DO YOU REALLY HAVE TO CHOOSE?

August 2002 Volume 2 Issue 8

WebServices
.NET J2EE XML JOURNAL

WSJ2.COM

Web Services Edge
International Web Services Conference & Expo

Jacob Javits Convention Center
June 24-27, 2002

2002 Show Report

Page 62

RETAILERS PLEASE DISPLAY
UNTIL SEPTEMBER 30, 2002
$6.99US $7.99CAN
08
0 71486 03420 9
SYS-CON MEDIA

FOCUS ON .NET & J2EE

IBM

www.ibm.com

Altaworks

www.altaworks.com

WebServices JOURNAL
.NET J2EE XML

INTERNATIONAL ADVISORY BOARD
Jeremy Allaire, Andrew Astor, Steve Benfield, Philip DesAutels, Graham Glass, Tyler Jewell, Paul Lipton, Norbert Mikula, Frank Moss, George Paolini, Simon Phipps

TECHNICAL ADVISORY BOARD
JP Morgenthal, Andy Roberts, David Russell, Ajit Sagar, Simeon Simeonov, Richard Soley

EDITORIAL
EDITOR-IN-CHIEF
Sean Rhody sean@sys-con.com
EDITORIAL DIRECTOR
Jeremy Geelan jeremy@sys-con.com
INDUSTRY EDITOR
Norbert Mikula norbert@sys-con.com
PRODUCT REVIEW EDITOR
Joe Mitchko joe@sys-con.com
.NET EDITOR
Dave Rader davidr@fusiontech.com
EXECUTIVE EDITOR
Gail Schultz gail@sys-con.com
MANAGING EDITOR
Cheryl Van Sise cheryl@sys-con.com
EDITORS
M'lou Pinkham mlou@sys-con.com
Nancy Valentine nancy@sys-con.com
ASSOCIATE EDITORS
Jamie Matusow jamie@sys-con.com
Jean Cassidy jean@sys-con.com
ASSISTANT EDITOR
Jennifer Stilley jennifer@sys-con.com

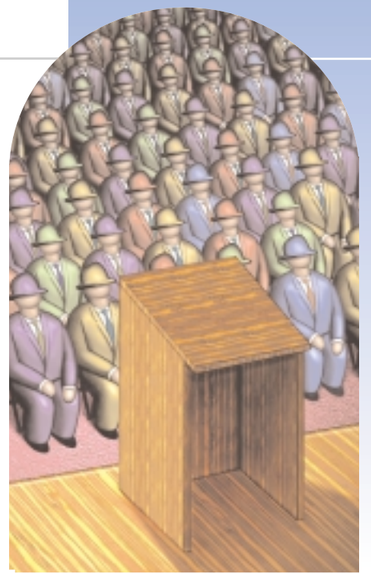
PRODUCTION
VP, PRODUCTION & DESIGN
Jim Morgan jim@sys-con.com
LEAD DESIGNER
Richard Silverberg richards@sys-con.com
ART DIRECTOR
Alex Botero alex@sys-con.com
ASSOCIATE ART DIRECTORS
Cathryn Burak cathyb@sys-con.com
Louis Cuffari louis@sys-con.com
Aarathi Venkataraman aarathi@sys-con.com
ASSISTANT ART DIRECTOR
Tami Beatty tami @sys-con.com

CONTRIBUTORS TO THIS ISSUE
Ben Bernhard, Scott Dietzen, Jeannine Hall Gailey, Murali Janakiraman, Ashish Kumar, Sean Rhody, Mike Rosen, Mike Sick, Nigel Thomas, Chandu Thota

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

SYS-CON MEDIA

Platforms



Written by
Sean Rhody

Author Bio:
Sean Rhody is the editor-in-chief of **Web Services Journal**. *He is a respected industry expert and a consultant with a leading Internet service company.*
SEAN@SYS-CON.COM

In the world of Web services, the question of the month is about platforms. Does the Web services world resemble the political system of the United States, with a two-party system, or is it a more free-wheeling system of coalitions and multiple interests?

Of course, I'm referring to platforms that are technical, not political, although similarities exist on a political level. This month our focus is on J2EE and .NET as platforms for Web services.

I had my share of discussions as to whether we're talking about Java or J2EE. Some tend to think of Java as a language, and then believe a comparison between Java and C# is appropriate. I think it's truly J2EE as a platform, the totality of the enterprise in Java, that is most comparable to .NET.

Even so, there are notable differences, similar to the differences between Republicans and Democrats. .NET is a platform, but it is also real software. J2EE is a platform, but it is a specification, not software.

That distinction tends to start the fracturing of the idea that Web services is a two-horse race. Look inside J2EE and of course we will see a host of vendors, such as SUN, IBM, and BEA, all trying to distinguish themselves in the next round of the game by offering something unique in Web services.

And then we see the WS-I and we start to realize that maybe we're in a coalition type of system after all. Add to that the entrance of the EAI vendors, the messaging vendors, and other players, and our landscape looks more like the Italian system than any political race in America.

Which is okay, because all of the involved parties have a vested interest in establishing standards. They've put their heads on the block – committing to interoperability at the data level at last. These standards finally allow organizations to overcome the inherent marketing position of platform vendors that their platform is the only platform you should use and all of the incompatibilities are the other platforms' fault.

Truth is, all but the smallest organizations have to use multiple platforms, and the reticence of platform vendors to interoperate and cooperate has cost businesses billions of dollars over the years. So instead of a two-party system with both sides determined to differentiate themselves, we have a group system where interoperability is the cornerstone. Obviously, differentiation will take place, but in the services provided through our standards rather than in the absence of such standards.

Obviously, not all platforms are equal. We have two clear heavyweights in J2EE and .NET, with other contenders such as EAI starting to weigh in. So, even though we're focusing on J2EE and .NET in this issue, we need to recognize that there are more platforms out there. And that no one platform is going to meet all our needs. It's more than a two-horse race.

.NET & J2EE

Interoperability in the Web services world

Written by Scott Dietzen

Java 2 Enterprise Edition (J2EE) has achieved critical mass as a platform for developing Web applications. Microsoft's .NET is also a strong contender in the Web world. Today both J2EE and .NET are evolving (via XML, Web services, etc.) from development-only platforms into development and integration platforms – a change that will transform enterprise application integration (EAI) and business-to-business integration (B2B) as we know them today.

Despite the competitive uproar, coexistence of J2EE and .NET will be the norm – most sophisticated IT organizations will deploy on both development platforms. J2EE already has a strong position in enterprise applications, and enterprise ISVs require a code base that can be deployed on whichever hardware/OS their customers demand. On the other side, much of Microsoft's existing ISV and small enterprise marketplace are already jumping on the .NET bandwagon.

The good news is that the promise of easy, Web services–based interoperation between the two platforms continues to pan out. While there are holes in the technology, I'm confident that XML and Web services will continue to deliver interoperability between "Web connected" but otherwise disparate applications. The World Wide Web is simply growing too quickly and too diversely (with phones, handhelds, on-board navigation systems, entertainment/gaming systems, etc.) for it to be in the financial interests of any vendor to break interoperability.

Happily, we've mostly squashed the false perception that J2EE is competing with Web services. In reality, the leading J2EE platforms today provide Web services support that is comparable to, or in the case of server-side support, ahead of, the Web services support .NET provides. Instead, the real conflict will continue to be between Java and C# and between J2EE and the .NET server family.

This leads us to another source of confusion: .NET, like Java, is a family of technologies targeting the spectrum from mobile devices (Pocket PCs) to PCs to servers (at least Windows servers). J2EE will compete directly against the .NET server, while client-side technologies like the Visual Studio .NET user interface builder will interoperate out of the box (via Web services) with server-side J2EE products like the BEA WebLogic Enterprise Platform. Already, companies are mixing-and-matching Java and .NET solutions to best meet their business needs. .NET is finding a sweet spot for programmed user interfaces, while J2EE continues to enjoy its sweet spot for server-side applications.

> "It's really the innovations which are still to come that will ultimately determine whether J2EE or the .NET server is the platform winner"

AUTHOR BIO:
Scott Dietzen is chief technology officer of BEA Systems, Inc.
SDIETZEN@BEA.COM

On the server-side, what factors will influence decisions between J2EE and .NET?

- ***Windows-centric vs multiplatform:*** Multiplatform support remains one of the most fundamental value-adds for Java. While C# and the Common Language Runtime are indeed being ported to UNIX, it seems unlikely that the full .NET server environment will make it there (i.e., Basic "the language" may be multiplatform, but Visual Basic is Windows-centric). At the same time, Java is not at all anti-Windows, since many platforms offer seamless development and deployment on Windows architecture.
- ***Investment protection in APIs:*** Java API standards are derived collaboratively (via the Java Community Process), much as Web services protocols are. Such "collective" innovation is arguably a major reason why the Java community got ahead of Microsoft in the first place (which in turn led to the launch of .NET). The good news for users is that competition will continue to spur innovation in both camps.
- ***Maturity:*** J2EE has four years of scaled, business-critical production under its belt. How many years did it take to fully harden Microsoft SQL Server, which is of comparable complexity to .NET? (And SQL Server was itself derived from a mature product whereas .NET is brand new.)
- ***Ease of development:*** For asynchronous Web services (arguably the Web services sweet spot), products such as BEA WebLogic Workshop (now the subject of Java standardization) provide ease of development.
- ***Application portability:*** While there is some truth to the "write once, test everywhere" complaint against Java, interface specifications and rigorous compliance testing have ensured that the Java community has done a far better job of protecting programming investment than prior standards like SQL and POSIX. What's more, the portability offered by a specific J2EE platform is comparable to that offered by .NET, but applies across virtually all of the hardware and operating systems on the Internet.
- ***Richness of component model:*** J2EE is language-specific, which allows more natural distributed intra-application invocation. (Web services are the better fit for interapplication calls.)
- ***Pluggability:*** J2EE offers service-provider interfaces (SPIs) for messaging and events (JMS), resource adapters (JDBC, J2EE connectors), XML processing (JAX), and so on. These SPIs allow a level of multivendor plug-and-play not afforded by .NET.
- ***Learning curve:*** The typical Visual Basic programmer will find .NET to be a major leap. This disconnect is going to have a multiyear impact on the reworking of Microsoft applications and the retraining of developers. The Java community is simply further down this path, and J2EE applications do not require rework for Web services. What's more, the Java platform has become the likely default in university curriculums.

It's really the innovations which are still to come that will ultimately determine whether J2EE or the .NET server is the platform winner. Innovations like:

- Providing reliability and security for asynchronous Web services
- Dramatically reducing the complexity of business process management and application integration
- Simplifying the aggregation and personalization of user interfaces
- Lowering the overall cost of ownership

The ultimate market leader will be determined by their innovations in these areas.

Written by Chandu Thota

Designing a Generic SOAP Client Using Visual Basic .NET

AUTHOR BIO:
Chandu Thota, a technical lead for Click Commerce, Inc., is coauthor of Understanding the .NET Framework, *from Wrox Press. Chandu is also the founder of www.eSynaps.com, an online .NET XML Web services portal.*
CSTHOTA@ATT.NET

Communicate with any Web service

As XML Web services invade the technology forefront, I almost feel as if an understanding of the SOAP protocol is becoming an everyday essential for me as a .NET developer.

Visual Studio .NET and the .NET Framework do a good job of hiding the details of low-level implementation of SOAP in Web services; however, my inner "SOAP sense" wasn't really satisfied with the little exposure to raw XML messages and SOAP protocol details I'd gotten while implementing Web services using Visual Studio .NET. So I began to implement my own SOAP Client to communicate with Web services while looking over the raw XML messages that are exchanged.

In this article I'll discuss how to build a generic SOAP Client using Visual Basic .NET. What I mean by "generic" is that this SOAP Client can be used to communicate with any Web service. I'll also discuss how to use this client to consume Web services. I chose UDDI (Universal Description, Discovery, and Integration) as a Web service to test my SOAP Client. Let's get started.

Anatomy of SOAP

SOAP is a general application-level protocol designed for application-to-application communication over any underlying transport protocol. The SOAP specification defines the XML formats for messages participating in a SOAP communication. If you have a well-formed XML document that conforms to SOAP specifications, you're ready to communicate with a Web service!

A SOAP message contains the following parts:

- ***Envelope:*** The top-level container representing the message.
- ***Header:*** An optional part of the SOAP message, usually used to carry data unrelated to the main application itself. For example, authentication information could be sent as a header in a SOAP message.
- ***Body:*** Contains mandatory information intended for the ultimate message receiver.

In simple terms, communicating with a Web service is nothing more than formatting an XML document and sending it to the Web service server. Because the SOAP protocol is independent of the transport layer, you can implement the communication at any level, such as TCP/IP, HTTP, etc. For the purposes of this article I'm going to use HTTP as the transport layer for the SOAP communication.

Let's take a look at a sample SOAP message using HTTP, shown in Listing 1. As you can see, it looks like an HTTP-POST request. However, the main differences

> "If you have a well-formed XML document that conforms to SOAP specifications, you're ready to communicate with a Web service!"

between a pure HTTP request and a SOAP request embedded in HTTP are the SOAP envelope payload and the SOAPAction header. With this background, let's move on to design a SOAP Client.

Designing a SOAP Client

Since I'm using HTTP as the transport layer for my SOAP Client, I need HTTP Request and Response objects for communication purposes. I'll define them in my class as shown below. (You can find these types in the namespace System.Net.)

```
Dim objHTTPReq As HttpWebRequest
Dim objHTTPRes As HttpWebResponse
```

The constructor for this class will create an instance of the HTTP Request class and bind to a specific network endpoint (the address of a Web service). So the class constructor expects a string as input, which is "Url" in the following code.

```
Public Sub New(ByVal Url As String)
objHTTPReq = _
CType(WebRequest.CreateDefault(New
System.Uri(Url)), HttpWebRequest)

objHTTPReq.ContentType = _
"text/xml; charset=""utf-8"""
End Sub
```

Now that the basic class is ready, follow these steps to build a SOAP Client:

- Set the HTTP and SOAP headers for the HTTP Request object.
- Build the SOAP envelope around the SOAP body.
- Attach credentials (if needed for the Web service) to the request.
- Send the SOAP request over HTTP to the server.
- Receive the SOAP response from the Web service.

Let's take a look at each of these steps in detail.

Set the HTTP and SOAP Headers (SetHeader)

I'm going to create a function called "SetHeader" to set various header values for the HTTP request. Even in pure HTTP communication there will be several headers associated with each HTTP request (such as "Method", "Accept", etc.). However, SetHeader will be used to set a SOAP-specific header called "SOAPAction", which is required for SOAP communication, along with the other generic HTTP headers. Since the SOAPAction header value is different for each Web service, you need to set it before communicating with any Web service.

As you can see from Listing 2, SetHeader accepts the header name and its value (name-value pairs). So when you add the SOAPAction header to your client, you need to pass both the SOAPAction literal and its value.

Build the SOAP Envelope (BuildEnv)

Now let's take a look at building the SOAP envelope and body. The SOAP body is specific to the Web service and depends on the parameters that the Web service expects, whereas the SOAP envelope is always in a standard format. BuildEnv (see Listing 3) accepts the SOAP body as a string and builds the standard SOAP envelope around it.

Attach Credentials (SetCred)

You can use SetCred (see Listing 4) from the SOAP Client class to set credentials that may be required by some Web services. For example, when you access the UDDI Publish API to publish your business/services you need to pass the credentials to invoke the Web services. In such cases SetCred comes in handy.

SetCred expects credentials (username and password) as input parameters, which it will attach to each HTTP request sent to the Web service.

Send the SOAP Request (Send)

Okay, it's time to write a function to send the actual SOAP request. The function "Send" in the SOAP Client class (see Listing 5) accepts the SOAP request message as a parameter and sends it over HTTP.

The SOAP message is simply written into the HTTP request stream using a StreamWriter class. The StreamWriter class can be found in the System.IO namespace.

Receive the SOAP Response (GetResponse)

The final step in building a SOAP Client is setting the "GetResponse" function. GetResponse is used to read the SOAP response back from the Web service. Since the response returned by the server is going to be another well-formatted XML document that conforms to the SOAP specification, I loaded the returned response into an XML DOM document to parse the SOAP body element.

The code in Listing 6 contains an interesting process. I'm trying to capture the response in the "Catch" block in case of an exception. This will help us in understanding what kind of exception occurred. I will discuss debugging SOAP applications in detail in the next section.

We've designed our SOAP Client. It's time to test it against a real Web service.

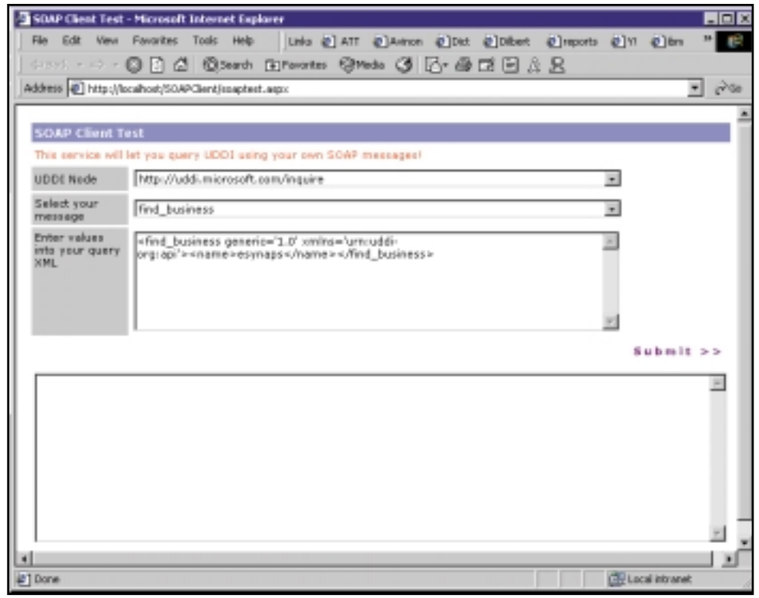


FIGURE 1 **The Web form designed to test the SOAP Client**

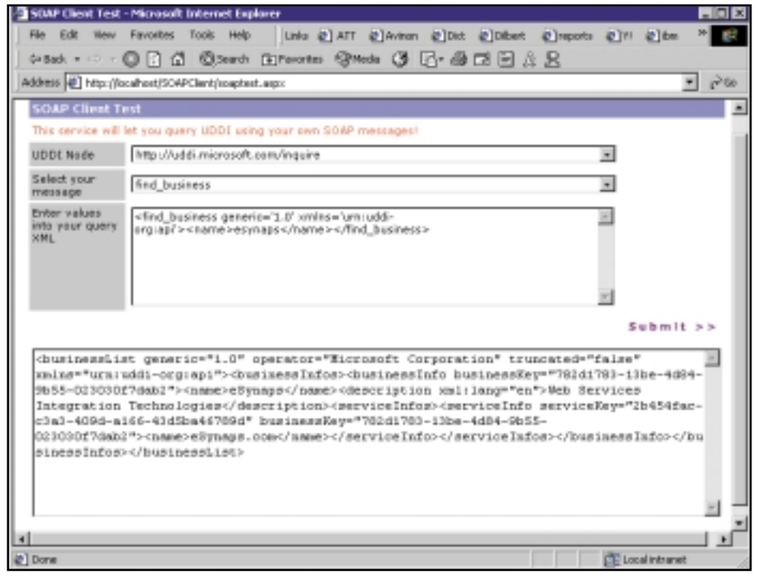


FIGURE 2 **XML returned by the find_business Web service**

Testing the SOAP Client with a UDDI Web Service

If you're new to UDDI services, they're Web-based open directories run by various operators (such as Microsoft, IBM, etc.) that expose information about businesses, services, and related technical interfaces. By querying a UDDI service you can discover the company details and the technical details of the (Web) services they provide. For details about UDDI, you can refer to www.uddi.org.

You need to format the query to the UDDI service in a prespecified XML format before sending it to the UDDI server. The specifications for these XML formats are available at www.uddi.org/pubs/ProgrammersAPI-V2.00-Open-200106_08.pdf. Now let's say, for example, I want to query a UDDI service for a business. I need to use the "find_business" function XML format (refer to the previously mentioned API document). So if I want to find a business by the name "eSynaps", I have to format the query XML as shown below:

```
<find_business generic='1.0'
xmlns='urn:uddi-org:api'>
<name>
esynaps
</name>
</find_business>
```

The above XML is going to be the SOAP body for the UDDI query. Let's move on to test the SOAP Client.

SOAPTest.ASPX

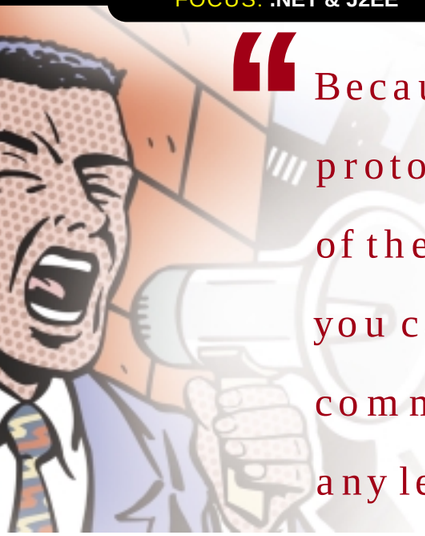
I've created a Web form, SOAPTest.ASPX (available from www.sys-con.com/webservices/sourcec.cfm), to test the SOAP Client (see Figure 1). I hard-coded the vendors (Microsoft and IBM) into the UDDI Node dropdown list. You can choose one of the two nodes to test the SOAP Client, then select the UDDI function you want to invoke. Let's select the find_business function. Since I've already hard-coded the XML formats for all the UDDI query functions (from the UDDI API document), the XML query string format will appear in the query XML text box area of the Web form when you select a particular UDDI query function. (I've used SelectedIndexChanged event.) When you submit the form by clicking on the submit button, the "button_click" function will be executed from the code-behind page.

The button_click function actually instantiates the SOAP Client, sending and receiving the SOAP messages. Let's take a closer look at what's really happening in this function. First, declare the SOAP Client:

```
Dim objMySoap As
MyComponents.MySOAPClient
```

Then create a new instance of the SOAP Client with the selected UDDI node from the Web form:

```
objMySoap = New
```



> “Because the SOAP protocol is independent of the transport layer, you can implement the communication at any level”

```
MyComponents.MySOAPClient(node.Selecte
dItem.Value)
```

Set the HTTP and SOAP headers required to communicate with the Web service: “Method”, “Accept”, and “SOAPAction”:

```
objMySoap.SetHeader("Method", "POST")
objMySoap.SetHeader("Accept",
"text/xml;charset=""utf-8""")
objMySoap.SetHeader("SOAPAction",
""""""")
```

Now prepare the body of the SOAP message by building the envelope around it.

```
strQry =
objMySoap.BuildEnv(query.Text)
```

Then send the XML message to the selected UDDI node using the SOAP Client’s “Send” method, as shown below:

```
objMySoap.Send(strQry)
```

Receive the response from the SOAP Client by calling the GetResponse function and set it to the text area in the bottom portion of the Web form.

```
servOut.Text = objMySoap.GetResponse()
```

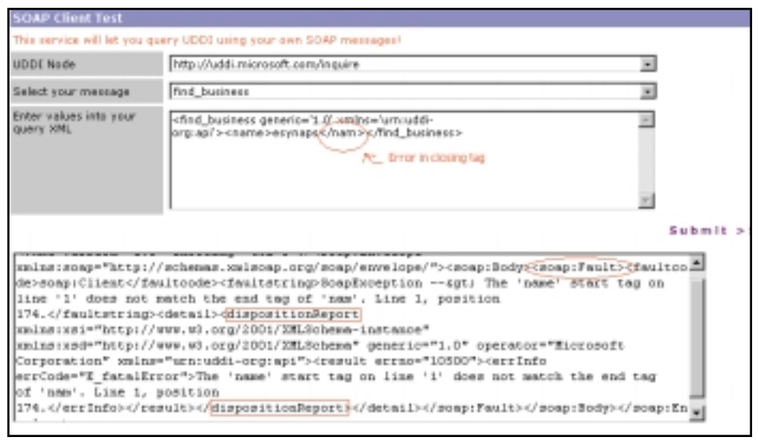


FIGURE 3 | **Example to simulate a SOAP Fault in the Web service**

Now you’ll see the response from the UDDI server. Figure 2 shows the XML being returned by the UDDI find_business Web service.

Now let’s revisit the topic of capturing the response stream from the HTTP Response object in the case of an exception. Even in case of a SOAP exception, the response stream will contain an XML message from the server with a SOAP Fault. This helps us greatly in understanding what went wrong in our communication with the Web service.

To simulate a SOAP Fault, I change the find_business query and submit the form again. The changed query and the Web service response are shown in Figure 3.

Changing the closing tag for the “name” element in the query XML creates an exception in the find_business Web service, causing the server to respond with a SOAP Fault message. To understand the details of the error, take a close look at the “dispositionReport”. The “errInfo” element clearly states that I’m sending a malformed query XML to the Web service.

This way of capturing the response stream to read the dispositionReport in case of an exception really helps in debugging SOAP applications.

Summary

In this article you’ve seen how to design a SOAP Client using Visual Basic .NET and the .NET Framework. The SOAP in .NET supports only implementations over the HTTP protocol, so I chose the HTTP Request class to send the SOAP messages. You also saw the SOAP Client in action when I queried the UDDI node using the find_business Web service. You learned to set the appropriate headers to the SOAP request depending on which Web service you’re going to invoke. Finally, I discussed how to debug your SOAP applications, using the disposition report from the SOAP Fault messages.

Now, since the SOAP Client is ready, you can use it against any Web service to send and receive SOAP messages to understand how the communication works and also to satisfy your “SOAP sense.”

Listing 1

```
POST /StockQuote HTTP/1.1
Host: www.stockquoteserver.com
Content-Type: text/xml; charset="utf-8"
Content-Length: nnnn
SOAPAction: "Some-URI"

<Envelope
 xmlns="http://schemas.xmlsoap.org/soap/envelope/"
 encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <Body>
       <m:GetLastTradePrice xmlns:m="Some-URI">
           <symbol>DIS</symbol>
       </m:GetLastTradePrice>
   </Body>
</Envelope>
```

Listing 2

```
Public Sub SetHeader(ByVal Name As String, ByVal Value As
String)

Select Case (Name)
Case "Method"
     objHTTPReq.Method = Value
Case "Accept"
      objHTTPReq.Accept = Value
Case Else
      objHTTPReq.Headers.Add(Name, Value)
End Select
End Sub
```

Listing 3

```
Public Function BuildEnv(ByVal Body As String) As String

BuildEnv = "<?xml version='1.0' encoding='UTF-8'?>" & _
"<Envelope xmlns='http://schemas.xmlsoap.org/soap/
   envelope/'>" & _
"<Body>" & Body & "</Body></Envelope>"

End Function
```

Listing 4

```
Public Sub SetCred(ByVal UName As String, ByVal UPass
   As String)

Dim objCred As New Net.NetworkCredential(UName, UPass)

objHTTPReq.Credentials = objCred

objCred = Nothing

End Sub
```

Listing 5

```
Public Sub Send(ByVal Message As String)

Dim objStream As System.IO.StreamWriter

Try

objStream = New StreamWriter(objHTTPReq.GetRequestStream(),
Encoding.UTF8)
objStream.Write(Message)
objStream.Close()

Catch

End Try

objStream = Nothing

End Sub
```

Listing 6

```
Public Function GetResponse() As String

Dim objXML As New System.Xml.XmlDocument()

Try

objHTTPRes = objHTTPReq.GetResponse()
objXML.Load(objHTTPRes.GetResponseStream())
GetResponse =
objXML.DocumentElement.FirstChild.InnerXml.ToString

Catch e As WebException

objHTTPRes = e.Response
objXML.Load(objHTTPRes.GetResponseStream())
      GetResponse = objXML.OuterXml.ToString

End Try

objXML = Nothing

End Function
```

Download the code at

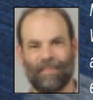
sys-con.com/webservices

Written by Mike Rosen & Ben Bernhard

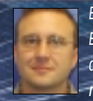
MAKING THEM WORK TOGETHER

A strategy for integrating .NET and J2EE platforms

AUTHOR BIOS:



Mike Rosen is chief technology officer of The Wilton Group, a consultancy specializing in the architecture and design of Web services and enterprise component applications for global corporations. Mike has more than 20 years of experience in distributed technologies and speaks and writes frequently, including Designing E-business Systems and Architectures: A Managers Guide *(2000).*
MIKE@MIKEROSEN.COM



Ben Bernhard engineering manager at IONA Ben's distinguished career has included the development of application servers, object-relational mapping tools, and other middleware and infrastructure products.
BEN.BERNHARD@IONA.COM

Another platform battle is brewing in the industry, or at least that's what the journals would have you believe. Will it be Microsoft's .NET platform, or J2EE supported by Sun, IBM, HP, and others? We don't know, but the more important question is: "Who cares?" Presumably, Microsoft and Sun care, but for the rest of us, the battle of the Web services platforms misses the entire point of Web services. Web services are everywhere (or will be in the near future), and Web services provides interoperability between platforms, languages, and applications. So in terms of interoperability and integration, we don't have to care about what platform we, or our partners, choose.

There will, of course, be valid reasons to choose a platform – enterprise scalability and reliability, tools support, ease-of-use, deployment, and so on, are all good reasons for choosing one platform over another. What we must do is compare the characteristics of the platforms against our requirements. When we do this, we typically find that one platform is better for certain types of applications and the other is better for other types. Generally, enterprises have many different applications requirements and end up with many application platforms. Web services enables us to choose the best platform for our particular application without having to worry about how it will interoperate with applications built on other platforms.

What Is Interoperability?

In the context of .NET and J2EE platforms, we can think of two different types of interoperability, shown in Figure 1. With client/server interoperability, either a .NET client is trying to make use of a Web service provided by a J2EE server application (scenario 1), or a J2EE client is trying to use a Web service provided by .NET (scenario 2). This may be a J2EE client or a J2ME client, such as a

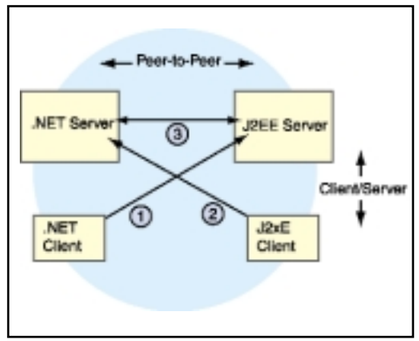


FIGURE 1 | **Logical interoperability scenarios**

> "What we must do is compare the characteristics of the platforms against our requirements"

mobile device. We assume that a .NET client can use .NET services and a J2EE client can use J2EE services.

With peer-to-peer interoperability (scenario 3), one service uses another as part of its implementation. In reality, this is a special case of scenarios 1 or 2 because, for example, client code to access the J2EE service is actually embedded in the .NET service. Peer-to-peer scenarios typically have added requirements for interoperability of security, transactions, and other services that may not be present in client/server scenarios.

Figure 1 illustrates logical interoperability, or the combination of roles and platforms that have to interoperate. What does this interoperation imply? It could mean the ability to:

- Discover Web services
- Connect and exchange messages
- Share or participate in common platform services (security, transactions, etc.)
- Cooperate in extended business applications

These issues illustrate another important dimension of the problem we'll call functional interoperability, or how two applications actually interact. Figure 2 shows the various layers of technology where interoperation must occur for complete application functional interoperability.

- ***Transport:*** At the lowest layer, the transport level, the two applications must agree on a common transport. With Web services, this is most often HTTP or HTTP/S (the secure version), although Web services support other transports as well.
- ***Messaging:*** The next level of agreement must be around the format and encoding of messages sent over the transport. In addition, many applications require an added degree of message-delivery guarantee, so some form of reliable messaging protocol needs to be shared between the sender and receiver.
- ***Platform services:*** Many applications, especially those used to conduct business, will require shared context around security and transactions. Interoperability at this level requires that the platforms share a common set of information for security and transactions, that they agree on how that information is represented, and that it's processed in a similar manner.
- ***Application-level services:*** In addition to security and transactions, some applications will require agreement on application-level services such as licensing and billing.
- ***Application semantics:*** Finally, once all of the transport, messaging, and service layers have been crossed, the applications have to agree on what the actual content of the exchanged messages means. For example, if they're automating a purchasing process, they must agree on what the fields in the message are, the syntax of those fields, and what the fields mean in terms of the purchase order process.

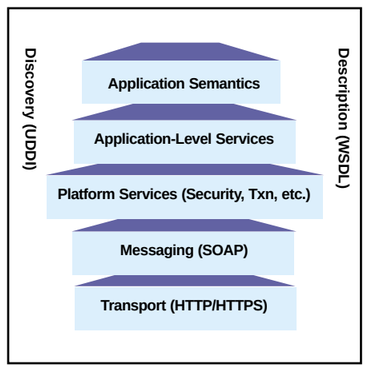


FIGURE 2 **Functional interoperability technology stack**

These layers are involved with the exchange of messages between a client and a service, or between applications. Before the exchange can take place, the client must locate the service. This is a two-part process, the first part being the description of the service and registration by the service provider, and the second being discovery and selection of the service provider by the client.

- ***Description:*** The platforms must agree on the format and schema associated with the Web service description, i.e., what and how it's described and the information required to associate a provider with a specific service. Web services cleverly separates these concepts so, for example, services can be described by industry-standards groups and providers can independently offer competitive versions of those services. In Web services, WSDL describes the messages and operations of a Web service, as well as its binding to specific providers' ports and transports.
- ***Discovery:*** The Web service provider needs a place to publish the information, and the client needs a place to look that information up. In Web services, this is in a UDDI repository, which allows lookup either by service name

> "Web services vendors understand the importance of standards and interoperability"

(white pages model) or by category (yellow pages model). Not only do the client and provider need to agree on the repository and the schema for storage, they also need to agree on the categories or other information used to search for or select a particular service or provider.

Today, Web services software vendors are concentrating on interoperability at the messaging layer and around description and discovery. (Interoperability around HTTP is already established.) The industry and standards community are working hard on security and transactions, but there aren't any agreed-upon specifications or implementations yet. Less activity is taking place around application services. Finally, some industry groups, like AIGI for the automotive industry and HL7 for healthcare, are working on providing XML Schema and definitions at the application semantics levels. Many of these activities are focused on using ebXML, sponsored by the United Nations and OASIS (see www.ebxml.org), to provide a framework for the lower levels of interoperability.

Interoperability at the SOAP Level

SOAP (Simple Object Access Protocol) is used to describe the Web services messages exchanged between a client and a service provider. Unfortunately, SOAP isn't quite that simple; SOAP isn't always SOAP. There are four different encoding and use styles that can be applied to SOAP messages:

- RPC/Encoded (common)
- RPC/Literal
- Document/Encoded
- Document/Literal (more robust with XML schemas)

These types identify whether the Web service is being used in an RPC-style exchange or a document-style exchange and how data is encoded.

In an RPC-style Web service, the input parameters are specified and provided separately (much like a typical object-method signature). In a document-style Web service, the input data is assembled in an XML document that is then passed as a single parameter to the Web service.

In the early use of Web services (where they are used to expose simple J2EE or COM objects), and with simple Web services, the RPC style is quite common. However, as Web services are applied to more-complex and higher-value business services, a different interface style is being adopted. This conveniently corresponds to the evolution of XML from the use of DTDs to the more semantically powerful use of XML schemas. Moving forward, we expect to see Web services using XML schemas to describe their data exchange and to use the document style of encoding to pass that data.

Most SOAP implementations don't support all four of the encoding styles, so for interoperability at the SOAP level, it's important to have two implementations that support an encoding in common. This is true whether it's two different J2EE-based products, or J2EE and .NET. In the latter case, .NET supports the RPC/Encoded and the Document/Literal formats, so it's important to choose a J2EE product that supports at least these two formats as well.

Interoperability and Discovery

Discovery of Web services takes place at two different but equally important times. Most often, we talk about discovery at runtime and a client looking up a Web service in a UDDI repository. We also need to find a Web service during development time so we can include it in our own client or service program.

Visual Studio .NET (VS. NET) uses Web Services Inspection Language (WSIL) to make external Web services available to a project. WSIL allows a server to describe all of the different Web services offered on that server, and to point to the appropriate WSDL for each one. When a developer provides VS .NET with the URL to the server, it can then inquire about the services offered and dynamically build a list of services that can be included into a VS .NET project (from a drop-down list). The metadata about those services (interfaces, types, etc.) is also available from the WSDL information referenced by WSIL.

If your interoperability scenario includes using J2EE Web services from .NET (which will be fairly common), then it's important that the J2EE platform support both types of discovery mechanisms, UDDI and WSIL.

Interoperability Testing and WS-I

How can you determine the capabilities of a particular product or platform? Web services vendors understand the importance of standards and interoperability. They recognize that the real value of Web services is that they are everywhere and they work together (much like an underpinning of the Web is that any browser can access any Web page). To this end, a lot of testing has already been done to demonstrate interoperability.

A group called SOAPBuilders began informal testing between different SOAP implementations in late 2001. Several sets of tests were performed around SOAP messages and data types, verifying interoperability for simple data types, constructed data types, and headers. The first round of testing measured interoperability between 27 different SOAP implementations. The second round measured interoperability based on WSDL definitions of the messages. The results of these tests (and all the gory details) can be found at www.xmethods.net/ilab.

One of the outcomes of this effort seems almost obvious in retrospect: all parties need to agree on the same set of specifications. To promote these and other interoperability issues, a new industry-wide organization called the WS-I, or Web Services Interoperability Organization (www.ws-i.org), was formed in early 2002. The group's official charter is to "promote Web services interoperability across platforms, operating systems, and programming languages." An important aspect of this work is the specification of Web services profiles, which are a set of interoperable specification versions. Through the use of profiles, the number of combinations of technologies that need to work together can be reduced to a manageable level. The first profile, WS-I Basic, specifies:

- XML Schema v1.0
- SOAP v1.1
- WSDL v1.1
- UDDI v1.0

Additional profiles will be created as work in Web services specifications continues, not only in the existing specifications (SOAP v1.2, WSDL v1.2), but in the other areas of message extensibility, binary attachments, routing, correlation, guaranteed message exchange, signatures, encryption, transactions, process flow, inspection, discovery, and so on.

Web Services Need to Be Designed

Another issue deals with the type systems supported by .NET and J2EE and their translation into XML as described by WSDL. For example, should the following data structure be mapped to java.util.Vector, java.util.List, or a Java array of SOAPStruct Objects?

```
- <complexType
name="SequenceOfSOAPStruct">
- <complexContent>
- <restriction base="SOAP-ENC:Array">
  <attribute ref="SOAP-ENC:arrayType"
wsdl:arrayType="xsd1:SOAPStruct[]" />
  </restriction>
  </complexContent>
  </complexType>
```

The problem is equally thorny when translating from certain Java or .NET types into XML. There are no simple answers, but when we look into the issue more deeply we see that it's not possible to simply take any existing Java or COM object and expose it as a Web service (despite what the marketing material may say). So, there are two important considerations. First, choose a Web services platform with a data type translation mechanism that is complete and flexible enough to support your requirements. More important, design Web services interfaces first and then map them to a specific implementation. There are several important reasons for this:

- Web services interfaces should be independent of implementation platform.
- We (or someone) may want to implement the same Web service on many different platforms.
- Existing J2EE or COM objects are typically too low level (fine grained) for proper Web services invocation over the Internet.
- The interaction style (lookup, creation, invocation) of J2EE or COM objects isn't appropriate for Web services (and shouldn't appear in the interface).
- Passing object references and other common J2EE or COM constructs is not appropriate for Web services interfaces.

Many reasons not related to interoperability should encourage us to design Web services first, rather than simply map existing J2EE, COM, or other objects to WSDL. Well-designed Web services will provide high-level, loosely coupled interfaces that correctly expose external business functionality while simultaneously hiding internal information and processes from the users of those services. These interface characteristics will allow Web services implementations and functionality to evolve without breaking existing clients and will help to create more reusable services. Conveniently, they also help with interoperability.

Recommendations

The issues of interoperability are complex, and it will take the industry a while to address the different levels. Currently, the focus for interoperability is around the Web services technologies described by the WS-I Basic profile. At the same time, the Web services community is busy advancing the specifications to cover reliability, security, transactions, and application services.

This doesn't mean we can't use Web services today. Many real business applications are using Web services to integrate .NET and J2EE platforms. From these applications, and our analysis of the problem, we can suggest a few basic guidelines to help achieve interoperability.

- ***Use profiles:*** Stick to the profiles published by the WS-I to minimize dependencies between specification versions.
- ***Design:*** Design your Web services first and then map their definitions to specific platform implementations (whether existing or new). This will not only increase interoperability, it will result in better, more reusable Web services.
- ***Choose wisely:*** Carefully examine your application's requirements and choose a Web services platform that meets them. Evaluate scalability, reliability, skillsets, ease-of-use, tools, cross-platform tool integration, performance in interoperability tests, sophistication of data mapping, and so on. Don't get caught up in the hype about the battle for the platform.

Following these principles, you can begin to use Web services in your applications with the confidence that they'll be interoperable with other platform implementations, and with the knowledge that you'll be building sound, reusable Web services for today and the future.

One or Both for Web Services Development

A multiple-choice quiz

Written by Nigel Thomas

Legions of distinguished commentators have already written a zillion words to explain that .NET is either "awesomely easy to use" or "monopolistic"; conversely, what seems like thousands of articles suggest that J2EE is "the only proven scalable platform" or is "dangerously fragmented."

Microsoft would love its decade-and-a-half-old desktop dominance to spread into the server world. It hopes that the Web services technology it's building into the Windows operating system with .NET architecture will help to achieve that – making it as easy to build services in the future as it has been to build GUIs in the past.

Meanwhile, Sun and the Java community haven't been entirely idle; weaknesses in the J2EE platform – particularly around ease of use – have been addressed or at least admitted to. With the help of industry leaders like BEA and Oracle (backed by jump-either-way pragmatists IBM and HP), J2EE is rapidly recovering ground lost to Microsoft's initial *blitzkrieg* assault on Web services standards.

In the face of the widespread zealotry rampant in both camps, how can you decide which of these platforms will give you, your Web services project, and your company the best return on investment?

The Horns of a Dilemma

When you're stuck on the horns of a dilemma, there is no easy answer. Making the right choice for your situation depends on understanding the critical success factors for your project, and matching them to the strengths and weaknesses of each platform.

AUTHOR BIO:

Nigel Thomas is the director of product marketing at SpiritSoft. Before joining the company, he spent five years in consulting, support, and development positions with Constellar and more than eight years at Oracle Corporation.

NIGEL.THOMAS@SPIRITSOFT.COM

The more I thought about the question, the more it sounded like an agony question typically posed to an advice columnist, so I decided to try to answer the .NET/J2EE debate with a slight advice-column twist. I created a multiple-choice quiz that I hope will help you in your decision.

What platform and language skills do your developers already have ?

It's tough to go against the grain. Retraining staff is time-consuming and expensive – Gartner analyst Joseph Feiman says converting from COBOL to Java can cost, on average, more than $50,000 per developer, taking all factors into account. If you're a Microsoft shop, you probably have the right infrastructure to support .NET and will want to stay that way. .NET supports several languages in addition to C# – even COBOL programmers can join in.

a) We already use Visual Studio for development.
b) We program in Java, of course, that's what we all learned in school.
c) We've outsourced or downsized our development group – we hire contractors whenever we need any work done.

What server infrastructure will you be using?

Unless you're building from the ground up, you probably have a great deal of hardware and software infrastructure already in place, which will certainly have a profound effect on your choice. Taking a major hit in platform transition costs may pay off with a lower total cost of ownership over five years, but if you take the hit now, will you still be around in five years to reap the rewards? In today's climate, you want to minimize the time needed to get a return on your investments.

a) We're Microsoft and Wintel through and through, and we'll stay that way.
b) We're predominantly a Linux/UNIX shop, though we do put up with Windows on the desktop.
c) We have a lot of different platforms – including considerable mainframe and other legacy hardware. We just deal with integration problems as they arise.

What kind of "portability" do you want?

J2EE and .NET address completely different types of portability. With Java and J2EE, the hardware and operating system become a commodity – you can cut across from Solaris to Linux, or from AIX to Windows, more or less at will. You can also (with more effort) switch from one vendor's J2EE application server to another – you can even replace the JVM (Java Virtual Machine). So you're never completely locked in to one vendor's hardware or software platform.

With Microsoft .NET, you can replace your Windows server (Dell to HP, for example) to achieve higher performance, but you don't get software platform portability – it has to be Windows, from Microsoft. You do get the freedom to use multiple languages other than Java, such as Visual Basic, Visual C++, C#, and COBOL.

So, what's your choice?

a) Our customers are mom-and-pop shops; whatever the brand, it's bound to be Windows!
b) Vendor lock-in has been a problem for us in the past. We want to be able to look the salesman in the eye and tell him we don't need his branding – we just want the best product that meets the standard.
c) We have to live with the choices that our different departments (or customers) have already made; we don't have the luxury of a clean slate.

> "When you're stuck on the horns of a dilemma, there is no easy answer"

Are you building desktop or server-side applications?

Your choice of platform will be strongly influenced by the kind of applications you're building. An investment bank may need to use complex derivative pricing algorithms that are encapsulated in existing spreadsheets or DLLs; a manufacturer relies on its enterprise resource planning software.

Microsoft already has an effective monopoly of the desktop, with both fat-client applications (typically written using Visual Basic or Visual C++) and thin clients (browser-based apps using Internet Explorer). Java applets and Linux desktops still haven't managed to persuade most of us to move off Windows as the overwhelmingly adopted personal operating system.

For developers who focus on the server, the J2EE platform (though complex and somewhat fragmented) provides a powerful, cost-effective, and well-supported framework for building scalable Web services and applications. J2EE is more mature than .NET in this respect, offering a wider range of implementation choices and better integration (not just through Web services) with legacy applications and databases.

a) We're building complex client-side applications that have to interface with spreadsheets, e-mail, and word processing as well as access Web services–based components.
b) We're building distributed Web services that will expose our existing enterprise applications to our partners; we need to think about throughput, latency, reliability, and recovery.
c) Our services will be accessed from diverse devices, including PDAs and cell phones as well as other servers; we have to be open to all channels of communication with our customers.

Are you writing new code or bundling existing code into Web services?

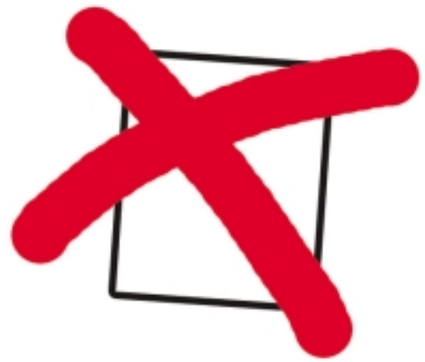
Microsoft's development tools are legendary; they get you from nowhere to a functioning application in double-quick time. But they haven't mastered round-trip and reengineering, nor have they come up with the range of well-thought-out development patterns and paradigms now being implemented in the Java community. Most of all, .NET seems geared toward instant gratification – support for open asynchronous messaging standards like JMS (Java Message Service) is simply not in place.

a) We're going to put our best Visual programmers on the job – they'll soon have it licked.
b) We rely on applications written over the last 10–25 years; we talk to these through asynchronous message queues, and JMS lets us integrate our legacy messaging systems into J2EE.
c) We want to make the best of the infrastructure we've already got, but we're prepared to invest in rewrites where necessary.

How many users will there be, who are they, and where are they?

Distributing complex client applications to hundreds of users can be costly. Should you be thinking about standardized desktop configurations and software asset management? Will Java applets take too long to download? Do you need to think about page caching or a multitier caching framework?

a) We'll have hundreds of users, mostly within the firewall, running on a managed desktop.
b) We expect to have thousands of users on the Internet and we can't be sure what technology they have – other than a browser
c) We're integrating intranet, Internet, and mobile users with our call center and want to offer the same services through every channel.



> "There's a good chance that whichever you pick as your main platform, the other will make an appearance sooner or later"

Are you an independent software vendor?

Both Microsoft and Java platforms support huge – and partially overlapping – economies of software vendors. Which economy would you like to play in? There have been many examples of Microsoft either defeating (Netscape) or absorbing (Hotmail) independent vendors who have trespassed in their economy. Mind you, the same can sometimes happen in the other camp too…

a) We would love to compete with Microsoft – maybe my share options would vest if they decided to acquire us!
b) We're looking forward to extensive "co-opetition" with IBM, Oracle, Sun, HP, BEA, …
c) We think our niche is too specialized for any of the big companies to take too close an interest – we'll just leverage their platform and tools to make life as easy as possible for our customers

How Did You Do?

Mostly (a): you're probably already looking at .NET, even if you haven't started using it on real projects yet.

Mostly (b): it sounds like you should be thinking about J2EE. .NET doesn't really address the complexities of your world.

Mostly (c): Congratulations! You're in the real, confusing world where there are no easy answers. Now, let's get down to the real evaluation and look at the business issues for your project – specific functionality, total cost, return on investment, time-to-market – rather than relying on simple preconceptions. There's a good chance that whichever you pick as your main platform, the other will make an appearance sooner or later.

The Final Choice

All differences aside, you can legitimately choose either or both platforms. "The question is not whether J2EE or .NET is the better architecture," says Yefim Natis, a Gartner analyst. "The question is, how to integrate them, how to make them work together, what are their strengths and weaknesses?"

With Web services standards in place, architects can assemble complex, heterogeneous applications where Microsoft "owns" only the fat-client piece, and the UNIX/Java world takes over – or at least gets a good share of – the server side (and some thin clients). A return perhaps to the typical client/server split of the early 1990s, but with Internet and Web services rather than the LAN as the common connection bus, and with a more complex network of loosely coupled processes.

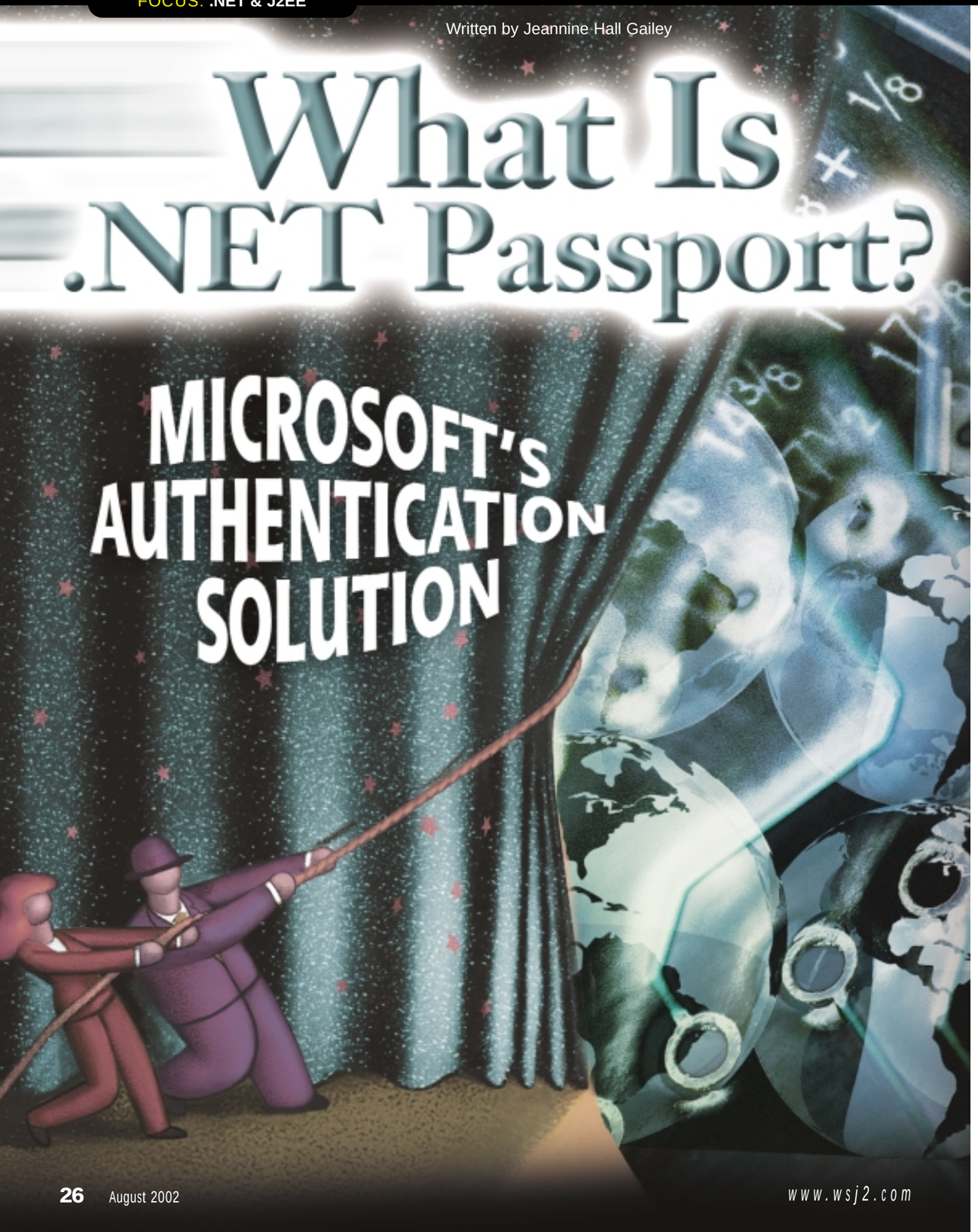
Software vendors – even those like SpiritSoft who are firmly in the Java camp – have to recognize that .NET has considerable merits. Even if we don't wholly approve, in the real world our customers are bound to adopt it to some extent. We can't simply ignore it and hope it goes away.

We can either whole-heartedly embrace it, or we can contain it by offering Web services–based interoperability with J2EE. That way, Java software vendors will ensure that we continue to play a major part in the heterogeneous environments that will inevitably be found in most real enterprises, and that our customers don't get trapped in a vendor dead-end by one side or the other.

Written by Jeannine Hall Gailey

What Is .NET Passport?

Microsoft's Authentication Solution



.NET Passport is a Microsoft-operated service that provides Internet authentication for Web sites, no matter what kind of devices they use for access. It provides reliable Internet authentication and allows users to sign in once to access a variety of .NET Passport–enabled Web sites. In addition, users can save time by using Passport data when registering at new Passport-enabled Web sites. Developers don't have to build and maintain custom authentication mechanisms – Microsoft does the work.

Microsoft launched .NET Passport in 1999, and there are now more than 200 million .NET Passport accounts and over four billion authentications per month. As a precursor for future Web services, .NET Passport has allowed Microsoft to deal with problems of scalability, privacy, and security. Although it's sometimes been a rocky road, they probably now have a better idea of what to expect as they aggressively pursue other .NET Web services, such as the much-anticipated Microsoft .NET Alerts.

Microsoft has stated that .NET Passport will be their Web service authentication solution for .NET. However, there's been an unexpectedly high degree of reluctance on the part of traditional Microsoft partners to deploy .NET Passport, citing customer concerns about Microsoft "owning" their personal data and the complexities of implementation. Since Microsoft realizes authentication is a key component in a Web services world, they're taking positive steps to redesign the .NET Passport service and allay some of these fears.

Internet Authentication

Authentication is the process by which two parties exchange information to prove who they are. For example, most Web sites request that users register to access certain features of their site by creating a login name and password, which are known only by the two parties. When a user returns to a site, he or she must enter a username and password as proof of identity, after which access to the site is granted. Conversely, a Web site may present digital certificates to client Web browsers to prove that they are, for example, really a secure banking site and not a hacker spoofing the bank's identity.

Single Sign-In

Originally, .NET Passport was conceived of to help solve problems created for consumers and Web sites by the existence of individual Web site–maintained authentication schemes that require users to sign in to each site they visit. The basis for the single sign-in (SSI) solution approach was the realization that for every site a consumer wanted to interact with, he or she needed to enter the same information – name, address, phone number, and date of birth – and then create a username and password. Studies have shown that when Internet users have to remember a separate username and password for each Web site, many just use the same information for each site. This means when the password expires at one site, it needs to be changed at every site. The same problem is seen by users who move and then have to update personal information at every Web site they are registered with.

Passport SSI solves this problem by allowing users to enter a minimal amount of personal information (name, address, e-mail address, and birth date) as well as a password for the Passport account, all of which is securely stored by Passport. With the exception of the Passport Express Purchase service, any additional data a site wants to collect must be maintained by that site. To log in to a Passport-enabled site, users present credentials to Passport (via the SSI interface) and get an encrypted ticket cookie containing a Passport Unique ID (PUID) that's decrypted by the site and used to authenticate them. When registering at a new Passport-enabled site, users simply verify that they have a Passport. The site then uses the personal information stored by Passport to create the new account. Users even control how much personal information is shared with the site (the minimum being just the PUID). SSI also provides an additional level of security in that partner sites never have access to the users' authentication information (Passport username and password); they receive only the PUID from Passport.

Privacy Worries

There was a flurry of articles written about .NET Passport and concerns about privacy. This is a case where users can share what they feel comfortable sharing. .NET Passport stores a very limited amount of personal information. When users sign up for a Passport, they're asked for several pieces of personal information. However, the only information that is required to get a Passport is an e-mail address and a password. Although users are invited to provide additional information that can save them time when registering at other Web sites (for example, a .NET Passport–partner site could import address information from Passport as shipping information), they're not obligated to do so. Also, users can view and change these privacy settings at www.passport.com/memberservices.

The .NET Passport Privacy Policy – posted on Microsoft's Passport site, www.passport.com – states that Microsoft will not "mine, rent, sell, publish, or share user data beyond what the users choose." They also claim they won't create and sell reports based on customer data. In an attempt to prove their commitment to consumer privacy, Microsoft has become a participant in the Safe Harbor Agreement, a binding group of privacy agreements in the U.S. and Europe that requires .NET Passport–enabled sites to comply with the Platform for Privacy Preferences Project (P3P). Microsoft also contractually requires that all partner sites that implement .NET Passport have a posted privacy statement, and these sites are also encouraged to register with an independent privacy-assurance group such as TRUSTe. Whether or not to trust Microsoft to fulfill these promises is up to the user. As one of the world's largest providers of end-user software and services, Microsoft has more to lose than anyone if they choose to violate their customer's privacy and trust.

.NET Passport Security

Currently, .NET Passport relies on a cookie-based system, which Microsoft

AUTHOR BIO:

Jeannine Hall Gailey is a full-time freelance writer. Previously the manager of the .NET My Services SDK documentation project team, she worked in technical documentation for seven years. She contributed to Microsoft .NET My Services Specification *(Microsoft Press) and is working on another book. Visit her Web site at www.webbish6.com for code downloads and discussions.*

WEBBISH6@HOTMAIL.COM

> Developers don't have to build and maintain custom authentication mechanisms – Microsoft does the work"

admits is not the ideal design and which they're working on improving. To address the potential vulnerabilities of cookie interception and misuse, Microsoft has taken several steps to improve security in the current version. These new security features of Passport 2.1 are:

- ***Cookie encryption:*** Passport encrypts cookies with site-specific keys to prevent access to user data. Cookies are encrypted using a Triple DES encryption algorithm with 168-bit keys. Each partner site has a unique encryption key, and keys can be rotated to decrease the likelihood of any one site's encryption key becoming compromised.
- ***Cookie timeouts:*** Passport has a default lifetime for the authenticated ticket cookies it creates, but sites can specify a shorter lifetime.
- ***Revalidation:*** Allows a site to verify that a user's e-mail sign-in name is a legitimate and monitored e-mail address.
- ***Secure sign-in:*** Requires that Secure Socket Layers (SSL) be used for all authentication round-trips and allows sites to require an additional security key for authentication.

While these changes have all been steps in the right direction, .NET Passport has a way to go. Fortunately, Microsoft has indicated that future versions will be based on the secure Kerberos v5 protocol, which will be described in greater detail later. The .NET Passport service will also likely be expanded to offer additional authentication mechanisms and security protocols to support digital certificates, smart cards, and even biometrics. The move to Kerberos technology alone will greatly increase the security of .NET Passport–enabled sites. In addition, .NET Passport will become one of many authentication services on the Internet that will compose a federated network of trust brokers based on forthcoming Web services security standards.

What Is .NET Passport Today?

.NET Passport v2.1 features a rather complex COM-based API, somewhat problematic implementation, and a cookie-based security system. It contains the following main components:

- ***.NET Passport Domain Login Server:*** Processes incoming user logins and issues Passport cookies, allows users to sign out and revokes cookies, and redirects new users to the registration server to create new Passports.
- ***.NET Passport Registration Server:*** Creates and manages user accounts.
- ***Passport Manager:*** Principal COM object installed on the partner site servers that provides sign-in logic and handles cookie encryption.
- ***Additional Passport objects:*** COM objects installed at the partner site that provide more specialized features, including Fast Authorization, manual encryption, and thread pooling.
- ***Kids Passport:*** Allows the extension of .NET Passport features to children under 18 only with the permission of an adult; the child's profile is then linked to the adult.

For more information on the .NET Passport SDK available today from MSDN, visit www.msdn.microsoft.com.

How It Works

A .NET Passport–enabled Web site redirects users to a separate .NET Passport Login server, which authenticates users and writes information to cookies stored on a user's machine. The authentication process is as follows:

1. A user arrives at a .NET Passport– enabled site and requests an ASP page.
2. During creation of the page, an instance of the Passport Manager object is created that sends a request back to the user's browser to check for valid Passport cookies.
3. If these cookies can't be found (or when the site requires revalidation), the requested page loads with a sign-in link for Passport displayed. Otherwise, the sign-

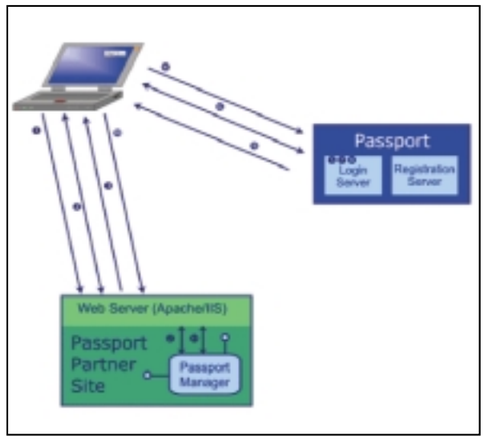
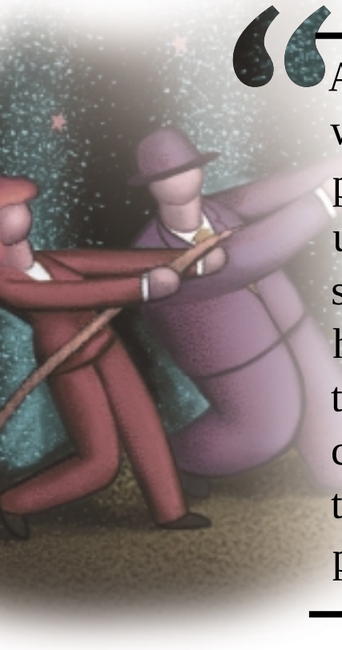


FIGURE 1 | **Authentication process**



> "As one of the world's largest providers of end-user software and services, Microsoft has more to lose than anyone if they choose to violate their customer's privacy and trust"

out link is displayed and the user is considered authenticated.

4. The user clicks the .NET Passport sign-in link to begin the authentication process and is redirected to the .NET Passport sign-in page. Generated by the Passport Manager object, the URL for this redirect contains query strings that pass both the assigned site ID of the Passport-enabled site and a return URL used to return to the site after authentication.
5. The Passport Login server checks whether the site ID and return URL are registered as partner sites. If so, a sign-in page is sent to the user's browser. If not, the authentication fails and the sign-in page is not displayed.
6. The user enters his or her .NET Passport login credentials on the sign-in page and the information is sent, via SSL, to the Passport Login server.
7. If the user can be positively authenticated by the Passport Login server, the server retrieves the user's PUID and Passport profile.
8. The Passport Login server creates the following three cookies that are encrypted using the partner site's encryption key and site ID:
   - ***Ticket cookie:*** Includes the PUID and a time stamp
   - ***Profile cookie:*** Stores the profile information
   - ***Visited Sites cookie:*** Stores a list of sites where the user has signed in

Note: For security, partner sites use the PUID rather than the login name to identify users. Authentication information isn't stored on the user's machine or at the partner site. Profile information is only sent if this option has been requested by the user in his or her Passport account settings.

9. These encrypted cookies are added as query parameters to the redirected URL that's returned to the user's browser.
10. The browser re-creates the encrypted cookies on the user's machine and redirects the user back to the partner site, with the encrypted cookies passed again in the query string.
11. Passport Manager at the partner site uses the site's encryption key to decrypt the cookies and extract the user's PUID and profile information, which can be added to or updated in the site's own database. Figure 1 depicts this authentication process.

It's interesting to note that in this version of Passport, no data is exchanged directly between the Web site and the Passport site. All data flows through requesting client.

Passport Manager

As we saw in the previous discussion, the Passport Manager object plays a crucial role in the authentication process. This COM-based object, installed on the partner sites' servers, is embedded as a server-side object once on each of the site's ASP pages. Passport Manager handles most of the Passport authentication logic with a minimal amount of site-specific programming in your ASP pages. The following shows the server-side ASP <OBJECT> tag used to embed the Passport Manager object:

```
<OBJECT RUNAT="SERVER"
  ID="PassportManager"
  PROGID="Passport.Manager">
```

You could also create the same object in ASP using VBScript as:

```
<%
Dim oPassportMgr
Set oPassportMgr =
Server.CreateObject("Passport.Manager")
%>
```

In addition to the Passport Manager object, each page must include the .NET Passport logo that users click on to sign in and out of the service. When an ASP page is loaded, and after the Passport Manager object has been instantiated, .NET Passport checks the user's computer for the cookies it needs to determine the user's sign-in state. Based on the user's state, the appropriate .NET Passport logo image is downloaded and created on the page (the location of the logo is dictated by Microsoft). If a .NET Passport cookie isn't found on the user's machine, the sign-in logo is displayed. Listing 1 demonstrates how the logo is created on the page.

For an unauthenticated user, Passport Manager generates the following HTML to create the sign-in link and logo on the page:

```
<A
HREF="http://www.webbish6.com/start.asp?
msppchlg=1&mspplogin=http://login.pass-
port.com/login.asp%3Flc%3D1033%26id%3D1%
26ru%3Dhttp%253A%252F%252Fwww%252Ewebbis
h6%252Ecom%252Fstart%252Easp%26tw%3D3600
%26kv%3D1%26ct%3D1022103728%26ems%3D1%26
ver%3D2.1.0173.1%26tpf%3D1872b03e5e3a74a
b72f0a931b38b6718"><IMG
SRC="http://current-
www.passportimages.org/1033/signin.gif"
CLASS="PassportSignIn" BORDER="0"
ALT="Sign in with your .NET
Passport"/></A>
```

In this case, we've specified the URL returned after authentication as strReturnURL. If a return URL value isn't specified, the URL will be the default value set in the registry. In addition to the return URL, Passport supports a number of other parameters on the LogoTag2 method that allow developers to take more control over the login behavior of the Passport Manager, for example, forcing revalidation and specifying that an SSL connection must be used.

The Passport Manager object should always be instantiated in page scope. Instantiating this object at the application or session scope can cause the Passport Manager to lose state for individual users. Passport also offers a solution for high-volume Web sites that experience performance problems when instantiating the Passport Manager at page scope. The Passport Factory object allows you to create Passport Manager Objects from a pool to improve performance. In addition to Passport Manager and Passport Factory, the .NET Passport SDK also ships with additional COM objects, including Passport FastAuth, Passport Crypt, and Passport LookupTable, that support additional .NET Passport functionalities.

In an effort to extend Passport beyond the walls of Microsoft, the Passport Manager has been ported to Apache Web servers running on Linux, Solaris, AIX, HPUX, and FreeBSD. To learn more about these objects, see the Passport 2.1 SDK available from MSDN.

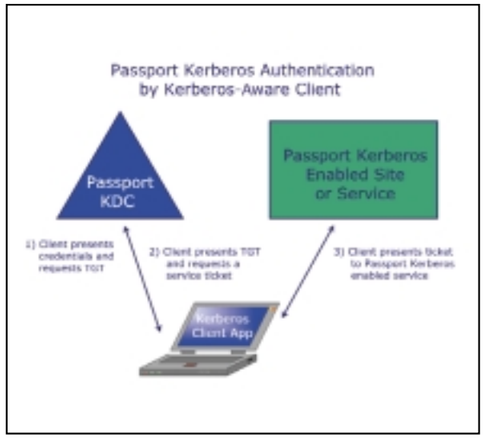


FIGURE 2 | **Passport Kerberos**

Where .NET Passport Is Heading

Following some of the most recent security enhancements, .NET Passport has become a fairly useful Internet authentication service in the B2C realm. However, its reliance on cookies, server-side installed components, and proprietary internal methods will prevent the current Passport 2.1 release from moving beyond end-user Internet applications. Also, as visionaries like Microsoft and IBM attempt to drive the world of Internet computing down the potentially beneficial (but mostly unfamiliar) road toward XML Web services, the functionalities of .NET Passport will certainly need to be expanded. Most importantly, .NET Passport will need to evolve into a true XML-based Web service in order to handle B2B authentication scenarios, where a Web server or service must authenticate against other Web services.

Indications from Microsoft are that upcoming versions of Passport will no longer be based on cookies and COM objects, but will have a true XML/SOAP Web services interface, which will make the service discoverable and easier to program against, especially for UNIX and Linux programmers. Implementation will go from a fairly complicated installation on your corporate servers to perhaps simply reading a WSDL file and being able to call the necessary Web services methods with the encrypted authentication information provided by your users. You'll even be able to accept tickets generated by a trusted, non-Passport authentication service. And best of all, implementing Web services security will follow XML-based standards.

Other new initiatives Microsoft is embracing for its .NET Passport services include adding Kerberos encryption, making .NET Passport a federated service, and adopting new XML Web services standards and protocols.

Kerberos v5 Encryption

Kerberos is a method of securely establishing credentials in a distributed environment by using a third-party "trust broker." This protocol was originally designed by researchers at MIT to better implement security permissions on distributed networks and through secure firewalls. The foundation of secure authentication in Windows 2000 and Windows XP, Kerberos manages the trusts between client applications and service providers by means of a third-party Key Distribution Center (KDC). The KDC has two functions: it acts as both the Authentication Server (AS), authenticating users based on their supplied credentials and distributes ticket granting tickets (TGT); and as the Ticket Granting Server (TGS), distributing service access "tickets" and encryption keys. In the Kerberos world, clients present credentials to the KDC and in return get tickets that are used to access desired resources on a server. For Web services, the KDC functionality will be provided by .NET Passport or a .NET Passport Federated Partner that must conform to the forthcoming Passport Kerberos specification. Figure 2 shows the basics of how Passport Kerberos will work.

Federation

Last year, Microsoft announced its decision to *federate* .NET Passport, that is, to make the .NET Passport service interoperable with other authentication services. Federation is a newly emerging concept in Web services that seeks to promote trust relationships between a variety of authentication infrastructures, such as Kerberos, Public-Key protocols, and Security Information Management (SIM), that are already widely used both in internal corporate networks and on the Internet. Federation will allow .NET Passport-based Web services to accept credentials from any network that has a trust relationship with .NET Passport, potentially expanding the scope of Web services to environments such as corporate intranets. The next release of Passport will support a version of the Kerberos v5 protocol, which will lay the groundwork for federation. There are plans to make federation a reality in 2003, pending the adoption of newly proposed XML Web Services Architecture standards.

Web Service Standards

The overall Web services security model

that .NET Passport is moving toward is being defined by Microsoft- and IBM-sponsored specifications. Built on the Simple Object Access Protocol (SOAP), these protocols will provide an additional, modular layer that will improve the utility of XML Web services in the enterprise by providing standards for message transport, reliable messaging, discovery, security, and trust. The foundation security specification is WS-Security, which defines an end-to-end message security protocol that supports a wide variety of security models. Following this specification, which is currently in the beta-release stage, there will be other protocols that further clarify how federation works, including WS-Trust, WS-SecureConversation, WS-Authorization, and WS-Federation. As Microsoft is a major proponent of these standards, you can be sure that .NET Passport will follow them.

The Next Version of .NET Passport

While many of these Web services–based improvements will not likely occur until 2003, expect to see updates to .NET Passport in the September release of the SDK. This release will include support for .NET Passport Kerberos as well as a Passport Client Runtime that provides rich client support for Passport Kerberos.

To sign up for .NET Passport service for your Web site, go to www.netmyservices-manager.com, which has a wizard that can walk you through getting an ID and setting .NET Passport up on your site. For the .NET Passport SDK, go to http://msdn.microsoft.com/netmyservices and go to the download center. To learn more about .NET Passport and .NET My Services, check out www.gotdotnet.com and www.passport.com.

Listing 1

```
<%
Dim strReturnURL 'specify the return URL
Dim strLogoTag 'calls the Passport Manager LogoTag2 method

strReturnURL=Server.URLEncode("http://www.webbish6.com/start
.asp")
strLogoTag=oPassportMgr.LogoTag2(strReturnURL)
%>

...

<html>
 <head>

 ...
```

```
 </head>
 <body>

...

   <%=strLogoTag%>

...
 </body>
</html>
```

Download the code at sys-con.com/webservices

Reviewed by Sean Rhody

Microsoft Visual Studio .NET Enterprise Architect Edition

As complete a development environment as can be imagined from a single company

About the Author:

Sean Rhody is the editor-in-chief of **Web Services Journal**. *He is a respected industry expert and a consultant with a leading Internet service company.*

SEAN@SYS-CON.COM

Microsoft

CONTACT:
Microsoft Corporation
One Microsoft Way,
Redmond, WA 98052

VISUAL STUDIO .NET PRICING INFO:
Enterprise Architect:
Full Package: $2,499
Version Upgrade: $1,799

ENTERPRISE DEVELOPER:
Full Package: $1,799
Version Upgrade: $1,079

PROFESSIONAL:
Full Package: $1,079
Version Upgrade: $549

URL TO PURCHASE THE PRODUCT:
http://shop.microsoft.com/devtools/default.asp

MAIN SKUS FOR VISUAL STUDIO .NET:
http://msdn.microsoft.com/vstudio/productinfo/overview.asp

PROCESSOR:
PC with a Pentium II-class processor, 450 megahertz (MHz)

OPERATING SYSTEM:
Microsoft Windows NT 4.0 or later

MEMORY:
Microsoft Windows XP Professional 160MB of RAM
Windows 2000 Professional 96MB of RAM
Windows 2000 Server 192MB of RAM
Windows NT 4.0 Workstation 64MB of RAM
Windows NT 4.0 Server 160MB of RAM

HARD DISK:
Standard Edition: 2.5GB on installation drive, which includes 500MB on system drive

PROFESSIONAL AND ENTERPRISE EDITIONS:
3.5GB on installation drive, which includes 500MB on system drive

DRIVE:
CD-ROM or DVD-ROM drive
Display: Super VGA (800 x 600) or higher-resolution monitor with 256 colors

MOUSE:
Microsoft Mouse or compatible pointing device

Web Services Journal received a copy of the latest visual development environment from Microsoft – Visual Studio .NET, Enterprise Architect edition. We loaded the software on a PC running Windows 2000 Server, with 384MB of RAM, but with a fairly slow processor (400MHz AMD K6-III). The recommended minimum is a 450MHz Pentium II class processor with 192MB of RAM under Windows 2000 Server, so this machine was roughly at the bottom of the capability range. Although our machine was somewhat underpowered, we're happy to report that Studio loaded quickly and was fairly responsive, even with IIS, SQL Server, and several other packages running.

The Enterprise Architect edition of Visual Studio is everything you need – plus the kitchen sink. The studio itself is four CDs – five if you count the Windows Components Update disk that updates part of the operating system. Included on another disk is Visio 2002, Architect Edition. Also included in the Enterprise pack is Visual Source Safe, Windows 2000 Server, SQL Server 2000, Exchange Server, Commerce Server, BizTalk Server, and Host Server. Most of the server versions are developers editions, rather than licenses for full product use. Altogether, there are over a dozen CDs in the distribution. The studio alone requires over 2GB of disk space.

The space is well worth it, however, particularly if you're interested in developing Web services. The studio provides support for Visual Basic, Visual C++, and Visual C#, and also includes Crystal Reports and a testing facility called Microsoft Application Center Test. Combined with Visual Source Safe and Visio, the studio provides a complete development environment for design, development, testing, and deployment of Windows .NET code.

Running Some Tests

We put Visual Studio .NET through its paces by developing some simple Web services in Visual Basic. The IDE is a fairly complicated, multipaned application, but fortunately there are wizards everywhere to make your job easier. Each pane contains information about a different facet of the application

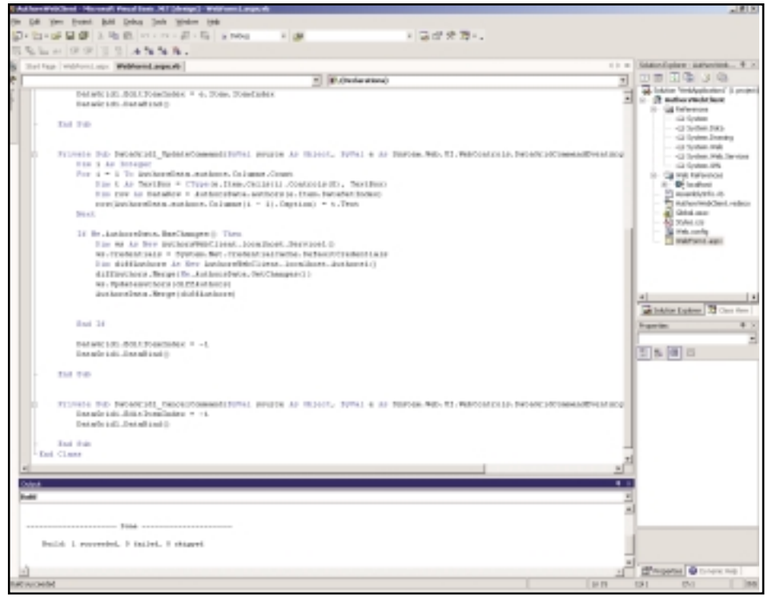


FIGURE 1 **IDE in action**

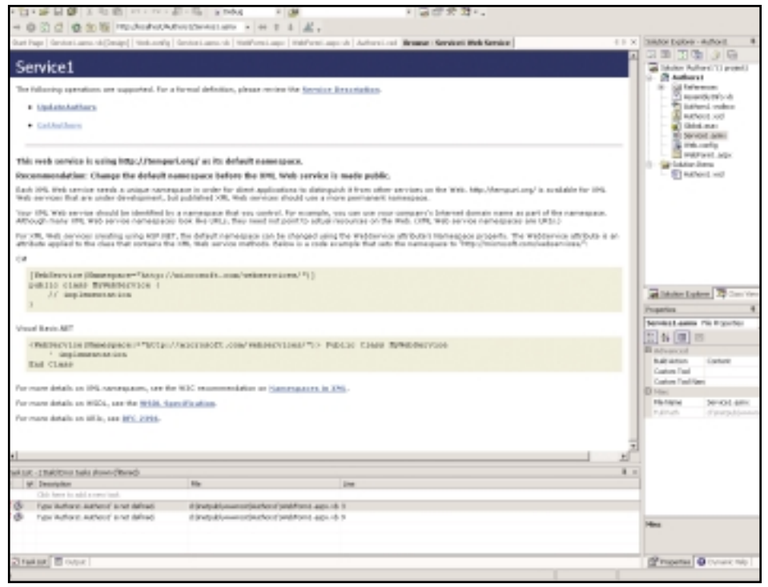


FIGURE 2 **Testing the code without writing your own client**

you're building. The Solutions Explorer is the basic project tree of the application, and shows all the files you've created. A number of files are created when you create any project. Gone are the days where you created a single file to do all your work. Now a Web services project generated by the wizard includes a web.config file, a services.asmx file, and a host of others. More are generated as you build a project, including WSDL files. The good news is that you don't have to build most of the files yourself, the IDE does most of the work (see Figure 1).

Another pane in the IDE is the Server Explorer, which shows connections to various servers, such as IIS or SQL Server. We used the wizard to connect to the PUBS database, one of the sample databases that ship with SQL Server. It took about 10 seconds to create a connection, and using the design view of the IDE, another 10 seconds to drag the connection onto the design window and incorporate a database connection into our trial Web service application. Almost any component can be dropped onto the design view to incorporate it into the application, although a certain amount of coding is still needed to make things useful. For example, we dropped a DataSet and a DataGrid onto a WebForm, but still needed to write a few lines of code to connect the DataSet to the database connection we had previously set up, and a few more to tie the DataGrid to the DataSet.

Probably one of the more difficult aspects of the environment is knowing exactly what to do. The wizards and the various panes within the IDE provide a great deal of information, but knowing what to do will require serious study. Microsoft sent along several books, including *ASP .NET: Tips, Tutorials and Code*, from Sams, which was helpful in understanding how to begin. We also used the included Visual Studio .NET Walkthrough manual to take a tour of the Studio, but found this manual a little difficult to use as the transitions between various projects and files were not quite clear in some cases.

The learning curve will vary depending upon how familiar the user is with the actual interface and the languages themselves. We were familiar with both Visual Basic and Visual C++ and had no problem understanding the syntax of the .NET versions of the languages. Our familiarity with Java helped us find C# quite familiar from a language aspect, but we hadn't used the studio in several years and found the arrangement of tools and menus somewhat challenging while we tried to get up to speed. ***WSJ*** recommends training for users new to the studio to let them get up to speed quickly.

One of the most gratifying things about Visual Studio .NET is that you don't have to code your own client for the code you've developed. Many developers spend consid-

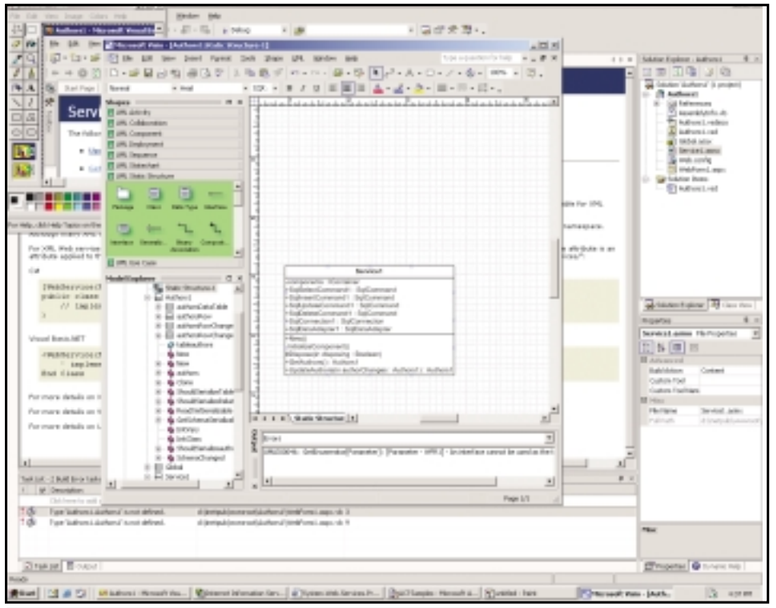


FIGURE 3 **Visio reverse engineering**

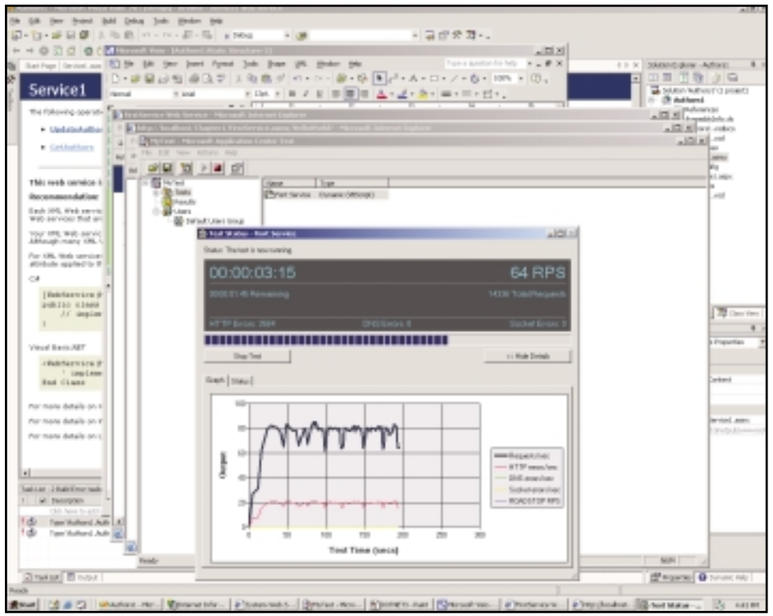


FIGURE 4 **ACT testing tool**

erable time developing their own clients just to test services. Visual Studio makes that all a thing of the past. Once the service is built (and automatically deployed to IIS), you can simply right click on the .asmx file and select "View in Browser," to see a complete client, generated in a browser page. In addition to supplying links to the various methods defined in the service, the page also provides sample SOAP code that can help you understand what's going on when you click the link and run the code. It's very satisfying to be able to test the code without having to write test code (see Figure 2).

Even more satisfying is the reverse-engineering feature built into Visual Studio courtesy of Visio. Most architects are familiar with Visio, having used it to draw network diagrams and the like for years. Microsoft has added a UML add-in module that allows for the creation of all the basic UML diagrams. While it won't give Rational Rose or Togethersoft a run for their money on total UML support, it's much more practical and hits the really meaningful diagrams very well. We didn't have enough time to test it, but Visio can also generate application code, in addition to doing reverse engineering, for a complete round-trip Modeling experience (see Figure 3). And Visio still retains all of its other features, which gives it a complete heads-up over other modeling tools in terms of formatting and printing.

We did take the time to create a simple test script using Application Center Test, the testing tool supplied with Visual Studio (see Figure 4). In our test we recorded a Web browsing session that accessed a Web service, and also were denied access to a page, generating a 404 error. Recording was very straightforward, with a simple interface similar to a tape recorder. We then ran the test, which ramped up the number of requests until it hit the limits of the machine (which didn't take much on our slow machine, but it was never intended to be a production server). While we were impressed with the ease of use, it did not seem that ACT is suitable for all testing needs, but it does provide a simple way to do repeatable unit testing.

Conclusion

Overall, we found Visual Studio .NET Enterprise Architect Edition full-featured, well-documented, and as complete a development environment as can be imagined from a single company. We highly recommend this as a suite of tools for development of Web services. The ease of use is remarkable, once the learning curve is overcome. ⓔ

Written by Michael Sick

NOTHING LOST in the Translation

ArtinSoft bridges gaps between languages, libraries, and frameworks

AUTHOR BIO:
Michael Sick is an independent Java architect who helps clients solve complex product definition and design problems. He has more than eight years of experience in the construction of distributed information systems and Internet technology, such positions as architect and VP of development.
MIKE_A_SICK@YAHOO.COM

Martin Fowler, of Extreme Programming fame, states: "any fool can write code that a computer can understand. Good programmers write code that humans can understand."

While true for every programmer in the business, this maxim also applies to code generators and translators. In this installment of "Web Services @ Work," we interview Federico Zoufaly, CTO and cofounder of ArtinSoft. His company is dedicated to creating code transformation tools that produce results humans can understand, rather than code that is suitable only for machines.

ArtinSoft's latest and most exciting products relate to the translation of Java to C#. To be honest, being a dedicated Java developer occasionally made it challenging to be a dispassionate and objective interviewer. While C# and the .NET Framework are becoming increasingly interesting to me, it's difficult not to have a knee-jerk reaction to things that advance a Microsoft agenda. But it's a brave new world of open standards and cross-language computing, and the best tool a developer can have is an open mind.

By the end of our conversation, I understood that ArtinSoft's products aren't about advancing one corporate agenda over another. Rather, they're about providing developers with choice in languages, libraries, and frameworks. That said, as a Java developer I'll be glad to see an ArtinSoft C# to Java translator when C# has the install base to justify it.

MIKE: Federico, please give us some background on ArtinSoft, how long it's been around, and what the company offers.

FEDERICO: We started ArtinSoft in 1993 and since day one we've been focusing our attention on source code–to–source code translation. We've been in the migration business for over nine years and the basis of our technologies is derived from research in artificial intelligence, specifically in theorem proving.

For instance, when you're proving a theorem in mathematics or in general logic, you have a set of statements that describe a certain reality. Then you start rewriting those statements, preserving equivalence each time in the right process until you reach some conclusion from the original statements. That same technique is what we apply to our source code–to–source code translation technology.

We take the original programming language or the original application, translate it to an intermediate format that we can manipulate, and then perform a lot of rewriting, preserving equivalence, until we reach the normal state of the program in its target language. We then print it with the correct syntax so that it's ready to be consumed by both the computer and the developer.

It's very important to distinguish our technology from normal compilation technology. A normal compiler goes from a certain level of abstraction down to a lower level of abstraction until it reaches the machine level. In our case, one of the main goals of our technology is to make sure that the generated code is understandable and maintainable by the original programmer. This means that we have to make sure the output is of high quality so the programmer can forget about the original source code and continue to evolve the application in the new platform. See Listing 1, a simple JSP to ASP.NET conversion.

MIKE: How did you come to start ArtinSoft and what is your background?

FEDERICO: I was one of the original founders of the company, so I've been working for ArtinSoft since '93. At the time, I was finishing my master's degree at the Costa Rican Technological Institute, and I joined Dr. Carlos Araya, who is the president and founder of the company. He is the one who had the original artificial intelligence ideas that we apply to our technology. So it was three other people and I that started ArtinSoft in '93. Shortly thereafter, another person joined us, providing the commercial background for the company, because the rest of the team was completely technical. Eventually, around '96, we opened the company to some local venture capital.

MIKE: Could you describe the business community in Costa Rica in terms of software development?

FEDERICO: Costa Rica is a small country with less than four million people. There are about 150 software companies right now. I think there are so many software companies in Costa Rica because in 1948 Costa Rica eliminated the army and decided to spend the entire budget on education. So the education level in Costa Rica is very, very good – computer science in particular. Most of the major universities have teachers with PhDs and master's degrees from U.S. and European universities. So the level of faculty education is also very high. At the same time, the business environment here has been good, and that is why software development has been a great success. In 1999 Intel made a big investment down here by building a large plant. They also added to the stability of the country and proved that Costa Rica is an excellent place to develop technology.

MIKE: What were the first practical applications of your conversion technology?

FEDERICO: Our first large, international account was doing research for Oracle, starting in '93. Then we moved on to the translation of Informix 4GL, where our first large customer was ICL of London. For ICL we translated a large Informix 4GL banking application to Java. At the time it was, in our estimation, the biggest Java application ever developed, having 1.8 million lines of Java code. It was a very good proof of concept for both Java and our technology.

MIKE: How did you establish a strong relationship with Microsoft?

FEDERICO: Microsoft was looking for a company that could help them bridge between Visual Basic 6 and Visual Basic .NET. As you may know, Visual Basic .NET is not backward-compatible with Visual Basic 6. Microsoft wanted to move Visual Basic into a first-class language on top of the .NET Framework. In order to add the features that they wanted, it wasn't possible to keep backward compatibility. However they didn't want to leave their install base stranded, so they looked for a company that could develop a migration engine for them. Eventually they chose ArtinSoft. We started to work with Microsoft at the end of '99, and we're still working with them. We have already

> It's a brave new world of open standards and cross-language computing, and the best tool a developer can have is an open mind"

delivered the Visual Basic .NET Upgrade Wizard, which has been integrated into Visual Studio .NET. We're also developing for Microsoft the Java-language Conversion Assistant, which is a Visual J++– to–C# translation tool. It will be included in the next revision of Visual Studio .NET. For now, the beta version is out and downloadable from the Microsoft Web site. The RTM version is going to probably happen later on this year.

MIKE: Now your latest product is the Microsoft Java-language Conversion Assistant. Could you give us a description of the feature set for that product, what someone could expect when he or she runs it, and what's going to come out the other side?

FEDERICO: We're going to take Visual J++ source code, which is the Java JDK 1.1.4 and is the last one [version] supported by Microsoft, and we're going to turn that into C#. Both the language and all of the object libraries are going to be mapped to C# and the object model for .NET. This we're building for Microsoft. On top of that, ArtinSoft is building the Java-language Conversion Assistant Enterprise Edition that will take all of J2EE and translate that into C# and the .NET Framework, including JSP to ASP.NET translations. So we're going to deliver that on top of the VJ technology for the J2EE platform.

MIKE: Will that be a product that any developer can simply download and run on their projects? How would someone begin to evaluate this product and see if it will work for his or her Java project?

FEDERICO: The Visual J++ product is now downloadable from the Microsoft Web site, and I think that even though it covers the Visual J++ object model instead of the full J2EE model, it's a good example of what the translation technology can do. From there you can start to evaluate our product. Eventually, we're going to license our JLCA Enterprise Edition as a product, so customers can license it, download it and then use it on their project.

MIKE: Do you feel like you get a good sense of every language that you end up either reading from or writing to?

FEDERICO: Yes. Definitely. We have engineers who are language experts in all the aspects of a language, from how to parse it, how to generate it, and what the transformational rules are that we have to apply in order to move from one to the other. We certainly have a lot of expertise on that, and we also use sample applications from real-world customers. We try to always collect a set of applications to also provide us with a feeling on how languages are used in the real world, and we focus on that. Our technology is based on pattern matching, so we try to find the program patterns that exist in real world applications, and that's what we translate in order to move the high-level abstraction from one platform to the other.

MIKE: Excellent. Some people on the IRC channel #java wanted me to ask you to compare and contrast Java and C#.

FEDERICO: Both Java and C# are derived from C++, so they have a common root. In that sense, they're similar enough. But you have to also understand that the language syntax itself is not as important to these languages as the object library that comes with them. So Java and C# are fairly simple languages. But what makes them very powerful is the object library that comes with them. You have all the Java SDK on one side and you have all the .NET CLR on the other side, and mapping those two is the real challenge in the translation business (see Table 1). Even though most of the functionality exists on both platforms, the way in which it is implemented is different. Our task is to match between the two platforms, which is not always an obvious task.

MIKE: How do your engineers translate higher-level concepts from Java to C# in a way that preserves these concepts and thoughts?

FEDERICO: They have to study both platforms and find how to accomplish a certain action in Java and how you would do the same thing in C#. It is not merely a one-to-one mapping, string class to string class. For very basic things, some mappings are very easy to find. For the most advanced features, you have to understand how the Java platform works and how the .NET platform works in order to find the right mapping. Remember, we are not building a compiler, so we want to make sure that the code we generate is understandable and is as close as possible to how it would have been implemented in the first place, by a native C# programmer.

TABLE 1: **Example Object Library Mappings**

Source Package	Target Packages
com.ms.lang	System Microsoft.Win32.RegistryKey
com.ms.wfc.util	System System.Diagnostics
	System.Collections
	System.Runtime.interopServices
	System.Resources System.Globalization
java.io	System.IO
java.lang	System System.Threading
java.util	System System.Collections
	System.Globalization System.Resources
	System.Configuration
com.ms.wfc.data	ADODB System.Runtime.InteropServices RDS
	System.Globalization System
	System.ComponentModel MSDASC
	System.Resources
java.net	System.Net System
	System.IO
java.sql	System.Data.OleDb System.DateTime

> ArtinSoft's products aren't about advancing one corporate agenda over another. Rather, they're about providing developers with choice in languages, libraries, and frameworks"

MIKE: What are the market forces behind ArtinSoft releasing the general Java to .NET Conversion Assistant?

FEDERICO: We think that both Java and .NET have a future in the IT community. We actually would say that in a few years from now, the market is going to be split among them. So there are always people who want to move from one to the other, merely because they want to get closer to the .NET platform or merely because a customer has a preference for Java instead of .NET or .NET instead of Java. For instance, a very good scenario is a software company, that builds applications. Perhaps they have their application written in Java right now, but they want the same functionality using a Microsoft platform. They might want that same functionality in .NET. I think .NET is a good alternative to Java. It's a platform that is definitely going to be successful.

MIKE: Now another question that people in #java came up with – are there plans for a .NET to Java converter?

FEDERICO: We at ArtinSoft try to pick our platform based on the market demand. Right now .NET is too new. There is not a lot of code written for the .NET platform yet.

WebServices JOURNAL
.NET J2EE XML

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

COO / CFO
Mark Harabedian mark@sys-con.com

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-Mussa megan@sys-con.com
ASSOCIATE SALES MANAGERS
Carrie Gebert carrie@sys-con.com
Alisa Catalano alisa@sys-con.com
Kristin Kuhnle kristin@sys-con.com
Leah Hittman leah@sys-con.com

SYS-CON EVENTS
VP, EVENTS
Cathy Walters cathyw@sys-con.com
REGIONAL SALES MANAGERS, EXHIBITS
Michael Pesick michael@sys-con.com
Richard Anderson richard@sys-con.com
CONFERENCE MANAGER
Michael Lynch mike@sys-con.com

CUSTOMER RELATIONS / JDJ STORE
MANAGER, CUSTOMER RELATIONS / JDJ STORE
Anthony D. Spitzer tony@sys-con.com
CUSTOMER SERVICE REPRESENTATIVE
Margie Downs margie@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
Catalin Stancescu catalin@sys-con.com
ONLINE EDITOR
Lin Goetz lin@sys-con.com

ACCOUNTING
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

SUBSCRIPTIONS
SUBSCRIBE@SYS-CON.COM
1-888-303-5282
FOR SUBSCRIPTIONS AND REQUESTS FOR BULK ORDERS, PLEASE SEND YOUR LETTERS TO SUBSCRIPTION DEPARTMENT
COVER PRICE: $6.99/ISSUE
DOMESTIC: $69.99/YR (12 ISSUES)
CANADA/MEXICO: $99.99/YR
ALL OTHER COUNTRIES: $129.99/YR
(U.S. BANKS OR MONEY ORDERS)

SYS-CON MEDIA

Therefore, writing a conversion tool for .NET to Java doesn't really make a lot of sense for us at this time. Eventually, if that becomes a market need, we will certainly consider it.

MIKE: Could you tell us a little more about the WinToWeb tool you launched at TechEd?

FEDERICO: It is a Windows Forms–to–Web Forms conversion assistant. So it takes your client/server Windows Forms applications and moves them to a thin-client approach, or ASP.NET. If you have Visual Basic 6 code, which normally is a client/ server or rich client, you will first move it to .NET using the Upgrade Wizard that is supplied by Microsoft. And then, once you're there, if you want to Web-enable your application, we provide the WinToWeb solution, the Windows Forms–to–Web Forms assistant that actually transforms it to ASP.NET. We launched the beta version at TechEd, and launched a release version early this summer.

MIKE: When will someone be able to download that product and work with it?

FEDERICO: The beta 1 version is available for download right now from our Web site – www.ArtinSoft.com. You can download it now for evaluation.

MIKE: Will the final conversion tool be downloadable?

FEDERICO: The final product is probably going to be downloadable – I don't know – [by] fourth quarter this year. Those are the dates we're looking at right now.

MIKE: Federico, any last thoughts?

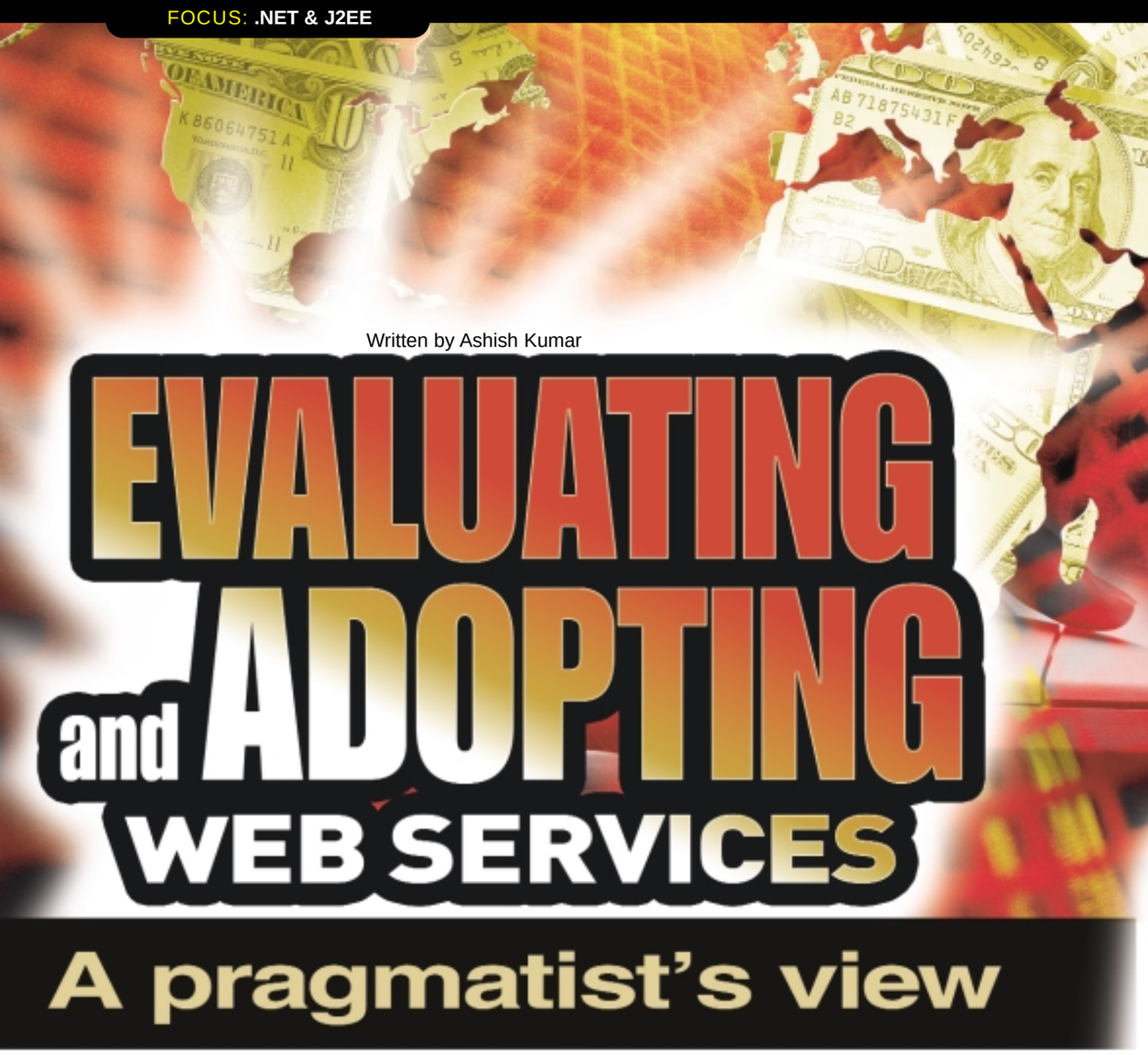
FEDERICO: I want to make a point that in the past, migrations have always been a very painful job for IT departments, but with ArtinSoft's technology, we are minimizing the work that has to be done in the migration process. We're providing companies with the freedom to choose where or which technology to use, instead of having to be locked in a particular platform.

MIKE: Thank you, Federico, and best of luck with your upcoming products.

Listing 1: Simple JSP to ASP.NET Conversion

```
JSP/Java Source Code
<%
//  set up  session attributes
session.setAttribute("key1", "value1");
session.setAttribute("key2", "value2");

// display session attributes
while (attributesEnum.hasMoreElements()){
     String attributeName =
      (String)
attributesEnum.nextElement();

     Object attributeValue =
      session.getAttribute(attributeName);
%>
     <%= attributeName %> : <%=
attributeValue%>
<%
}
%>

ASP.NET Migrated Code
<%
// set up session attributes
Session.Contents["key1"] = "value1";
Session.Contents["key2"] = "value2";

// display session attributes
System.Collections.IEnumerator
attributesEnum =
 Session.GetEnumerator();
while (attributesEnum.MoveNext())
{
         System.String attributeName =
          (System.String)
attributesEnum.Current;

         System.Object attributeValue =
          Session[attributeName];
%>

<%=attributeName%>:<%=attributeValue%>
<%
}
%> a
```

Download the code at
sys-con.com/webservices



Written by Ashish Kumar

EVALUATING and ADOPTING WEB SERVICES

A pragmatist's view

As the CTO of a technology integrator, I help customers derive tangible business value from technology solutions. Web services holds a lot of promise in this regard. After shedding some light on what's new about Web services, this article will present our view on the .NET versus J2EE debate and highlight areas where our customers are recognizing business value today with Web services technologies.

What's Really New Here?

Web services is a new approach to architecting and using software – the most recent in a long line of software paradigms that promised to make it easier to construct, connect, and reuse software systems. The predecessors to Web services included such technologies as structured programming, object-oriented programming, binary component standards like COM/DCOM and CORBA/IIOP, EAI middleware, and others. Each of these approaches tried to address some of the pains of software development, and each enjoyed varying degrees of success.

Web services builds on the technologies and learnings of its predecessors, adding the concept of software as a service, the use of standard technologies (XML, SOAP, and WSDL), and the use of the Internet as the delivery vehicle for software. These three factors are what's new, and their combination is what positions Web services to have a breakthrough impact on software development.

Focus on Business Capability

The service concept of Web services can potentially close the gap between business people and technology people. A software service that performs a well-defined business function is more easily understood by a business person than the mishmash of applications, databases, and Web sites that might be used to provide that function. Business people can talk about the cost, sourcing, and exploitation of Web services in terms that are meaningful to them.

Web services can be combined to implement complex business workflows and create business value chains and can also be discovered dynamically through UDDI registries. This aspect of Web services is probably the furthest out on the timeline, but it's tremendously powerful as a vision to build value chains based on changing business circumstances. For example, if a company's supplier were unable to provide parts required to fulfill a big order, the company could dynamically search for other suppliers to meet that need and electronically connect with them to complete the transaction.

Practical Approach to Reusing Software

Web services provides a practical approach to software reuse compared to the somewhat unwieldy component technologies of the past. Web services standards are broadly available on most computing platforms, which allows for maximum reuse of existing software investments since almost any legacy code can be wrapped as a Web service.

Web services can also be consumed by end users or other Web services to form a chain of services, and they can operate through firewalls, which makes it possible to implement business workflows that leverage existing code across organizational boundaries. And because Web services can be placed in UDDI directories, organizations will eventually have a broader base of candidates when making software "buy-versus-build" decisions.

Potentially Unlimited Reach

Through the use of a simple URL, you can invoke a Web service anywhere in the world. With the explosion of IP-enabled mobile devices, Web services will have the potential to reach billions of devices and enable an unlimited set of work and personal computing scenarios. This would not have been possible without the Internet and the use of standards like XML.

Still Version 1 Technology

With all its potential, Web services is still in the very early stages of technical maturity and customer adoption. Web services today can address many EAI scenarios but aren't yet ready to natively support real-time, reliable, high-security business transactions. Also, most customers aren't properly equipped with the IT capabilities to develop, deploy, and operate Web services. There are several standards organizations actively working on enhancing XML and SOAP specifications, and technology vendors have announced aggressive roadmaps to support Web services. The success of Web services will be based on the success of these efforts as well as the ability to maintain use of standard technologies while adding new capabilities. On the business side, Web services depends on the network effect – in other words, the value proposition increases geometrically based on the number of organizations that standardize on XML formats and deploy reusable Web services.

The .NET vs J2EE Debate

As organizations evaluate this new approach, the debate around choice of vendors, platforms, and tools is sure to arise. While Web services operate across multiple platforms and vendor products, individual project teams will tend to pick a preferred solution stack for their software development. Usually, this leads to the .NET versus J2EE debate, with Microsoft as the .NET provider and vendors such as IBM, Sun, Oracle, and BEA in the J2EE camp.

In order to implement Web services, organizations should consider the programming model, software product offerings, and preexisting Web services provided by the different vendors. The .NET and J2EE programming models generate a lot of discussion, but the real dollars are spent on software products that provide capabilities like databases, portals, e-commerce, caching, content management, etc. While most vendors have comprehensive software offerings, Avanade's experience has been that the .NET Enterprise Servers from Microsoft provide equivalent or better capabilities at a fraction of the cost of competing offerings. A number of independently conducted benchmarks demonstrate this as well (www.tpc.org). Microsoft also has existing services that can be leveraged as building blocks of Web services.

On the programming model topic, Microsoft's .NET offers more programming languages and has the best developer tools. It runs only on the Windows operating system and achieves scalability through multiple high-end hardware providers. J2EE offers a wide range of operating systems and hardware but is limited to Java, and the developer tools aren't as rich.

TABLE 1: **Implementing Web Services**

	Microsoft	IBM, Sun, Oracle
Programming Model	.NET	J2EE
Software Products	.NET Enterprise Servers	WebSphere, iPlanet, Oracle9*i*
Web Services	Passport, Messenger, MyServices, MapPoint	

AUTHOR BIO:

Ashish Kumar is the CTO of Avanade, Inc., a technology integrator focused on delivering solutions based on the Microsoft enterprise platform. Avanade's customers include Global 2000 companies and emerging market-makers seeking solutions for technology infrastructure and enterprise application development.
ASHISHK@AVANADE.COM

> The service concept of Web services can potentially close the gap between business people and technology people”

In our experience in examining the total solution stack, J2EE doesn't provide much portability over .NET. This is because different J2EE application servers have proprietary features that don't translate universally and aren't available across all the operating systems that support the Java language.

We've also found that the ASP.NET technology has a particularly strong value proposition for improving Web site performance and developer productivity. Some benchmark numbers comparing .NET with J2EE are available at www.gotdotnet.com/team/compare/veritest.aspx.

In summary, both J2EE and .NET can support implementation of Web services in the enterprise. .NET has a compelling value proposition, and organizations should consider it for new projects.

Where Are Customers Seeing Value?

We've helped many customers recognize value from Web services technologies. To date, this has been in the areas of implementing intracompany and intercompany workflows in an effective and efficient manner, managing cost of ownership of legacy systems, and reducing time-to-market for new business functions through use of a Web-services middleware layer.

i-Deal (www.i-deal.com) is a company that automates the process of raising capital in the global fixed income and equity markets. Their business platform is designed to increase the speed and efficiency of this complex, multiparty business process. Avanade worked with i-Deal to architect and implement their Web-based offering in less than six months. Using XML and .NET technologies, i-Deal was able to securely and reliably implement their service without needing to worry about the technical details of all the external systems they need to connect with.

We've also leveraged our application integration expertise and solution assets to successfully design, build, and deploy workflows that connect our customers' enterprise applications to other internal and external applications. The ability to deliver personalized and dynamic information from these workflows to front ends like enterprise portals and Microsoft Office has a significant benefit for employee productivity.

At the Australian government Department of Employment and Workplace Relations (DEWR), www.dewrsb.gov.au, we helped develop Web services that integrate and aggregate information about job seekers and job opportunities. This information resides in multiple mainframe applications and is made available through a Web service to authorized users of the department and to other public- and private-sector organizations. These services are registered in http://uddi.microsoft.com and allow users and applications to submit or query for job openings.

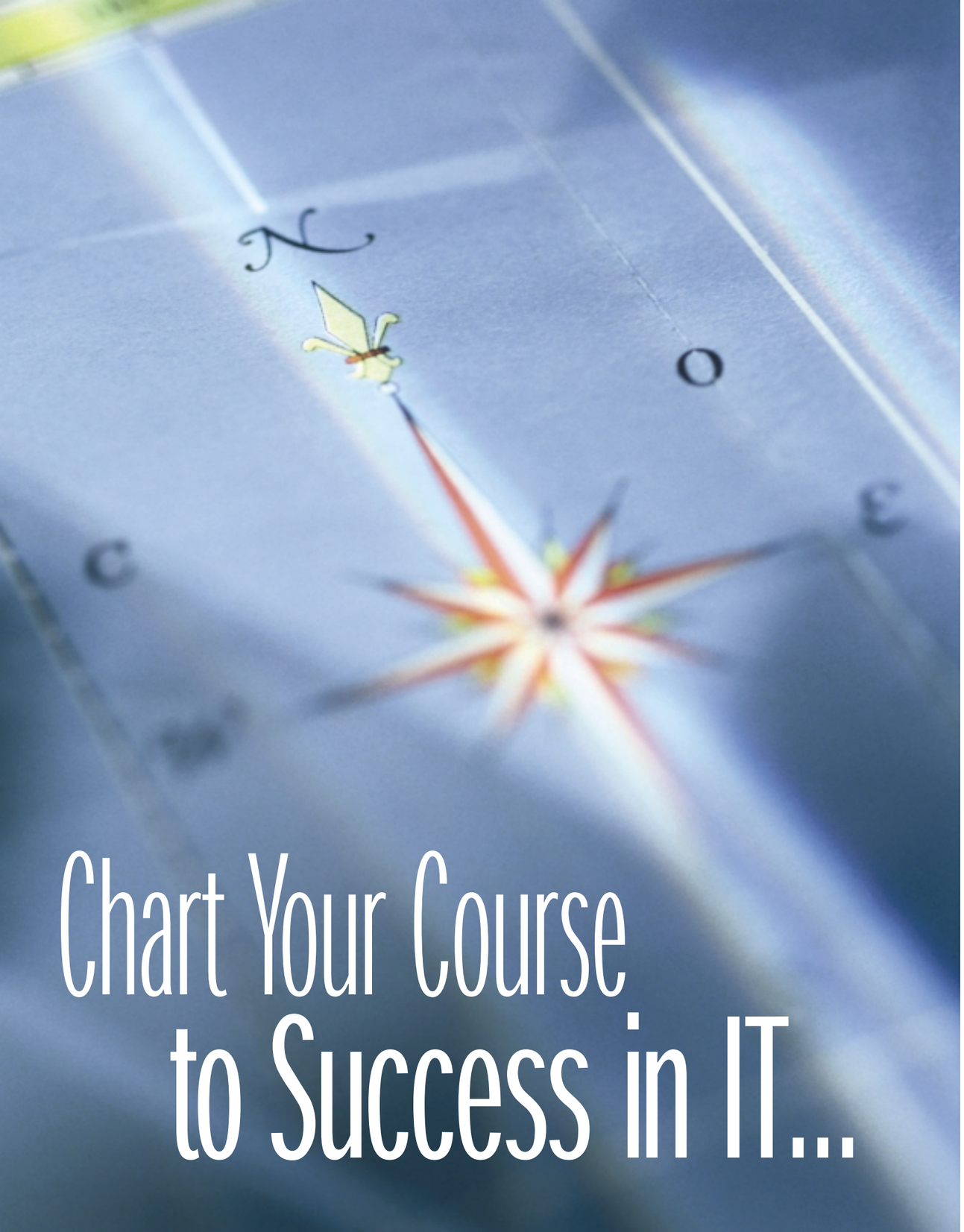
With a Web services abstraction layer, DEWR can now provision new business capabilities more quickly and economically than they would have been able to with just the mainframe environment. They can also reach a much broader audience through the use of UDDI-advertised Web services.

Conclusion

Web services provides a compelling value proposition to IT departments and has tremendous potential to become a technology with breakthrough business impact. Web services allows organizations to reassess their approach to software development and make strategic decisions on sourcing and partnerships. Web services products and proof points have started to emerge this year, but it will take a good 12–18 months before they prove themselves in the marketplace. Organizations should be aggressive in evaluating Web services solutions, but pragmatic in adoption to ensure positive ROI with every step. ©

TABLE 2: .NET vs J2EE

	.NET	J2EE
Languages	C#, VB .NET, C++, more	Java
Development Tools	Visual Studio .NET	WebLogic Workshop, WebSphere Studio
Middleware	.NET runtime, COM+ services	WebSphere, WebLogic, iPlanet
Operating System	Windows	Windows, Solaris, Linux, more
Hardware	Various – From 1 processor to 32 processors	Various – From 1 processor to mainframe

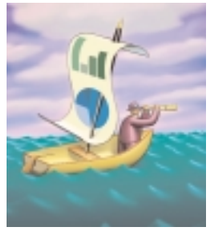


...order your copy of

Java Trends: 2003

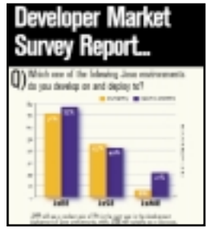
Available July 1*

Don't go astray. In the vast sea of Internet technology, market conditions change constantly. Will Java remain the hot platform it is today? Will C# rapidly gain ground? What are the world's foremost Java developers aiming their sights toward? Which companies come to mind when planning their next project? How can their thinking direct your efforts?



EnginData Research has studied the IT industry extensively, spotting many key trends and alerting industry leaders along the way. In the first quarter of 2002, they embarked on their most challenging mission ever: the most in-depth study ever done of the Java development marketplace.

After months of analyzing and cross-referencing more than 10,500 comprehensive questionnaires, the results are now available.



Here's just a sample of the critical data points the Java report delivers...

- ✔ Current and projected usage of languages and platforms
- ✔ Multiple rankings of hundreds of software vendors and tools
- ✔ Types of applications being developed
- ✔ Databases being used
- ✔ Purchasing patterns
- ✔ Company size
- ✔ Development and purchasing timelines
- ✔ Perceptions and usage of Web services
- ✔ Preferred Java IDE
- ✔ J2EE, J2ME, J2SE usage comparisons

As an IT specialist, **EnginData Research** focuses exclusively on three leading drivers in IT today – Java, Web services, and wireless development. Coming soon, our Web services survey will deliver such critical knowledge as:

- ✔ Time frame for developing Web services – enabled apps
- ✔ Percentage of apps with Web services today
- ✔ Sourcing and hosting Web services
- ✔ Perceived leaders in Web services tools and solutions

EnginData's research is invaluable to leading IT vendors, integrators, Fortune 500 companies, trade-show organizers, publishers, and e-business organizations worldwide.

Navigate your company through the challenging IT marketplace with the trusted knowledge and intelligence of **EnginData Research**. Order your copy of the 2002–2003 Java market study by calling Margie Downs at 201-802-3082, or visiting our Web site.

engindata RESEARCH

www.engindata.com

*A limited number of preview copies will be available at our booth (#624) at JDJEdge International Java Developer's Conference & Expo, June 24–27, at the Jacob Javits Convention Center in New York.

Written by Murali Janakiraman

Building Blocks Part II

An Overview of Web services technology

AUTHOR BIO:

Murali Janakiraman is Rogue Wave's software architect for the XML Products team. He has been a developer, senior developer, and tech lead on almost all of Rogue Wave's database products, including DBTools.h++, JDBTools, DBTools.h++ XA, and RWMetro.
MURALI@ROGUEWAVE.COM

This article continues from the *Web Services Journal* article "Building Blocks – An overview of Web services technology" (*WSJ* Vol. 2, issue 7). In Part I, I provided a functional classification of Web services technologies and examined technologies in the areas of service description, communication protocols, complex payloads, on-site inspection, and general discovery. In this installment, we examine technologies that address enterprise strength issues, such as transactions, security, and routing, and technologies that enable B2B collaboration.

Transactions

There are two kinds of transactions in the Web services space: background transactions and foreground transactions.

Background Transactions

Web services are typically front-end applications that deal with some back-end systems for persistent data storage and manipulation. Though a particular Web service may not support user-initiated transactions, simple invocation of the service may trigger transactional operations on the back-end systems that the Web service deals with. For example, a Web service performing updates on back-end systems may operate transactionally in order to guarantee ACID semantics. Transactions that take place without the involvement of the Web service user are called background transactions. In a background transaction, the Web service doesn't support the concept of transaction at its interface level, and hence doesn't allow itself to be invoked transactionally.

To implement a Web service that performs only background transactions, we don't need any new technologies. The transaction semantics provided by the back-end systems with which the Web service interacts provide the intended behavior. For example, if a Web service deals with a database to accomplish its job, it can rely on the transactional interface (such as XA or simple SQL commands like "begin transaction") provided by the database. Similarly, if a Web service is dealing with CORBA or EJB back ends, the respective transactional interfaces (such as OTS or JTS) provided by the back-end systems would suffice.

Note: In order to ensure background transactional capabilities, the back-end system must support transactions. It's possible to aggregate Web services in such a way that a Web service may become a back end for another Web service. To ensure background transactional characteristics, however, the Web service at the back end must support transaction at its interface level.

Foreground Transactions

Unlike background transactions, foreground transactions allow users to initiate and terminate transactions on the Web service, thus combining many operations (and possibly using many Web services) under one transaction.

Existing transaction technologies such as XA, OTS, and JTS cannot be directly applied to support foreground transactions on Web services. In the Web services world, transactions can span corporate boundaries with no single coordinating authority. Transactions can be long; the systems and resources involved are loosely coupled, and they work over networks prone to failure. Foreground transactions in Web services don't typically fit the traditional XA notion of a transaction.

New efforts are under way to arrive at standardized message formats and interaction models to support foreground transactions. In March 2001, OASIS formed a technical committee to work on an XML-based technology for transactions on the Internet. On May 1, 2002, that group released a 0.9.6.2 draft version of the specification called BTP.

BTP – Business Transaction Protocol

The Business Transaction Protocol (BTP) is designed to allow coordination of application work between multiple participants controlled by autonomous organizations. BTP's ability to coordinate between services offered by autonomous organizations makes it ideally suited for Web services. At its core, BTP is an interoperation protocol that defines roles played by participating software entities, the messages that pass between such entities, and the obligations and commitments of participating entities.

BTP is based on a concept of a business transaction that differs from traditional ACID transactions. Business transactions carry a more relaxed notion of a transaction than ACID transactions. For ACID transactions, there's only one successful outcome to a transaction – the transaction completely succeeds. In a business transaction, multiple successful

outcomes are possible. Business rules determine the outcome of a transaction. In a business transaction, participants can come and go and transactions can proceed to a commit, even though some participants may not favor the final outcome. In a business transaction, there is no central coordinating authority. Parties involved in a business transaction have distinct and autonomous application systems, and they communicate through well-defined service interfaces.

BTP provides a layered approach to meet the requirements of business transactions. It uses a two-phase coordination protocol to ensure the overall consistent state of the application. At its lowest levels, BTP supports two types of transactions: Atoms and Cohesions. An Atom is the standard atomic transaction that behaves like the existing XA-based, two-phase commit transactions. Within an Atom, the transaction either fully succeeds or rolls back. BTP demands that ACID properties be maintained within an Atom.

A Cohesion is a non-ACID transaction that can move to a successful outcome even if some participants don't favor the final outcome. A Cohesion typically consists of a set of Atoms manipulated by the business rules of the initiator of the transaction. It is run by a voting process in which the initiator holds the final approval or rejection vote. Many Atoms can be enlisted as part of the transaction, and Atoms are free to join or leave the transaction using an enrollment or resign process.

The transaction model of BTP identifies the different roles played by the participating entities and defines the relationships, commitments, obligations, and interaction model between the different roles. The specification defines the messages that flow between different roles at different stages of the transaction.

BTP messages are encoded in XML, allowing easy integration with XML-based Web services. BTP defines default binding to the SOAP and SOAP with Attachments specifications, further facilitating its use in Web services.

Security

As Web services becomes mainstream, a key issue is security. Ensuring integrity, confidentiality, message authentication, and security is critical to the acceptance and use of Web services.

The issue of online security isn't new to the industry. There are many existing security technologies, such as the public key infrastructure, Kerberos, X.509 certificates, and SSL. There are also many new security initiatives in the XML area, some of which specifically target Web services. This section focuses on these new XML-based security initiatives, which apply to XML-based messaging in general and to Web services in particular.

Some of the main XML-based security initiatives are:

- XML Encryption
- XML
- XKMS
- WS-Security
- SAML
- XACML

XML Encryption

The XML Encryption initiative is an effort by the W3C to specify a process for encrypting data and representing the result in XML. An XML document can be encrypted in its entirety using existing encryption technologies. However, there's no standard way to apply different encryption treatments to different parts of the document. The ability to encrypt individual XML elements or groups of elements enables authorized viewing of different pieces of the document by different parties.

The XML Encryption Specification achieves just that. Using XML Encryption, we can encrypt only the content of an element, or an entire element, or the entire document. In these cases, the encrypted portion is replaced with an EncryptedData element. An example of encrypting a part of a simple XML document is shown in Listing 1. Listing 2 shows the encryption of the entire CreditCard element (encrypted document). This example is taken from Section 2.1 of the XML Encryption Specification (www.w3.org/TR/xmlenc-core).

The raw encrypted data is contained in the CipherValue element. Unlike the example, if the encrypted data is referred through a URI, the reference is contained under a CipherReference element. XML Encryption allows encryption of EncryptedData elements. However, the EncryptedData element cannot be a parent or child of another EncryptedData element. Also important in XML Encryption is the EncryptedKey element, which is similar to EncryptedData, except that the data encrypted is always a key value. The schema for both the EncryptedData and the EncryptedKey element allow other optional information, such as encryption algorithm and key information.

The W3C published a Candidate Recommendation in March 2002.

XML Signature Specification

The XML Signature Specification was jointly developed by the IETF and W3C and published as a W3C Recommendation in February 2002. The XML Signature Specification defines XML syntax and processing rules for creating and representing digital signatures. As with XML Encryption, existing technologies can be successfully used for digitally signing an XML document as a whole. However, difficulty arises when parts of an XML document need to be signed by different parties. The XML Signature Specification enables digital signing of specific parts of an XML document.

The XML Signature Specification defines a schema for capturing the result of a digital signature operation applied to any arbitrary data. It can be applied to any digital content and is not limited to XML content. XML Signatures are represented by the Signature element. Listing 3 details the structure of the Signature element.

The SignedInfo element contains the information actually signed, while the SignatureValue carries the value of the encrypted digest of the SignedInfo element. The CanonicalizationMethod specifies the algorithm used to canonicalize the SignedInfo element before it's digested, and the SignatureMethod specifies the algorithm used to convert the canonicalized SignedInfo into SignatureValue. The Reference element carries the digest method and the digest value of the resource to be signed. The resource is identified by the URI attribute. Signature validation requires that the signed data object be accessible as a URI referenced from XML Signature or be embedded within the XML Signature. The KeyInfo element indicates the key to be used to validate the signature. The Transforms element lists the processing steps applied to the resource's content before it was digested.

The XML Signature Specification is an important security technology in XML with widespread adoption.

XML Key Management Specification

XKMS specifies protocols for distributing and registering public keys for use with XML Signature and XML Encryption. XKMS greatly simplifies the management of authentication and XML Signature key elements by creating a Trust Service based on the Web services model using SOAP 1.1 and WSDL. The client of the service can delegate part or all of the tasks required to process XML Signature KeyInfo elements. The service insulates user applications from the complexity and syntax of the underlying public key infrastructure (PKI), enabling the PKI to be based upon a different specification, such as X.509/PKIX, SPKI, or PGP.

XKMS is an important component in security as applied to XML documents. It has been published as a W3C Note, and W3C has started an XML Key Management working group.

WS-Security

While XML Encryption, XML Signature, and XKMS deal with security related to XML documents in general, WS-Security deals with encoding security information in Web services, specifically in SOAP messages. In a nutshell, WS-Security defines a standard set of SOAP headers that can be used to implement message integrity, message confidentiality, and propagation of security information in Web services. WS-Security itself doesn't define any new method for message integrity or confidentiality, but rather provides a standard way to encode this information in SOAP messages.

WS-Security achieves this by leveraging XML Signature and XML Encryption in conjunction with security tokens. A security token is defined as a representation of security-related information, such as a user name, an X.509 certificate, or a Kerberos ticket. WS-Security defines a general-purpose mechanism for associating these security tokens. In WS-Security encoding, all the security-related information is carried under a Security header element that can be targeted to any SOAP node using the SOAP actor attribute.

The WS-Security Specification was published jointly by IBM, Microsoft, and VeriSign in April 2002 and had at the time of this writing not been submitted to any standards body. IBM and Microsoft have also proposed other security initiatives. Details can be found at http://msdn.microsoft.com/ws-security.

Security Assertion Markup Language

SAML is an effort by the OASIS Security Services technical committee to develop an XML-based framework for exchanging security information. Though SAML supports other usage scenarios, its main goal is to enable sin-

gle sign-on. SAML envisions a scenario where users can log on to a site, submit their credentials to the SAML security engine on that site, get security information as a response, do their work, and move on to another site with the SAML security information traveling along with them, avoiding another logon session at the new site.

Security information is expressed in the form of assertions about subjects, where a subject is an entity with an identity (either human or computer). Assertions convey information about attributes of subjects, authentication acts performed by subjects, and authorization decisions about subjects' access rights. Assertions are issued by SAML authorities (any SAML-aware security engines).

SAML defines a schema for the assertions and a request-response protocol for the client-SAML authority communication. Though this protocol can be bound to many underlying communication and transport protocols, SAML defines a default binding to SOAP over HTTP. As part of this binding, SAML defines how assertions can be carried in SOAP messages.

SAML 1.0 was published in April 2002 but is yet to be approved as a final specification. SAML relies on XML Signature for the signing of request-response messages. While there is no conflict of interest between WS-Security and SAML, it appears that some coordination is possible between the SOAP binding of SAML and WS-Security SOAP headers.

Extensible Access Control Markup Language

XACML is closely related to SAML but focuses more on access control and rights management by providing a representation for expressing policies and rules for information access. Specifically, XACML is expected to define XML representations for rules that specify who can exercise which access privileges for a particular XML document.

XACML is also from OASIS, which is expected to coordinate closely with SAML and to adopt SAML work as baseline documents. By May 2002, this group had published a 0.13 version of the proposal.

With all these new initiatives, good progress is being made in the area of Web services security, although there's still a long way to go to provide convenience of use, reliability, and robustness.

Routing

In the Web services model, communication protocols such as SOAP depend on binding to underlying transport protocols to actually exchange a message. These transport protocols define their own message path models and message exchange patterns, which differ from the communication protocols message model. This section looks at technologies that define message routing in communication protocols.

Microsoft announced a SOAP-based routing protocol called WS-Routing in October 2001.

WS-Routing

Although SOAP supports the notion of intermediaries and provides some ability to address parts of a message to intermediaries, it doesn't define a mechanism for sending a message from one node to another. WS-Routing is a simple, stateless protocol that enables the routing of it messages in an asynchronous manner over a variety of transports, such as TCP, UDP, and HTTP. WS-Routing enables the entire message path to be described directly within the SOAP Envelope. To enable such routing, WS-Routing defines a new SOAP header named Path containing four new elements: from, to, fwd, and rev.

The from element indicates the originator, the to element indicates the ultimate receiver, the fwd element contains the forward message path, and the rev element contains the reverse message path. The reverse message path is optional and exists to support two-way as well as one-way messages. WS-Routing doesn't expect the entire message path to be determined in advance; with some restrictions, the intermediaries can dynamically insert additional intermediaries into the message path.

It's important to note that WS-Routing doesn't define the means for reliable or secure messaging. However, WS-Routing may possibly be combined with WS-Security and new SOAP-based reliability protocols to provide reliable and secure messaging.

Microsoft published the WS-Routing specification in October 2001, but it hasn't yet been submitted to any standards body. Microsoft has also released a closely related protocol called WS-Referral, which allows SOAP nodes in a message path to be dynamically configured to delegate part or all of their processing responsibilities to other SOAP nodes. Discussing WS-Referral goes beyond the scope of this article.

B2B Collaboration

In Part I, we discussed service description standards as part of the discovery stack of Web services technologies. At the higher end of the discovery stack are the standards that help model business processes to enable cross-enterprise business-process automation for higher levels of automated B2B collaboration. Automating business processes across corporate boundaries is key to truly realizing the potential and promise of e-commerce.

Complete Protocols

So far, we've looked at technologies that address discrete functional areas of Web services. There are also some efforts that provide several of these technologies under one umbrella. One of the well-known efforts in this area is ebXML.

EbXML

EbXML is the result of a joint effort by

UN/CEFACT and OASIS to develop an XML-based, modular, end-to-end solution for enabling secure, global, electronic interoperability using open standards and replacing EDI (Electronic Data Exchange). EbXML is based on three main goals:

1. To provide an infrastructure for data communication
2. To provide a framework for commercial interoperability
3. To provide a mechanism to discover, agree, and conduct business

Based on these goals, the group published a set of three core specifications in May 2001. They are:

1. The Messaging Service Specification
2. The Registry and Repository Specification
3. The CPPA Specification

These specifications don't depend on one another and can be used separately.

Although it's been a year since the release of the ebXML specifications, there are few implementations that support it. Most of the focus is on WSDL, SOAP, UDDI, and other developing standards. Note that ebXML is still an ongoing effort and that different groups are in the process of releasing newer versions.

Conclusion

This two-part article has provided a high-level overview of the technologies that address the different functional areas of Web services. So far, preliminary development in these areas has come from individual companies, groups of companies, and in some cases standards organizations. Interoperable standards are a must for Web services to enable cross-enterprise communication and loosely coupled systems. Consortiums and standards bodies are getting more and more involved in adapting and building on these early developments. These efforts should help move Web services from the current era of trial applications into mainstream use.

Resources

- BTP: www.oasis-open.org/committees/business-transactions/documents/2002-05-01.BTPdraft 0.9.6.pdf
- XML Encryption: www.w3.org/TR/xmlenc-core/
- XMLSignature:www.w3.org/TR/xmldsig-core/
- XKMS: www.w3.org/TR/xkms/
- WS-Security: http://msdn.microsoft.com/ws/2002/04/Security/
- SAML: www.oasis-open.org/committees/security/
- XACML: www.oasis-open.org/committees/xacml/
- WS-Routing: http://msdn.microsoft.com/ws/2001/10/Routing/
- WS-Referral: http://msdn.microsoft.com/ws/2001/10/Referral/
- WSFL: www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf
- XLANG: www.gotdotnet.com/team/xmlwsspecs/xlang-c/default.htm#Toc515125575
- BPML: www.bpmi.org/bpml.esp
- EbXML-CPPA and CPPA Specification: www.oasis-open.org/committees/ebxml-cppa/
- UBL: www.oasis-open.org/committees/ubl/
- EbXML: www.ebxml.org
- Messaging Service Specification: www.oasis-open.org/committees/ebxml-msg/
- Registry and Repository Specification: www.oasis-open.org/committees/regrep/

Listing 1: Encrypting part of an XML document

```
<?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <Number>4019 2445 0277 5567</Number>
    <Issuer>Example Bank</Issuer>
    <Expiration>04/02</Expiration>
  </CreditCard>
</PaymentInfo>
```

Listing 2: Encryption of the entire CreditCard element

```
 <?xml version='1.0'?>
<PaymentInfo xmlns='http://example.org/paymentv2'>
  <Name>John Smith</Name>
  <EncryptedData
Type='http://www.w3.org/2001/04/xmlenc#Element'
                xmlns='http://www.w3.org/2001/04/xmlenc#'>
    <CipherData>
      <CipherValue>A23B45C56</CipherValue>
    </CipherData>
  </EncryptedData>
</PaymentInfo>
```

Listing 3: XML digital signature structure

```
<Signature ID?>
  <SignedInfo>
    <CanonicalizationMethod/>
    <SignatureMethod/>
     (<Reference URI? >
       (<Transforms>)?
       <DigestMethod>
         <DigestValue>
    </Reference>)+
  </SignedInfo>
  <SignatureValue/>
  (<KeyInfo>)?
  (<Object ID?>)*
</Signature>
```

Note: ? denotes zero or one occurrence, + denotes one or more occurrence, and * denotes zero or more occurrences.

Download the code at sys-con.com/webservices

WSJ News

Mongoose Technology Launches PortalStudio 3.0 in Latin America

(Mexico City) – Mongoose Technology, Inc., a leading U.S. enterprise portal solution vendor, has launched Mongoose PortalStudio 3.0 in Latin America as leading systems integrators presented advanced online solutions for doing business in the real-time economy. Over 150 business and IT managers were on hand as systems integrator partners including Gedetec, Grey Interactive Mexico, Hildebrando, Meta Data, OmniSys, and Teysa demonstrated real-time business solutions built with Mongoose PortalStudio. Mongoose PortalStudio 3.0 adds new application integration and Web services capabilities to enhance integration and access to critical IT resources.
www.mongoosetech.com

W3C Publishes First Public Working Draft of WSDL 1.2

(Amsterdam) – The World Wide Web Consortium (W3C) has issued Web Services Description Language (WSDL) 1.2 and WSDL 1.2 Bindings as W3C Public Working Drafts. WSDL 1.2 is an XML-based language that describes a Web service – the data exchanged, the protocol to use, and its location on the Web. WSDL 1.2 Bindings describes how to use WSDL 1.2 with SOAP 1.2, HTTP, and MIME. These are the first in a series of WSDL 1.2 drafts; W3C invites the Web development community to review and comment on these and subsequent versions.

The W3C has been developing an architecture for Web services which takes into account the needs of users as well as technology vendors pursuing the development of open standardized components for Web services, as it has done for HTML, the XML family of technologies, scalable vector graphics, and Voice-Web technologies.
www.w3.org

Connotate's Instant Web Services 1.0 Transforms Internet/Intranet Sites

(New Brunswick, NJ) – Connotate Technologies, Inc., has announced the release of Instant Web Services 1.0. Used in conjunction with Connotate's Web Mining Server, Instant Web Services 1.0 provides a mechanism for sharing essential content with multiple groups such as employees, customers, and supply-chain partners.

To create a Web service, users select from a library of Information Agents – definable software objects that know how to monitor, extract, filter, and deliver information. Once a user has selected an agent, the system prompts for configuration information, and proceeds to create a Web service complete with WSDL. Optionally, the Web service can be published through UDDI.
www.connotate.com

EnvoyWorldWide Takes Web Services from Concept to Reality

(Bedford, MA) – EnvoyWorldWide, a provider of real-time notification services to wired and wireless devices, has announced support for Web services with the availability of a SOAP-enabled API.

EnvoyWorldWide's SOAP- and XML-based Web services are designed to enable organizations to integrate the functionality of EnvoyWorldWide's real-time message delivery engine into their own applications or between applications, shortening development time, leveraging existing technology to trigger events, and streamlining the communications process.
www.envoyww.com

Stoneware Web Portal V3.5 Supports Microsoft's Active Directory

(Indianapolis) – Stoneware, Inc., a corporate portal software maker, has released Stoneware 3.5., "an affordable out-of-the-box" portal with more services and features than other portal solutions on the market.

This release offers several new features that expand Stoneware's traditional offering, including data exchange service; WAP; port manager; support for Microsoft Active Directory, in addition to Novell NDS and eDirectory; Web administration; and full support of industry standards.
www.stone-ware.com

Epicor Announces Microsoft .NET XML Web Services for 'e by Epicor'

(Irvine, CA) – Epicor Software, a provider of integrated enterprise, e-business, and collaborative commerce software solutions, has announced the immediate availability of Epicor Web Services for its integrated enterprise suite, e by Epicor. Built exclusively with Visual Studio .NET and the Microsoft .NET Framework, Epicor Web Services is a set of software components that makes it easier to integrate and securely share information across systems, both within the enterprise and with customers and suppliers.

Epicor's native Microsoft .NET applications support the Web services "software as a service" delivery model and empower midmarket companies to achieve rapid results from their enterprise applications at a lower total cost of ownership. Epicor Web Services will allow e by Epicor customers to streamline business operations and interactions with customers, suppliers, partners, and employees through easy-to-deploy Web applications that give businesses flexibility, control, and interenterprise integration.
http://e.epicor.com

Microsoft Announces Content Management Server 2002 Beta

(Redmond, WA) – Microsoft Corp. has announced the availability of the beta version of Microsoft Content Management Server 2002. The new release will provide customers with a faster, easier, and more cost-effective way to create, deploy, and maintain mission-critical, content-rich Web sites. Part of the Microsoft .NET Enterprise Server family, Content Management Server 2002 offers support for XML Web services, enhances content authoring and site deployment capabilities, and provides developer support through its integration with Visual Studio .NET and support for ASP.NET.

The beta version of Content Management Server 2002 is available now for download at www.microsoft.com/cmserver. It is scheduled for release by the end of this year.
www.microsoft.com

WSJ ADVERTISER INDEX

ADVERTISER	URL	PHONE	PAGE
Actional	www.actional.com/whitepapers/wsjblues	800-808-2271	9
Altaworks	www.altaworks.com	603-598-2582	3
Altova	www.xmlspy.com		15
BEA	www.bea.com		68
BEA WebLogic Developer's Journal	www.weblogicdevelopersjournal.com	888-303-5282	47
ColdFusion Developer's Journal	www.sys-con.com/coldfusion	888-303-5282	52
engindata Research	www.engindata.com	201-802-3082	48, 49
IBM	www.ibm.com		2
JDJ Store	www.jdjstore.com	800-303-JAVA	41, 60, 61
Mongoose Technology	www.portalstudio.com		67
Muscular Dystrophy Association	www.als.mdausa.org	800-572-1717	33
New Atlanta	www.newatlanta.com		23
PolarLake	www.polarlake.com		19
Power Builder Developer's Journal	www.sys-con.com/pbdj	800-303-5282	54
Sitraka	www.sitraka.com/jclass/ws		13
Sonic Software	www.sonicsoftware.com/websj		4
SYS-CON Events	www.sys-con.com	201-802-3069	43
SYS-CON Media	www.sys-con.com	888-303-5282	47, 55
Web Services Edge Conference & Expo	www.sys-con.com	201-802-3069	65
Web Services Journal	www.sys-con.com	888-303-5282	37
webMethods	www.webmethods.com/ews		29
WebSphere Developer's Journal	www.sys-con.com	888-303-5282	32

Advertiser is fully responsible for all financial liability and terms of the contract executed by their agents or agencies who are acting on behalf of the advertiser.

SYS-CON MEDIA SYS-CON EVENTS

Welcome to Web Services Edge International Web Services Conference & Expo

June 24-27, 2002 • Jacob Javits Center, New York City

web services EDGE conference&expo

2002 Show Report

A resounding success for the largest Web services event in the world!

Web Services Journal was in New York City the week of June 24 to cover the second annual Web Services Edge East International Conference & Expo. After its very successful debut last year, Web Services Edge doubled in size, becoming the educational arm not only for this show, but also for the prestigious PC Expo conference, held at the same time and place.

Web Services Edge Draws Record Crowds to Jacob Javits Convention Center, NYC

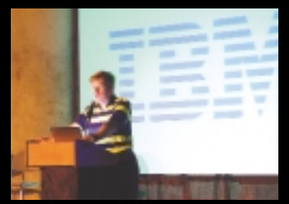
WSJ editor-in-chief Sean Rhody served as tech chairman for the conference, and opened the show with a few choice remarks. "Last year we had just experienced the biggest tragedy in U.S. history. Web services were nascent, but no one was sure how quickly they would go mainstream, and what the impact of September 11th would be on our economy and the landscape of information technology. We needn't have worried. This is the year of Web services," said Rhody. "The question on everyone's mind this year is not whether to adopt Web services, but when."

Web services has become the umbrella term for the huge new shift in the world of e-business toward distributed computing on both the Internet and corporate intranets. Companies in all lines of business are adopting Web services, seeing in them the capability to do application integration with an open standards approach that doesn't lock them into a single proprietary solution. Not just within the corporate firewall, entire supply chains are looking at Web services as a way to standardize enterprise-to-enterprise communications throughout their particular industry.

Exhibitors spend time with qualified attendees

Crowds enjoy exhibitor product demonstrations

Delegates listen intently to class on IT Strategies



Don Ferguson, IBM Fellow and "father of WebSphere," delivers opening keynote

Supercharging Java with Web Services, discussed by Simon Phipps, Sun Microsystems; Tel Shelton, Borland; Rick Ross, JavaLobby; and Ajit Sagar, Contributing Editor, *JDJ*, moderated by Alan Williamson, editor-in-chief, *JDJ*

Peter Hoversten, Sybase, discusses Web Services: A Pragmatic Approach

Keynote panels were a hit with an interactive audience

World-Renowned Speakers

The conference featured four world-renowned keynote speakers and five panel discussions. Don Ferguson (the "father of WebSphere") of IBM led the first keynote with a riveting discussion of Web services technologies and their applications, digging deep into the paradigms of Web services and their applicability to today's distributed computing problems. Peter Hoversten of Sybase spoke on integrating applications via Web services. Phillip Merrick, CEO of webMethods, gave an insightful discourse on enterprise Web services as the next level of integration; and Eric Newcomer of IONA offered a frank review of Web services integration.

The panel discussions were also very intriguing. Industry leaders, pundits, and subject-matter experts gathered to discuss a variety of topics. Celebrity guests included Patricia Seybold, CEO of Patricia Seybold Group; Jean-Francois Abramatic, former head of the W3C; Patrick Gannon, CEO of the OASIS group; and Sue Aldrich, also of Patricia Seybold Group. The discussions were lively and included topics such as Web services integration, supercharging Java with Web services, and the ROI case for Web services. The final panel discussion featured *Web Services Journal* editor Sean Rhody, ***Java Developer's Journal*** editor Alan Williamson, and representatives from Sun and Microsoft in a lively discussion of J2EE vs .NET, moderated by ***XML-Journal*** coeditor JP Morgenthal.

"Industry leaders, pundits, and subject matter experts gathered to discuss a variety of topics"

Phillip Merrick, webMethods, discussed Web Services: The Next Level of Integration

Eric Newcomer, IONA, explored Web Services: Integration Technology for the 21st Century

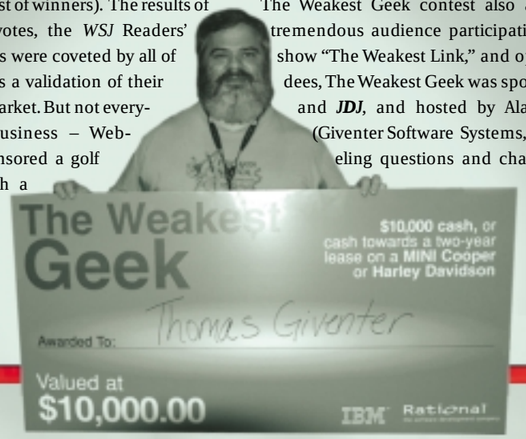
Dave Chappell, Sonic Software; Sue Alldrich, Patricia Seybold Group; Peter Hoversten, Sybase; Jean-Francois Abramatic, ILOG; and Jeremy Allaire, Macromedia, gave their perspective on Integrating Applications with Web Services.

Conference attendees requested longer sessions to learn more

Special Events

The show floor was a buzz of activity, with demonstrations running constantly. ***WSJ*** presented its first annual Readers Choice Awards (see page 66 for a list of winners). The results of over 11,000 votes, the *WSJ* Readers' Choice Awards were coveted by all of the vendors as a validation of their place in the market. But not everything was business – Web-Methods sponsored a golf challenge with a Mercedes Benz as the reward for a hole-in-one. Unfortunately for the attendees, no one had the skills of Tiger Woods and the prize went unclaimed. Better luck next time.

The Weakest Geek contest also attracted a lot of attention, and tremendous audience participation. Modeled after the television show "The Weakest Link," and open to all of the conference attendees, The Weakest Geek was sponsored by Rational Software, IBM, and ***JDJ***, and hosted by Alan Williamson. Thomas Giventer (Giventer Software Systems, Ithaca, NY) survived all of the grueling questions and challenges and received a check for $10,000 from Rational Software and IBM. Winners of the preliminary rounds took home copies of Rational XDE and WebSphere Studio, valued at more than $7,000.



2002 Show Report

"Looking to the future, make sure not to miss our other upcoming events devoted to Web services, Java, XML, and .NET"

Packed Classes

But as is always the case with shows such as these, the real focus, the real area of attention, was the courses. Web Services Edge featured five concurrent tracks – Java, Web Services, .NET, XML, and IT Management. Attendance at these sessions was incredible, with standing-room only at a number of classes. And the reaction to the material was uniformly positive. "This will help me deploy Web services today, rather than a year from now," said one satisfied attendee.

And help it did. Featured classes included such topics as Java Advanced Security Concepts, Web Services Fundamentals, .NET Essentials, and Key Trends and Technologies for Building Web Services. Each track featured 12 value-packed sessions that covered the attendees' needs, regardless of whether they were a Java developer, a Microsoft developer, an architect, or the manager of a development group.

The show also featured intense, hands-on tutorials from Sun on Java. There were three paths on Monday; Java 2 Platform Fast Path for the developer; Java 2 Platform Architecture Fast Path, which leads to an architectural perpspective on J2EE; and finally the Web Component Developer Fast Path, focusing on the development of Web components and services.

Overall, the show was a rousing success, with more than 8,000 attendees visiting the show floor during the three-day exhibition. The combination pass that allowed attendees to also visit PC Expo and DV Expo was especially well received. These two shows provided an even greater insight into the industry, merging the corporate and independent developer perspectives with the consumer perspective.

Plan Ahead

Looking to the future, make sure not to miss our other upcoming events devoted to Web services, Java, XML, and .NET. Join us at Web Services Edge West International Conference & Expo, October 1–3, 2000, in San Jose, California, and next year's East Coast show, scheduled for March 25–28, 2003, in Boston.

Web Services Journal Announces First Annual Readers' Choice Awards Winners

On June 25, 2002, ***Web Services Journal*** (www.sys-con.com/webservices), the world'*s first and leading Web services magazine, announced the results of its first annual Readers' Choice Awards. The recipients were selected through reader-submitted nominations in 17 categories, followed by online voting by more than 11,000 participants. An independent research firm managed the voting process.

The winners were:

Best Application Server for Web Services ★★

Winner: BEA WebLogic Server 6.1 (BEA Systems)
1st Runner Up: IBM WebSphere Application Server 4.1 (IBM)
2nd Runner Up: Oracle9i Application Server (Oracle)

Best Web Services Automation Tool ★★

Winner: BEA WebLogic Server 6.1 (BEA Systems)
1st Runner Up: IBM WebSphere Studio (IBM)
2nd Runner Up: Oracle9i Application Server (Oracle)

Best Web Services Book ★★

Winner: Java Web Services (O'Reilly & Associates)
1st Runner Up: Programming Web Services with SOAP (O'Reilly & Associates)
2nd Runner Up: Building Web Services with Java (SAMS Publishing)

Best Web Services Class Library ★★

Winner: CLX: Component Library for Cross-Platform (Borland)
1st Runner Up: Web Services Invocation Framework (IBM)
2nd Runner Up: SilverStream eXtend Director 3.0 (SilverStream Software)

Best Web Services Foundation Platform ★★

Winner: SonicXQ 1.0 (Sonic Software)
1st Runner Up: IBM WebSphere Application Server 4.1 (IBM)
2nd Runner Up: BEA WebLogic Server 6.1 (BEA Systems)

Best Framework for Web Services ★★

Winner: IBM WebSphere (IBM)
1st Runner Up: BEA WebLogic Server 6.1 (BEA Systems)
2nd Runner Up: Oracle9i JDeveloper (Oracle)

Best Web Services IDE ★★

Winner: VisualAge for Java - WebSphere Studio Application Developer (IBM)
1st Runner Up: Oracle9i JDeveloper (Oracle)
2nd Runner Up: Delphi 6 (Borland)

Best Integrated Services Environment ★★

Winner: Delphi 6 (Borland)
1st Runner Up: IBM WebSphere Studio (IBM)
2nd Runner Up: XML Spy Suite 4.2 (Altova)

Best Web Services Integration Tool ★★

Winner: Oracle9i Application Server (Oracle)
1st Runner Up: BEA WebLogic Integration (BEA Systems)
2nd Runner Up: IBM MQ Series Integrator (IBM)

Best Web Services Legacy Adapter ★★

Winner: BEA WebLogic Integration (BEA Systems)
1st Runner Up: CICS Gateway (IBM)
2nd Runner Up: Oracle9i Application Server (Oracle)

Best Mass Market Web Service ★★

Winner: IBM UDDI Business Registry (IBM)
1st Runner Up: dataLive (LongReach Software)
2nd Runner Up: N-central IT Management Solution (N-able Technologies)

Best Web Services Middleware ★★

Winner: SonicXQ 1.0 (Sonic Software)
1st Runner Up: IBM WebSphere (IBM)
2nd Runner Up: BEA WebLogic Server 6.1 (BEA Systems)

Best Web Services Platform ★★

Winner: BEA WebLogic Server 6.1 (BEA Systems)
1st Runner Up: IBM WebSphere Application Server 4.1 (IBM)
2nd Runner Up: Oracle9iAS and Oracle9i JDeveloper (Oracle)

Best Web Service Site ★★

Winner: IBM developerWorks (IBM)
1st Runner Up: XMethods (XMethods)
2nd Runner Up: IBM Web Services Technology Newsletter (IBM)

Best Web Services Testing Tool ★★

Winner: Oracle9i JDeveloper (Oracle)
1st Runner Up: IBM alphaWorks Web Services Testing Area (IBM)
2nd Runner Up: XML Spy Suite 4.2 (Altova)

Best Web Services Training Tool or Program ★★

Winner: XML Spy Suite 4.2 (Altova)
1st Runner Up: IBM Web Services Toolkit (IBM)
2nd Runner Up: RADical Web Services for e-Business (Borland)

Best Web Services Utility ★★

Winner: Oracle9i JDeveloper (Oracle)
1st Runner Up: IBM Web Services Toolkit (IBM)
2nd Runner Up: XML Spy Suite 4.2 (Altova)