

This article originally published by:
The World Organization of Webmasters (WOW),
Vol. 4 Issue 03, April-May 2002
http://www.joinwow.org/newsletter/04_4/bil/

Setting the Stage for Web Services

By: Devan Shepherd - CEO & Chief Technical Officer, XMaLpha Technologies, LLC.

This is the first in a series of short articles about the issues, design considerations, players and emerging technologies that are being used to implement Web services. In this article, an initial definition for Web Services is offered and a few of the major players that have emerged as early leaders in Web service development options are identified. As the series progresses, I will examine the issues surrounding the construction and deployment of Web services while exploring the component technologies, such as the eXtensible Markup Language (XML), the Simple Object Access Protocol (SOAP), the Web Services Description Language (WSDL), the Universal Description, Discovery and Integration standard (UDDI) and selected vendor-specific tools used for creation, deployment and consumption of Web services.

What are Web Services?

You have probably read something about them, heard something about them and may have even used them, without really being able to say definitively just what Web services are. Major vendors are announcing tools and technologies that purport to make the development and deployment of Web services more approachable, and yet there isn't a single definition for just what these mysterious entities are. Like many new technologies in Information Technology, until the standards themselves settle down, and toolsets mature, the definitions will seem more like moving targets scattering over a landscape of loosely related concepts. For now, let's place some scope around the concept to provide a common understanding of just how important Web services are in the fabric of technological innovation.

From an application standpoint, Web services might include anything from popular courier company package tracking systems, through product catalogue management, interactive inventory monitoring, and credit card verification/validation services. Other typical examples include language translation, currency conversion, universal login systems (e.g., MS Passport), price-comparison shopping "bots", portfolio and stock tracking agents, and almost any other service one can conceive of. Web services provide component functionality while utilizing existing technologies, such as XML and the HTTP protocol, to enable unrelated applications to communicate with one another.

In a general sense a Web service is a platform-independent software component, interface, or program that provides information derived via a predefined function, through programmatic exchange between two distinct processes, often over the Internet. Well even that sounds complex, but in simple terms a Web service can be any code, such as an object or program, which makes information available over standard

Internet and Web protocols, like HTTP. Although, just to make the whole topic even more confusing, sometimes a Web service may have no reliance on the Web for transfer of data. Web services, in fact, can describe simple programmatic exchanges of data between two processes on the same machine, not traversing the Web at all. For the most part, however, Web services, particularly as the large-scale players in the industry promote them, include functional software components that are:

- Exposed to client applications over the Internet;
- Written to exploit the power and intelligence of standardized XML messaging;
- Registered on a repository service or object catalogue;
- Defined by a service description language;
- Invoked through an accessible Application Programming Interface (API);
- Obtainable over a network, via a standard protocol;
- Often comprising other services

To qualify as a Web Service, a component or program, must be fully described using a description language formulated on a known vocabulary, such that those wishing to invoke the code know precisely the details required to interact with the service. This means that the expected message format, available transport protocols, precise service location, and detailed object or program behaviors are all defined and published for retrieval using a standard XML description vocabulary, such as WSDL. In subsequent articles, we will explore each of these aspects of Web services in more detail.

Amongst the definitive qualities of Web services is the fact that the programmatic interface provides "black box" functionality, hiding the implementation details so that they may be invoked using practically any software or hardware platform. This means that Web services are available, regardless of their underlying technology suites and, are therefore ideally suited to use within Business-to-Business Integration (B2Bi) solutions. In other words, Web services designed, built and running on one platform, say Microsoft .NET, can be invoked by applications running on a different platform, such as Unix, or Macintosh OS, etc. This is accomplished by relying on XML encoding vocabularies, standard communication protocols and Internet technologies that seamlessly cross the platform boundaries of the Web. For instance, the protocols like SMTP, HTTP, and FTP serve to relay information with no inherent awareness or requirement regarding the platforms on which data originate. Likewise, XML and the descriptive access protocols that are based on XML, such as SOAP, offer universal, ubiquitous, encoding logic that is completely separate from implementation.

Web services, then, are not accessed via any object-model-specific, application language-specific, or platform-specific protocols. This releases Web services from the bonds of dependencies that plagued predecessor technologies. The vision that is quickly emerging, is that Web services will certainly improve upon and ultimately replace proprietary protocols, such as Distributed Component Object Model (DCOM), Remote Method Invocation (RMI), or Internet Inter-ORB Protocol (IIOP) in the universe of transaction data exchange methodologies.

Web services are characterized, somewhat strictly, in terms of the messages they accept and generate. This approach represents a definitive and agnostic service exposure, creation and consumption model that describes a distinct operation. As such, Web

"...Using XML Technology to turn structured data into intelligent business knowledge..."™

services can be operated alone or in combination with other Web services as aggregate transactions in complex business data exchange models.

Having stated all of these characteristics, it remains that the definition of a Web service is still a bit broad and general. This harks back to the introduction of client-server technologies, or Object Oriented Programming in the early days. It wasn't until the technical boundaries for either were stabilized that everyone could really agree on just what those technologies were. The standards for Web services are not mature, by any means, and as such lend themselves to interpretation by vendors and programmers alike. This is an evolutionary process.

Which vendors are Implementing Technologies for Web Services?

An easier question to answer might be: which companies are not leveraging Web service technologies? For the purpose of this article, consider just IBM, Microsoft and Sun. All three have announced major development and paradigm releases that revolve entirely around the delivery of Web services. IBM offers WebSphere as a solution, while Microsoft has retooled all of its software to focus on the .NET platform, and Sun promotes Java development via the Sun One toolset. Each of these companies, not surprisingly, has a different delivery and marketing focus, but all are starting to agree upon a fundamental set of "roles" or service behaviors, and a core stack of protocols, including XML, SOAP, HTTP, UDDI, and WSDL.

In subsequent articles, each of the standard roles (e.g., service requestor, service provider, service registry), and the component protocols will be explored in greater detail. Until then, if you would like a hands-on test-drive of Web services for yourself, visit IBM's demonstration Web services browser at:

<http://demo.alphaworks.ibm.com/browser>.

Biography:

*Devan Shepherd is the author of **Teach Yourself XML in 21 Days**, 2/e, ISBN: 0-672-32093-2. He has more than 25 years of progressive experience in the IT industry as a developer, executive, solutions provider, public speaker, journal author, and instructor. Devan is CEO and Chief Technical Officer of **XMaLpha Technologies, LLC** (<http://XMaLpha.com>), a successful consulting and technology training practice. XMaLpha has a proven track record of providing leading-edge high tech solutions for public and corporate clients with a focus on XML, Web Services, and e-Business. Devan, licensed by the Minnesota State Colleges and Universities directorate also holds a full research and faculty position at St. Paul Community College, where he teaches XML and other Web Development technologies, database fundamentals, programming and various IT courses.*

"...Using XML Technology to turn structured data into intelligent business knowledge..."™