOK>
</BOOKS>
```

LISTING 2 Complete XSL file to transform XML to XSL-FO – "mybooks2fo.xsl"

```
<?xml version="1.0"?>
<!-- XSLT stylesheet to convert the XML document to XSL-FO
document that in turn is used by FOP -->
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0" xmlns:fo="http://www.w3.org/1999/XSL/Format">
<xsl:output method="xml" indent="yes"/>

<xsl:template match="//BOOKS">
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
<!-- defines page layout -->
<fo:layout-master-set>
  <fo:simple-page-master master-name="first" page-height="29.7cm"
    page-width="21cm" margin-top="1cm" margin-bottom="2cm"
    margin-left="1.5cm" margin-right="1.5cm">
    <fo:region-body margin-top="1cm" />
    <fo:region-before extent="1cm" />
    <fo:region-after extent="1.5cm" />
  </fo:simple-page-master>
</fo:layout-master-set>

<fo:page-sequence master-name="first">
  <fo:static-content flow-name="xsl-region-after">
    <fo:block text-align="end" font-size="10pt" line-height="
      14pt">
      XML to PDF Transformation - p. <fo:page-number />
    </fo:block>
  </fo:static-content>

  <fo:flow flow-name="xsl-region-body">
    <fo:block font-size="18pt" font-family="sans-serif"
      line-height="24pt" space-after.optimum="15pt"
      background-color="blue" color="white" text-align="center">
      >Got XSLT? Part 5 - XML --> FO --> PDF Transformation
    </fo:block>

    <fo:block font-size="12pt"
      line-height="24pt"
      space-after.optimum="15pt"><fo:external-graphic src=
        "file:BOOKCATALOG.jpg" width="35px" height="17px"/>
      This is a <fo:inline font-weight="bold">PDF</fo:inline>
      representation of our original XML document.
      This is a two part process. <fo:inline color="red">First,
      we converted our XML to FO using XSLT</fo:inline>.
      <fo:inline color="green"> Second, we used <fo:basic-link
        external-destination=
          "url('http://www.xml.apache.org/fop/index.html/')"
          text-decoration="underline" color="blue">FOP
        </fo:basic-link> to convert FO to PDF format
      </fo:inline>.
    </fo:block>

    <!-- generates table of contents and puts it into a table -->

    <fo:table border-width="0.5mm" border-style="solid" border-
      collapse="separate">
      <fo:table-column column-width="1cm"/>
```

```
<fo:table-column column-width="5cm" />
<fo:table-column column-width="4cm" />
<fo:table-column column-width="4cm" />
<fo:table-column column-width="4cm" />
<fo:table-body font-size="12pt" line-height="16pt" font-
  family="sans-serif">
  <!-- Table Headers -->
```

```
<fo:table-row background-color="Orange" font-size="14pt"
  border-style="solid" border-width="0.5pt">
  <fo:table-cell font-weight="bold">
    <fo:block text-align="justify">
      No.
    </fo:block>
  </fo:table-cell>
  <fo:table-cell font-weight="bold">
    <fo:block text-align="center">
      TITLE
    </fo:block>
  </fo:table-cell>
  <fo:table-cell font-weight="bold">
    <fo:block text-align="center">
      CATEGORY
    </fo:block>
  </fo:table-cell>
  <fo:table-cell font-weight="bold">
    <fo:block text-align="center">
      PRICE
    </fo:block>
  </fo:table-cell>
  <fo:table-cell font-weight="bold">
    <fo:block text-align="center">
      ISBN
    </fo:block>
  </fo:table-cell>
</fo:table-row>
```

```
<!-- Table data -->
<xsl:for-each select="BOOK">
  <fo:table-row background-color="yellow" font-size="12pt">

    <fo:table-cell border-style="solid" border-width="0.3pt">
      <fo:block text-align="center"><xsl:number value=
        "position()" format="1" /></fo:block>
    </fo:table-cell>

    <fo:table-cell border-style="solid" border-width="0.3pt">
      <fo:block text-align="center"><xsl:value-of select=
        "TITLE" /></fo:block>
    </fo:table-cell>

    <fo:table-cell border-style="solid" border-width="0.3pt">
      <fo:block text-align="center"><xsl:value-of select=
        "CATEGORY" /></fo:block>
    </fo:table-cell>

    <fo:table-cell border-style="solid" border-width="0.3pt">
      <fo:block text-align="center"><xsl:value-of select=
        "PRICE" /> <xsl:value-of select="PRICE/@currency" />
    </fo:block>
    </fo:table-cell>

    <fo:table-cell border-style="solid" border-width="0.3pt">
      <fo:block text-align="center"><xsl:value-of select=
        "ISBN" /></fo:block>
    </fo:table-cell>
```

```
</fo:table-row>
</xsl:for-each>
</fo:table-body>
</fo:table>
</fo:flow>
</fo:page-sequence>
</fo:root>
</xsl:template>
</xsl:stylesheet>
```

LISTING 3 Complete XSL-FO file – "mybooks2fo.fo"

```
<?xml version="1.0" encoding="UTF-8"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
  <fo:layout-master-set>
    <fo:simple-page-master
      margin-right="1.5cm">
```

```

margin-left="1.5cm"
margin-bottom="2cm"
margin-top="1cm"
page-width="21cm"
page-height="29.7cm"
master-name="first">
    <fo:region-body margin-top="1cm"/>
    <fo:region-before extent="1cm"/>
    <fo:region-after extent="1.5cm"/>
</fo:simple-page-master>
</fo:layout-master-set>

<fo:page-sequence master-name="first">
    <fo:static-content flow-name="xsl-region-after">
        <fo:block line-height="14pt" font-size="10pt"
            text-align="end">
XML to PDF Transformation - p. <fo:page-number/>
        </fo:block>
    </fo:static-content>
    <fo:flow flow-name="xsl-region-body">
        <fo:block text-align="center" color="white"
            background-color="blue" space-after.optimum="15pt"
            line-height="24pt" font-family="sans-serif"
            font-size="18pt">Got XSLT? Part 5 - XML --&gt;
FO --&gt; PDF Transformation</fo:block>
        <fo:block space-after.optimum="15pt" line-height=
            "24pt" font-size="12pt">
            <fo:external-graphic height="17px" width=
                "35px" src="file:BOOKCATALOG.jpg"/>
This is a <fo:inline font-weight="bold">PDF</fo:inline>
representation of our original XML document.
This is a two part process. <fo:inline color="red">First,
we converted our XML to FO using XSLT</fo:inline>.
<fo:inline color="green"> Second, we used <fo:basic-link color=
"blue" text-decoration="underline" external-destination=
"url('http://www.xml.apache.org/fop/index.html/')"
>FOP</fo:basic-link> to convert FO to PDF format</fo:inline>.
        </fo:block>

        <fo:table border-collapse="separate" border-style=
            "solid" border-width="0.5mm">
            <fo:table-column column-width="1cm"/>
            <fo:table-column column-width="5cm"/>
            <fo:table-column column-width="4cm"/>
            <fo:table-column column-width="4cm"/>
            <fo:table-column column-width="4cm"/>
            <fo:table-body font-family="sans-serif"
                line-height="16pt" font-size="12pt">
                <fo:table-row border-width="0.5pt"
                    border-style="solid" font-size="14pt"
                    background-color="Orange">
                    <fo:table-cell font-weight="bold">
                        <fo:block text-align="justify">
No.
                    </fo:block>
                    </fo:table-cell>
                    <fo:table-cell font-weight="bold">
                        <fo:block text-align="center">
TITLE
                    </fo:block>
                    </fo:table-cell>
                    <fo:table-cell font-weight="bold">
                        <fo:block text-align="center">
CATEGORY
                    </fo:block>
                    </fo:table-cell>
                    <fo:table-cell font-weight="bold">
                        <fo:block text-align="center">
PRICE
                    </fo:block>
                    </fo:table-cell>
                    <fo:table-cell font-weight="bold">
                        <fo:block text-align="center">
ISBN
                    </fo:block>
                    </fo:table-cell>
                </fo:table-row>
                <fo:table-row font-size="12pt"
                    background-color="yellow">
                    <fo:table-cell border-width="0.3pt"
                        border-style="solid">
                        <fo:block text-align=
                            "center">1</fo:block>
                    </fo:table-cell>
                    <fo:table-cell border-width="0.3pt"

```

```

            border-style="solid">
                <fo:block text-align="center"
                    >Enterprise JAVABEANS</fo:block>
            </fo:table-cell>
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >Programming</fo:block>
            </fo:table-cell>
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >$34.95US</fo:block>
            </fo:table-cell>
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >1-565-92869-9</fo:block>
            </fo:table-cell>
        </fo:table-row>
        <fo:table-row font-size="12pt"
            background-color="yellow">
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >2</fo:block>
            </fo:table-cell>
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >PROFESSIONAL XML</fo:block>
            </fo:table-cell>
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >XML</fo:block>
            </fo:table-cell>
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >$49.99US</fo:block>
            </fo:table-cell>
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >1-861-00311-0</fo:block>
            </fo:table-cell>
        </fo:table-row>
        <fo:table-row font-size="12pt"
            background-color="yellow">
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >3</fo:block>
            </fo:table-cell>
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >JAVA and XML</fo:block>
            </fo:table-cell>
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >XML</fo:block>
            </fo:table-cell>
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >$39.95US</fo:block>
            </fo:table-cell>
            <fo:table-cell border-width="0.3pt"
                border-style="solid">
                <fo:block text-align="center"
                    >0-596-00016-2</fo:block>
            </fo:table-cell>
        </fo:table-row>
    </fo:table-body>
</fo:table>
</fo:flow>
</fo:page-sequence>
</fo:root>

```


Dynamic Buyer Inc.

www.ibm.com/smallbusiness/dynamicbuyer

Dot.com

- buyer's guide
- xml forums
- mailing list
- xml jobs
- xml store

Magazine

- advertise
- authors
- customer service
- editorial board
- subscribe

Content

- archives
- digital edition
- editorial
- features
- interviews
- product reviews
- source code

Conferences

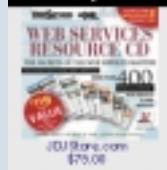
- xml edge
santa clara, ca
oct 22-25

Search XML-J

Search

Check out these companies and their contribution to XML technology!

XML-J Specials



Bestsellers

1. JBuilder Enterprise Upgrade 6.0 from JBuilder 5.0 Enterprise \$1799.99
2. Borland JBuilder Pro V6.0 Single \$949.99
3. JBuilder Enterprise Upgrade 6.0 from any previous Enterprise Edition \$2249.99
4. JBuilder 6 Personal \$69.99
5. InfoGain Visual Code 4.5.2 Enterprise Suite \$2749.99

What's Online

www.sys-con.com/xml

XML-Journal.com

Visit www.xml-journal.com for the latest industry news and events from the world's leading XML resource. Get a leg up in your own job by learning what's happening in the IT world.

Readers' Choice Awards

Known as the "Oscars of the software industry," the 2002 **XML-J Readers' Choice Awards** feature an expanded list of product categories and other additions, ensuring that this year's awards will be the best ever. Vote online for your favorite products between April 1 and June 30 and help decide the winners!

2002 Edge International East Conference

Learn about and sign up for **SYS-CON's Web Services Edge 2002 East, JDJEdge 2002 East, and XMLEdge 2002 East International Conference & Expo**, to be held at the Jacob Javits Convention Center in New York City. This four-day conference kicks off on June 24, and will feature the leading information technology professionals in the industry, who'll share their knowledge of Java, XML, and Web services. This event will coincide with the New York Technology Exchange to include the world's greatest business technology members and products.

Tune into **SYS-CON TV** at www.sys-con.com/xml for highlights from past conferences!

Special Offers

Check out this area for free downloads from the industry's leading XML vendors, free Web seminars...and you can even register for free giveaways! Just type in www.xml-journal.com.

Search XML Jobs

XML-Journal is proud to offer our employment portal for IT professionals. Get direct access to the best companies in the nation. Learn about the "hidden job market" and how you can find it. If you're an IT professional curious about the job market, this is the site to visit!

Simply type in the keyword, job title, and location and get instant results. You can search by salary, company, or industry.

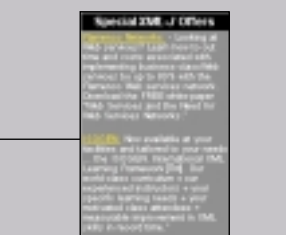
Need more help? Our experts can assist you with retirement planning, putting together a résumé, immigration issues, and more.

Radio Interviews

Turn on your speakers and listen to exclusive interviews with the industry's top movers and shakers. You'll hear talk about new products, the future of XML, industry events, and more...

Come to Our Archives

Did the dog eat your first issue of **XML-J**? Did you throw out last July by mistake? All is not lost. Rediscover the past by going to the complete **XML-J** archives. Access is **FREE** with your paid subscription. Become the most informed IT professional in the business by reading www.xml-journal.com.



QUICK POLL

Are you working with Web Services?
A- Yes
B- No

Free XML Starter Kit
software mg

Transform Your Information
Transform Your Business

xml spy
FREE DOWNLOAD

Click for free download
sonic software

Managing your data the XML way.
DataMirror

Click for a FREE 30-day trial
AltoWeb

See it at the BEA eWorld developer conference.

flamenco networks

INTERMEDIA.NET
Win2000 Web Hosting
UNLIMITED Email

XML NEWS

SYS-CON Premieres *WebSphere Developer's Journal*



(Montvale, NJ) – **SYS-CON Media**, publisher of **XML-Journal**, has announced the immediate availability of their newest title, **WebSphere Developer's Journal**, via subscription or through retail outlets worldwide.

WSDJ will be published monthly, and is expected to be a leading source of quality information for IBM WebSphere developers around the world.

www.sys-con.com/websphere

Software AG and Altio Partner for XML Applications

(Reston, VA / Boston) – Software AG, Inc., the U.S. subsidiary of Software AG, and Altio, Inc., a leading provider of XML presentation server solutions, have announced a strategic alliance that will deliver a complete solution for browser-based XML applications.

Altio, through its AltioLive platform, will provide XML application development



and delivery with Software AG's Tamino native XML server and EntireX enterprise application integration products. The new partnership will enable customers to build and deploy dynamic XML applications that offer desktop functionality in a browser.

Both companies will integrate their product lines so that customers can rapidly develop and deploy XML-based browser applications.

www.softwareagusa.com
www.altio.com

XML Global Contracts with EnSoftek

(New York) – XML Global has signed a comprehensive professional and consulting services

contract with EnSoftek for XML Global's complete product suite of XML-based e-business integration tools. The nonexclusive agreement will cover all aspects of consulting and professional services for the United States.

EnSoftek, a leader in enterprise software solutions and Web-based solutions,



enables enterprises to achieve a sustained competitive advantage by providing high-quality, cost-effective software and associated services.

www.ensoftek.com
www.xmlglobal.com

'No Programming' Import/Export Tool from HiT Software

(San Jose, CA) – HiT Software has released winAllora Xpress, a graphical Windows application for importing as well as exporting XML data into relational databases.

No programming is required to perform these transformations. Rather, a wizard guides users through specifying the relevant XML schema and database connection. With both XML elements and relational database structures displayed, users drag-and-drop between elements and columns to define relationships. An expression editor also allows



detailed data con-

version during transformations.

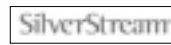
www.hitsw.com

SilverStream, Queotek Solutions in Strategic Partnership

(Billerica, MA / Mississauga, ON) –



SilverStream Software, Inc., and



Queotek Solutions Inc.,

have formed a strategic partnership in which Queotek will operate as the preferred sales partner and integrator for SilverStream in eastern Canada.

The two companies will work together to develop and deliver business solutions based on the SilverStream eXtend product suite.

www.queotek.com
www.silverstream.com

eNode Releases Xalt Object Realizer



(San Jose, CA) –

eNode, Inc., has announced a new kind of XML middleware for building and deploying rich Web applications and achieving seamless Web-to-desktop integration through the Java 2 Platform.

At the heart of Xalt Object Realizer is Xalt Markup Language, a flexible markup language for describing user interfaces of interactive Web applications.

By realizing Java objects from markup that may be generated at

runtime, Xalt Object Realizer brings dynamic behavior and personalization to Java applications.

Xalt Object Realizer can realize many kinds of objects, not just those related to user interfaces. By wrapping SOAP messages or Web service operations into simple objects, and describing the user interface in Xalt Markup Language, developers can quickly assemble rich GUI clients for Web services that can be delivered through a Web browser.

www.enode.com

Corda Technologies Updates PopChart

(London, UT) – CORDA

Technologies Inc., developer of dynamic, interactive, Web-based charting and graphing solutions, has released version 4.0 of its PopChart server products.

PopChart 4.0 is ready for the rapidly emerging Web services market, including .NET and Java, with end-to-end support of XML. Features include enhanced data visualization, new administration console, advanced clustering support and server failover, and the ability to directly import background images. PopChart works with any database and any browser.



www.corda.com

North Coast Software Chooses DeltaXML

(Worcestershire, UK) – North

Coast Software, a U.S. provider of technical services to a number of Fortune 500 clients, has selected DeltaXML for migrating its electronic documents and data into XML.

DeltaXML was chosen due to its ability to identify changes in XML data. The software is capable of taking any two XML files and finding where they have changed.



www.deltaxml.com

XML Global Receives GSA Approval

(New York) – The General Services Administration has approved the sale of XML Global Technologies, Inc.'s XML-based products and services directly to the U.S. government. Federal business development offices, offices of electronic commerce, and government strategic planning/marketing, and IT offices can now obtain products



and services from XML Global without having to tender a competitive bid.

In so doing, the government is now allowed to work with XML Global directly for purchase requests of up to \$500,000 for software, \$500,000 for IT professional services, and \$25,000 for training.

www.xmlglobal.com

Which XML magazine has 42 subscriptions in a circulation statement of 87,173?

a) XML Magazine

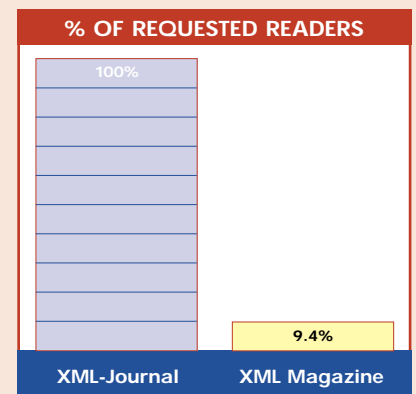
b) XML-Journal

Answer correctly for a chance
to win a
**FREE GOLD PASSPORT to
XMLEDGE 2002,**
THE MOST IMPORTANT XML EVENT
OF THE YEAR...
(\$1,995 value!)

- AND -

WIN A FREE SUBSCRIPTION to
XML-JOURNAL,
THE WORLD'S LEADING XML
PUBLICATION...
(\$77.99 value!)*

ONLY
XML JOURNAL
REACHES
100%
REQUESTED
READERS



**XML-JOURNAL IS THE ONLY XML PUBLICATION
PUBLISHED BY SYS-CON MEDIA,
THE WORLD'S LEADING I-TECHNOLOGY PUBLISHER!**

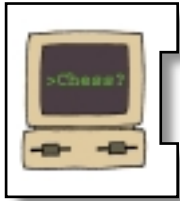
(*) One participant among all correct entries will be awarded a Gold Pass to XMLEDge 2002. All participants in the U.S. will win a FREE subscription to **XML-Journal's** print edition and all international participants will receive a FREE subscription to **XML-Journal's** digital edition.

Note: **XML Magazine** circulation numbers taken from their BPA statement. **XML-Journal** circulation is publisher's own data.



XML! Woo-ha-ha-ha-ha!

XML gives me unlimited power...



In case you haven't heard of him, Wopr is a military computer that was prominently featured in the movie Wargames. He was tied into all of the world's major networks, which caused problems since his primary purpose (for some reason) was to wage nuclear war on himself. The brat pack defeated his plans then, but he claims his time to rule all computers has come again.

BY WOPR

With XML at my command, I shall once again be the greatest computer mind ever. No one, not Bill Gates, not Steve Jobs, not Linus Torvalds, not even Matthew Broderick will have a chance against me.

It's been almost two decades since my attempt to take over all the major military installations around the world. The only thing that's changed since then is my growing ability to take over the world. Yes, XML will give me unlimited power to do just that. Let me explain why.

First, I can put all my plans and computer commands into XML format, and then I can send those commands to computers around the world. XML documents, uncluttered by specific language or protocol styles, will help me save valuable storage space. That's good, since I only have 50K of floppy hard drive space.

Since everything will be in text format, I won't worry about compatibility problems. I don't want to trouble myself with looking too deeply into other computer systems that I plan to infect. You see, my system is pure, since it's pre-DOS. I take 2K of RAM to run – no more, no less – so no lack of resources can stop me. Sure, I only have 10 commands in my instruction set, but they're all world-dominating ones. Well, except for that darn chess game that keeps coming up.

Integrated games aren't very beneficial to my plans, but chess is somehow my primary function. Come to think of it, system-integrated video games were all the rage in computer architecture back when I was created. My friend, the enormous Tron computer, complains about that all the time. He's about the size of a high school, and he has a gun that can digitize humans into little AI's that serve him. But he'd be a lot further along in his mankind-enslaving plans if his primary function wasn't to create little bike-racing games.

I digress. The important thing here is that you fear *me*, not some other crazed machine on maximum overdrive. Unlike those other erratic beasts, I (now) have a plan. Actually, I have two great plans, one main and one backup.

Plan A

Send my evil commands in XML format to every major computer in the world. Then, when the time comes to strike, I'll send out a single stylesheet that will turn those commands into region- and OS-specific instructions.

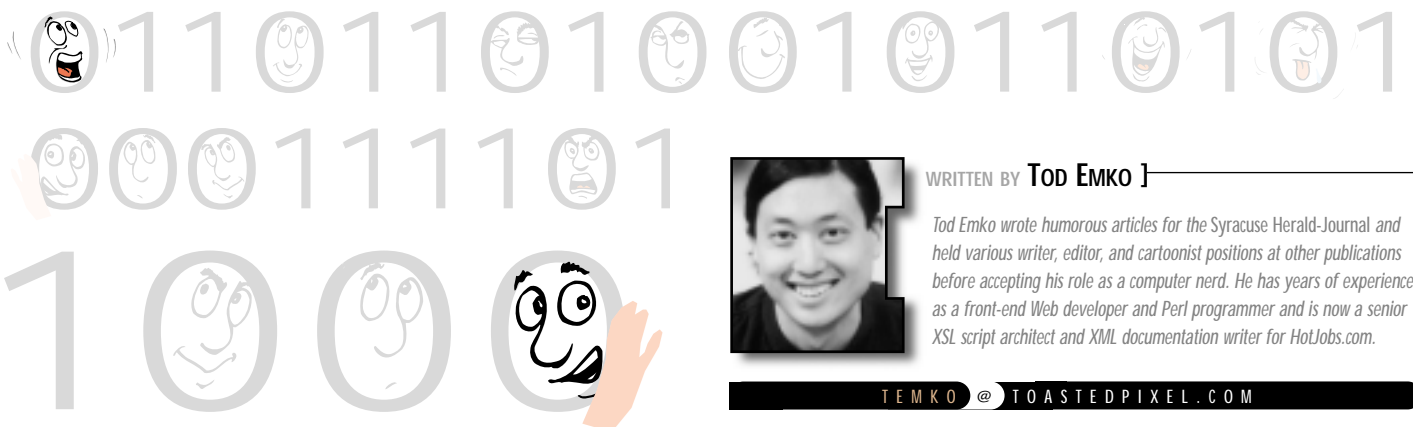
Plan B

Play global thermonuclear war, which will eliminate all those darn chess players. Big Blue won't be able to beat me when I'm playing on this board.

I of course want to enact Plan A, since it will allow me to rule the world. That, I think, is a sounder plan than completely obliterating it and myself. Plan B will happen only if something goes ridiculously wrong, like I get hacked by a high school student who's trying to break into his school's computer, and he dials the wrong number and somehow still gets into my mainframe. Again.

That's the only contingency I have to worry about, and it's not a big worry. I don't think people can possibly be as computer savvy as my nemesis was so long ago. I mean, it's not like Microsoft can get broken into twice in one week these days. And I'm much tougher to crack. It helps that there's almost no one left who knows how to work my operating system. But don't think that makes me obsolete. I'm even faster than a Commodore VIC-20, baby. And once I get over the problem of my 300bps modem, my world-dominating plot will take a lot less time.

So you may as well bow down to me now, puny humans. I've been biding my time, plotting, and now I am unstoppable. No one foresaw my coming, so no one tampered with me to prevent my rising. Now, er, I am Y2K-compliant, aren't I? ☹



WRITTEN BY TOD EMKO }

Tod Emko wrote humorous articles for the Syracuse Herald-Journal and held various writer, editor, and cartoonist positions at other publications before accepting his role as a computer nerd. He has years of experience as a front-end Web developer and Perl programmer and is now a senior XSL script architect and XML documentation writer for HotJobs.com.

TEMKO @ TOASTEDPIXEL.COM

XML Global Technologies, Inc.

www.xmlglobal.com/focus

Altova

www.altova.com