Web Services: Building the Next Generation of E-Business Applications

Christophe Coenraets
JRun Product Marketing Manager

October, 2nd 2001

Table of Contents

Executive Summary 3
Current Web Model 4
- Isolated Information 4
- Changing Business Relationships 5
- Accelerated Pace of Change 5

Web Services: A New Web Model 6
- What are Web Services? 6
- Service-Oriented Architecture 7
- Hype and Reality 7
- Example 8
- SOAP 9
- WSDL 10
- UDDI 10

Macromedia Involvement in the Web Service Community 11
- Web Services Support in JRun 11
- Other Web Services Initiatives at Macromedia 12

Executive Summary

Since the inception of the Internet, no other technology has been adopted as quickly as Web services. Web services proponents are a fast-growing community that includes J2EE vendors, Microsoft, the open source and the standards communities. Already, people in the industry use the same terms and fundamentally agree that Web services are an important new paradigm that sets the standard for Web application development. That is an incredible testament to the potential Web services have for developers in the future.

Web services represent the next generation of e-business applications. Their architecture removes the stumbling blocks associated with traditional e-business. The key benefit of Web services is that they allow previously incompatible applications to interoperate on the Web regardless of language, platform, and operating systems. Because they dramatically facilitate the integration of disparate applications, Web services create new e-business opportunities, making it easy for companies to adapt to changing business relationships and develop new e-partnerships.

Macromedia has been involved in Web services before the term was coined and continues to provide industry leadership in the development and implementation of the Web services standards.

Macromedia participates in W3C working groups and, as part of the Java Community Process (JCP), is active in many Java expert groups related to XML and Web services.

Macromedia has also taken a leadership role in open source efforts related to Web services. The company has dedicated full time engineers to work on the Axis Web services engine, a project of the Apache Software Foundation.

Macromedia engineers leverage this experience to provide JRun with a powerful Web services engine that supports SOAP 1.1, WSDL 1.0, and additional productivity features. Building a Web service in JRun is as simple as building a regular Java class or an EJB and then exposing it as a Web service. Existing Java classes and EJBs can instantly be turned into Web services.

A preview of this technology can be installed today on JRun 3.1. The download is available at: www.macromedia.com.

Current Web Model

Until now, the Web has mainly been used for *business-to-consumer* applications. Individuals use a browser to navigate through linked documents, download files, or manually initiate purchases and simple transactions.

Today, with virtually every computer and potentially every application connected to one single network, there are increased opportunities for a new breed of *business-to-business* applications. The fact that all these connected applications have the potential to communicate with each other and facilitate business over the Internet is extremely appealing.

However, the industry has been slow to develop and integrate these applications. Many stumbling blocks prevent applications from interoperating in a way that establishes a network of connected applications.

These stumbling blocks include:

- The isolation of business processes within incompatible systems
- The fact that business relationships often change
- The accelerated pace of change in technology and standards

Isolated Information

Over the years, companies have developed and acquired applications that have been written in a variety of languages. These applications run on different operating systems and on different hardware platforms. Today, the business processes that these applications encapsulate are held hostage to these incompatible systems. To share these business processes on the Internet and make them available to business partners, it is necessary to have a distributed architecture that allows disparate applications to communicate with each other. Different attempts have been made to provide a standard distributed architecture. These architectures are generally evaluated in terms of three criteria: language independence, platform independence, and availability of an easy-to-use, standardized implementation.

Microsoft DCOM allows ActiveX components to communicate with each other in a distributed environment. DCOM is somewhat language independent: ActiveX components can theoretically be written in different languages. However, as a primarily Windows-based architecture, DCOM is not platform independent.

Java RMI (Remote Method Invocation) allows components written in Java to communicate with each other in a distributed environment. RMI is platform independent – Java components run on any platform that has a Java Virtual Machine – but it only supports components written in Java.

OMG CORBA allows components written in any language – providing that language has a CORBA binding – and running on any platform to communicate with each other. CORBA has been the closest thing to the much-needed federator of business applications.

However, its heavy weight and complex nature, along with the incompatibility between different CORBA Object Request Brokers (ORB) implementations prevented its widespread adoption. In addition, CORBA's IIOP protocol doesn't integrate well with existing Web standards. For example, many firewalls are reluctant to let anything but HTTP packages penetrate.

The industry needs a communication protocol that enables applications isolated on the Internet to interoperate. This protocol must be language and platform independent. For broad adoption and easy implementation, it must also be simple, lightweight, and provide a good integration with existing Web protocols.

Changing Business Relationships

In a fast changing economy, the nature of business relationships constantly changes. Reorganization, mergers, and new business partnerships are relentlessly negotiated. This creates a major integration challenge for IT departments.

Adapting to the architecture of a new business can be cumbersome and slow down integration. In some cases, the complexity of the task at hand can even jeopardize the integration all together.

For that reason, the industry needs an architecture that enables applications to interoperate in a *loosely coupled* environment. This requires broadly adopted standards that allow a server application (or service provider) to publish and describe the services it provides and a client application (service requestor) to discover and dynamically invoke these services.

Accelerated Pace of Change

Technology changes fast and IT departments often face the daunting task of architecting for an unknown future. This is why IT departments are clamoring for a standards-based architecture that has broad industry support.

Web Services: A New Web Model

What are Web Services?

Web services are next-generation e-business applications with an architecture that removes the stumbling blocks associated with e-business that were highlighted above.

Web services unlock previously isolated business functions. They allow incompatible systems to interoperate on the Internet and on Intranets regardless of language, platform, and operating systems. A Microsoft .NET component can communicate with a J2EE component such as an Enterprise JavaBean (EJB) and vice-versa. An application residing on a Palm device could invoke a Web service residing on a mainframe.

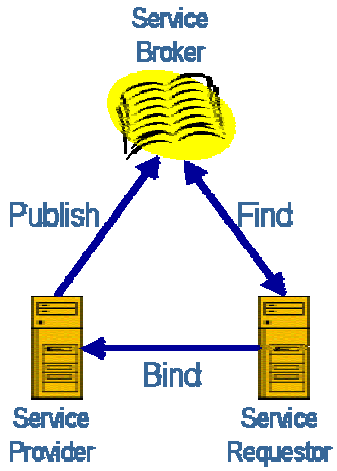
Web services are self-containing, self-describing modular applications that can be published, located, and invoked across the Web. Simply put, Web services are business functions made universally available on the Internet. They allow applications to interoperate in a loosely coupled environment, discovering, and connecting dynamically to services without any previous agreements having been established between them.

The Web services architecture is based on three standards:

- Simple Object Access Protocol (SOAP): The XML-based communication protocol used to access Web services.
- Web Services Description Language (WSDL): The Web services description API
- Universal Description, Discovery, and Integration (UDDI): The publishing and discovery API for Web services

Service-Oriented Architecture

Web services are based on a service-oriented architecture. A service-oriented architecture typically involves three participants: the service provider, the service broker, and the service requestor.



- Figure 1: Service-Oriented Architecture

- The role of the service provider is to create the service and *publish* its description in a UDDI repository.
- The service broker maintains a UDDI repository and acts as a white and yellow page service for Web services.
- The service requestor *finds* a service in a UDDI repository and then *binds* to that service.

Hype and Reality

There has been a lot of hype around Web services. For example, it is not uncommon to hear that applications can be built from Web Services that are dynamically selected at runtime – based on their cost, quality, and availability. While this might become true, this is still hype at the present state of the technology. The reality today is that Web services enable incompatible systems to interoperate on the Internet and on Intranets. For this reason, Macromedia currently defines a Web service in terms of a SOAP object and its WSDL description.

Web Services Example

A company has a Web application that enables users to submit purchase orders. The application allows customers to fill in a purchase order using a browser, but it does not automate the ordering process by interoperating with other businesses.

Using the Web services architecture, the purchase ordering system can be exposed as a Web service. This service accepts orders over the Internet as an XML form over SOAP. The rules for using the service are described in a WSDL document.

A partner of this company would request the WSDL document. The partner would use this WSDL document to create an application that consumes this Web service (again using XML and SOAP as described by the WSDL document). This application would be integrated with their order/inventory management system to automate the ordering process.

This integration establishes a closer supplier/buyer relationship and improves efficiency for both companies. This can easily translate into improved profitability for both.

SOAP

Simple Object Access Protocol (SOAP) is a lightweight XML-based protocol for sending messages to and invoking methods on remote objects. Messages and method invocations are defined as XML documents and are typically sent over HTTP. The SOAP specification also allows other transport protocols to be used. This includes SMTP, FTP, and JMS. Because of its good integration with existing Web protocols and the ubiquitous nature of XML, SOAP is rapidly becoming the universal communication protocol that allows incompatible systems to interoperate. SOAP is both language and platform independent. Because it is based on XML and existing transport protocols, SOAP is also easy to implement and doesn't require the addition of heavy infrastructure at the endpoints. SOAP is the backbone of the Web services architecture.

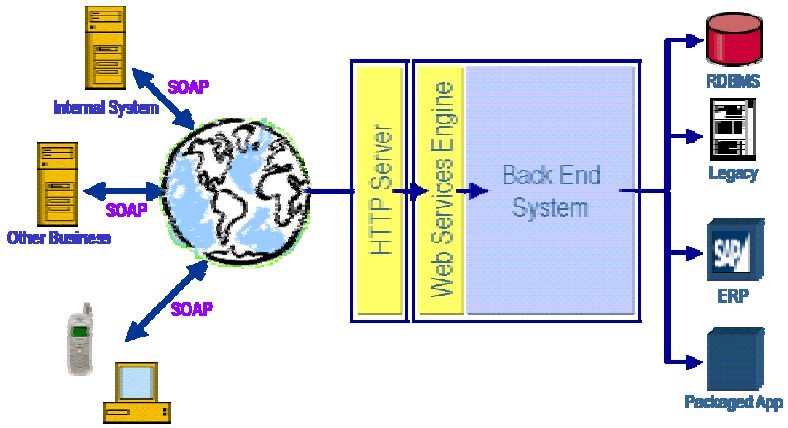


Figure 2: Typical Web services architecture

Figure 2 illustrates a typical Web services scenario. A service requestor – an application on a company's Intranet, a partner application on the Internet, or an application on an end-user device – invokes a method on a Web service using the SOAP protocol. The method invocation is sent as an XML document embedded in an HTTP request. The Web server receives the HTTP request and passes it to the Web services engine. The Web services engine reads the XML-formatted method invocation and invokes the appropriate method on the backend system. The actual object providing the business logic behind the Web service can be a Java class, an EJB, a .NET component, or could even be some kind of Cobol program. The Web services engine generates an HTTP response that embeds the result of the method invocation formatted as an XML document. The HTTP response is sent back to the service requestor.

You can find additional information about SOAP in the W3C's SOAP 1.1 note by visiting:

http://www.w3.org/TR/SOAP/

WSDL

WSDL is the Web Services Description Language. The WSDL description of a service is an XML document with a specific format that provides the technical aspects of the service. The description includes the operations that can be invoked, their associated data types, the supported transport protocols, and the endpoint for the service. The description is typically used by tools to automatically generate client code that accesses the service. WSDL is currently being standardized. Its specification is at version level 1.1.

You can find additional information about WSDL in the W3C's WSDL note by visiting:

http://www.w3.org/TR/wsdl

UDDI

Universal Description, Discovery, and Integration (UDDI) is a SOAP-based API for publishing and discovering Web services.

The publishing API allows service providers to register themselves and their services with a UDDI registry. A UDDI registry can be viewed as the yellow pages for Web services. UDDI registry nodes replicate Web services information among them to provide the same information from any node.

The discovery API allows service subscribers to search for available services. The UDDI registry provides the WSDL document allowing the consumer to use the Web service. The way a UDDI registry is searched is similar to how domain names are looked up in the Web using the DNS architecture.

You can find additional information about UDDI by visiting:

http://www.uddi.org/about.html

Macromedia Involvement in Web Services

Macromedia has been involved in Web services before the term was coined and continues to provide industry leadership in the development and implementation of the Web services standards.

Macromedia has been involved in Web services technologies since 1998 when the company shipped an open source technology called Web Distributed Data Exchange (WDDX). WDDX provides a language and platform-independent architecture for application integration on the Web. WDDX allows developers to integrate Java, PHP, ColdFusion, and Microsoft technologies in a distributed environment.

Macromedia also participates in the W3C working group on XML protocol (XMLP) and the company co-submitted the WSDL specification with IBM, Microsoft and others to the W3C.

In addition, Macromedia participates in W3C working groups and, as part of the Java Community Process (JCP), is active in many Java expert groups related to XML and Web services including JAXB (Java APIs for XML data binding), JAXM (Java APIs for XML messaging), and JAX-RPC (Java APIs for XML RPC).

Macromedia has also taken a leadership role in open source efforts related to Web services. The company dedicates full-time engineers to work on the Axis Web services engine, a project of the Apache Software Foundation. The project also includes team members from IBM, HP, and other vendors.

Web Services Support in JRun

Macromedia engineers leverage this experience to provide JRun with a powerful Web services engine that supports SOAP 1.1, WSDL 1.0, and additional productivity features. Building a Web service in JRun is as simple as building a regular Java class or an EJB and then exposing it as a Web service. Existing Java classes and EJBs can instantly be turned into Web services. You can also create object and tag-based clients that invoke methods on remote Web services even when those services reside on non-Java platforms – such as Microsoft's .NET platform.

A preview of this technology can be installed today on JRun 3.1. The download is available at: www.macromedia.com.

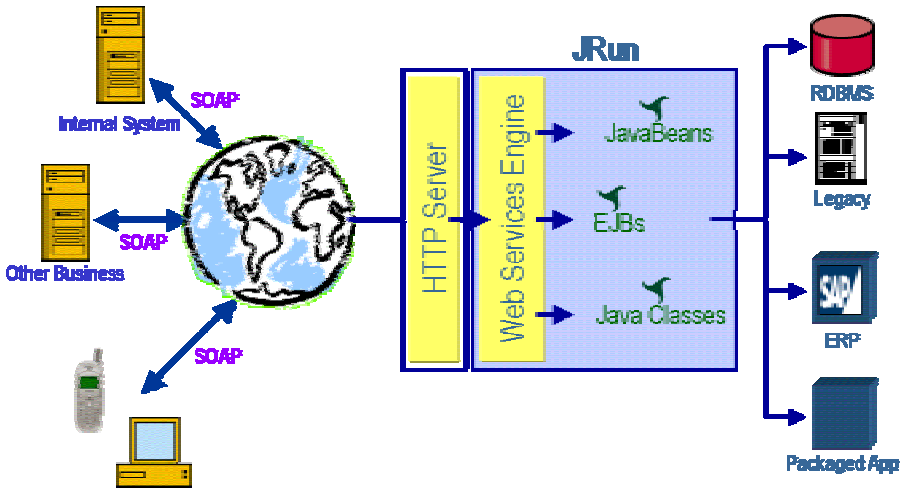


Figure 3: JRun Web services architecture

Figure 3 illustrates a typical Web services scenario in JRun. A service requestor – an application on a company's Intranet, a partner application on the Internet, or an application on an end-user device – invokes a method on a Web service using the SOAP protocol. The method invocation is sent as an XML document embedded in an HTTP request. The Web server receives the HTTP request and passes it to JRun's Web services engine, which reads the XML-formatted method invocation and invokes the appropriate method on the Java class or EJB that provides the business logic for the Web service. Axis generates an HTTP response that embeds the result of the method invocation formatted as an XML document. The HTTP response is sent back to the service requestor.

Additional support for Web services includes:

- The generation of WSDL files
- The generation of client code based on a WSDL file
- The automatic deployment of Web services located in WAR or EAR files
- A Web services assembly tool
- A JSP custom tag library for invoking Web services operations from JSP clients

Other Web Services Initiatives at Macromedia

Macromedia is committed to providing the highest level of support for Web services across all products. In addition to JRun, superior Web services integration is also being built into other Macromedia products including Flash, UltraDev, and ColdFusion.