This article is taken from our forthcoming publication *Web Services: A Technical Introduction*, which will be available in late July. For pre-publication information, please visit **www.deitel.com** and periodically check **www.informit.com/deitel** for updates on ordering this new publication.

- View the **Table of Contents**

To view all the Deitel products and services available, visit the Deitel Kiosk on InformIT at **www.informIT.com/deitel**.

To follow the Deitel publishing program, sign-up now for the *DEITEL™ BUZZ ONLINE* e-mail newsletter at **www.deitel.com/newsletter/subscribeinformIT.html**.

To learn more about our Deitel instructor-led corporate training courses that can be delivered at your location, visit **www.deitel.com/training** or contact our Director of Corporate Training Programs at (978) 461-5880 or e-mail: **christi.kelsey@deitel.com**.

Note from the Authors: This DEITEL™ article is an excerpt from Chapter 2, Sections 2.2 and 2.7 of *Web Services: A Technical Introduction*. Web services are software programs that use XML to exchange information with other software via common Internet protocols. This article introduces key concepts behind Web services and the standards used to enable Web service interactions, including SOAP (Simple Object Access Protocol), WSDL (Web Services Description Language) and UDDI (Universal Description, Discovery and Integration).

2.2 What are Web Services?

Anyone familiar with the information-technology industry is aware of the current buzz surrounding Web services. For example, over 66 percent of respondents to a 2001 *InfoWorld* magazine poll at least somewhat agreed that "Web service are likely to emerge as the next business model of the Internet."[1] In addition, Gartner Group predicts that Web services could increase the efficiency of IT projects by up to 30 percent.[2] But what are Web services, and how will they change the nature of business conducted over the Internet?

To facilitate business processes, enterprise applications must communicate with one another and share data. Historically, this was accomplished through proprietary specifications and data formats. However, the emergence of the World Wide Web and *XML* (*eXtensible Markup Language*)—an open technology for data exchange—has increased the possibility for interoperable system-to-system communication. Web services are software programs that use XML to exchange information with other software via common Internet protocols. Basically, a Web service communicates via the Web to provide *operations* (specific tasks performed by computers, often called *methods* or *functions*) that yield some result to an application. This means that an application residing on one computer can send requests to, and receive responses from, applications running on other computers, all using the Internet infrastructure. Web services can transfer data using many Internet protocols, but most employ *Hypertext Transfer Protocol* (*HTTP*)—the key communication protocol of the World Wide Web.

Web services can perform almost any kind of task. For example, a Web portal might obtain top news headlines from an Associated Press Web service, whereas a financial application might employ a Web service that checks a stock quote and returns its current value. A Web service's functionality can be as trivial as multiplying two numbers together—or as complex as the functions carried out by an entire customer relationship management (CRM) software system.

Web services possess certain characteristics that distinguish them from other computing models. Paul Flessner, senior vice president of Microsoft's .NET Enterprise Server Division, identifies several of these traits in his writings on Web services. First, Web services are programmable. A Web service encapsulates a task; when an application passes data or instructions to it, the service processes that information and returns something to the application. Second, Web services are based on XML. XML and the XML-based *SOAP* (*Simple Object Access Protocol*) are technologies that enable Web services to communicate with other applications, even if those applications are written in different programming languages and run on different platforms. Web services also are self-describing, meaning that they are accompanied by information explaining what they do and how other applications can access and use them. These descriptions typically are written in *WSDL* (*Web Services Description Language*), an XML-based standard explored later in this chapter. In addition, Web services are discoverable. This refers to the ability of applications and developers to search for and locate desired Web services through registries, such as those based on *UDDI* (*Universal Description, Discovery and Integration*), another emerging Web services standard.[3]

2.7 Key Web Services Technologies

As we explained earlier, part of what distinguishes Web services from similar computing models is the use of XML and other specific technical standards—most commonly SOAP, WSDL and UDDI. These technologies enable communication among applications in a manner that is independent of specific programming languages, operating systems and hardware platforms. SOAP provides a communication mechanism between Web services and applications, WSDL offers a uniform method of describing Web services to other programs and UDDI enables the creation of searchable Web services directories. When deployed together, these technologies allow developers to package applications as services and publish those services on the Web.

Figure 2.6 depicts the role of various standards in common Web services architectures. Although readers might be unfamiliar with the technologies and relationships portrayed in the diagram, the next sections provide clarification by describing each core Web services standard.

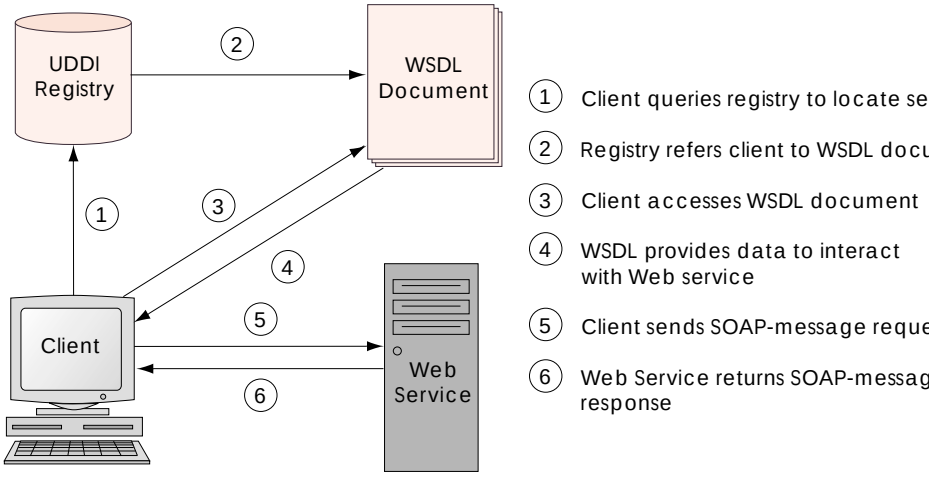

Fig. 2.6 SOAP, UDDI and WSDL in a Web service interaction.

2.7.1 XML (eXtensible Markup Language)

XML is a widely supported, *open* (i.e., non-proprietary) technology initially released in 1996 by the World Wide Web consortium (W3C) to facilitate data exchange. As the popularity of the Web exploded in the 1990s, the limitations of HTML (Hypertext Markup Language) became apparent. HTML's lack of extensibility (the ability to change or add features) and its inability to describe the data it formatted frustrated developers. Furthermore, its ambiguous definitions allowed erroneous HTML to proliferate. In response to these problems, the W3C added limited extensibility to HTML and created a new technology for formatting HTML documents, called Cascading Style Sheets (CSS). These were, however, only temporary solutions—the need for a standardized, fully extensible and structurally strict language was apparent. As a result, the W3C created XML. XML combines

the power and extensibility of its parent language, *Standard Generalized Markup Language (SGML)*, with the simplicity that the Web community demands.

Data independence, or the separation of content from its presentation, is the essential characteristic of XML (and, we should note, one that is lacking in HTML). Because XML documents describe only data, any application that understands XML—regardless of the application's programming language or platform—has the ability to format XML in a variety of different ways. Recognizing this, software developers are integrating XML into their applications to improve Web functionality and interoperability. XML documents contain data, but no formatting instructions, so applications that process XML documents must decide how to display the documents' data. For example, a PDA (personal digital assistant) might render an XML document differently than a wireless phone or desktop computer would render that document.

When an application encounters an XML document, the application must ensure that the document contains the particular data that the application expects. Such assurance involves processing the XML document with a software program called an *XML parser*. Parsers check an XML document's syntax. They also enable software programs to interpret marked-up, or formatted, data. An XML document optionally can reference another document that defines the XML document's structure. This other document is either a *Document Type Definition (DTD)* or a *schema*. When an XML document references a DTD or schema, some parsers (called *validating parsers*) can read the DTD/schema and check that the XML document follows the structure that the DTD/schema defines. If an XML parser can process an XML document successfully, that XML document is well formed (syntactically correct).

As applications become more Web enabled, it seems likely that XML will become the universal technology for representing data passed between Web applications. All applications employing XML will be able to communicate, provided that they can understand each others' XML markup, or vocabulary. This high level of interoperability makes XML an ideal technology to enable Web services, which communicate among systems without regard to operating systems and hardware platforms. The core standards enabling Web services, which are described in the next three sections, are based on XML.

2.7.2 SOAP (Simple Object Access Protocol)

SOAP is one of the most common standards used to deliver Web services. Initially developed by representatives from DevelopMentor, Userland Software and Microsoft, SOAP was conceptualized in 1998 and released publicly as SOAP 1.0 in 1999.[4] The newest version of SOAP, SOAP 1.2, was published by the W3C in December 2001. The release of SOAP 1.2 represents a major progression of the standard and, in part, explains the current industry excitement regarding SOAP and Web services deployments.

The purpose of SOAP is to enable data transfer between systems distributed over a network. When an application communicates with a Web service, SOAP messages are the means through which the two systems exchange data. A SOAP message sent to a Web service invokes a method provided by the service, meaning that the message requests that the service perform a particular task. The service then uses information contained in the SOAP message to perform its function and return the results via another SOAP message.

As an XML-based communication protocol, SOAP basically consists of a set of standardized XML schemas. The schemas define a format for transmitting XML messages over a network, including the types of data that the message can include and the precise way in

which the message must be structured so that the server on the other end can interpret it correctly.[5] SOAP primarily is used to transfer data over the Internet using protocols such as HTTP; however, SOAP also can facilitate exchange over other networks, such as local area networks (LANs) within companies. The use of HTTP allows Web services to communicate across firewalls, because most firewalls are designed to accept HTTP service requests.[6]

A SOAP message consists of three main parts: An *envelope*, a *header* and a *body*. The envelope wraps the entire message; it describes the content of the message and indicates the message's intended recipient. The next part of the SOAP message is the header, which is an optional element that provides information regarding such topics as security and routing. The body of the SOAP message contains the application-specific data that is being communicated. The data is marked up as XML and adheres to a specific format, which is defined by the schemas we mentioned earlier. This formatting enables the recipient to process the data correctly. SOAP messages are received and interpreted by SOAP servers, which, in turn, trigger Web services to perform their tasks.

Several protocols could be used instead of SOAP to enable Web services. For example, *XML-RPC* is an older technology that provides similar functionality. However, most major software vendors have chosen to support SOAP over other possible technologies. There are several technical reasons for industry support of SOAP, many of which refer to aspects of the protocol beyond the scope of this text. However, it is important to note the main advantages of SOAP—simplicity, extensibility and interoperability.[7] Basic SOAP messages do not involve extensive amounts of code, and there is little "special software" needed either to send or to receive SOAP messages. SOAP also provides mechanisms that enable developers to extend the standard to meet specific needs. Furthermore, because SOAP employs XML to communicate over HTTP, SOAP can be used to transfer data between any two systems that are connected to the Internet, regardless of programming languages, operating systems and hardware platforms.

2.7.3 WSDL (Web Services Description Language)

Another standard that plays a crucial role in enabling Web services is WSDL. We mentioned earlier that a defining feature of Web services is that they are self-describing, meaning that every Web service is accompanied by information that enables developers to employ the service. These descriptions are written in WSDL, an XML-based language through which a Web service conveys to other applications the methods that the service provides and how those methods can be accessed.

When SOAP and other Web services technologies were first developed, software vendors realized that applications calling services across a network would need information about a specific service before interacting with it. However, each vendor began building its own method of description, resulting in service descriptions that were incompatible with one another. The WSDL specification emerged when vendors Microsoft and IBM decided to combine their description technologies into a universal standard. In March 2001, Microsoft, IBM and Ariba submitted WSDL 1.1 to the W3C; a W3C technical committee is working to standardize the language further. Although the technology is still under development, nearly all Web services products now provide support for WSDL.

Every Web service published on the Internet is accompanied by an associated WSDL file, which lists the service's capabilities, states its location on the Web and provides

instructions regarding its use. A WSDL file defines the kinds of messages a Web service can send and receive, as well as specifying the data that a calling application must provide for the Web service to perform its task. Theoretically, an application searching for a Web service to fill a specific need could analyze the WSDL files of comparable services, then use the WSDL data to choose between the services. WSDL files also provide specific technical information that informs applications how to connect to and communicate with Web services over HTTP or another communications protocol.

It is important to realize that WSDL is a language meant to be read by applications, rather than by humans. Although the structure of WSDL files might appear complex, computers that understand WSDL can process the files and extract the information they need. Furthermore, most Web services development tools generate WSDL files automatically. This means that, if a programmer develops a Web service, the software used to build the service creates an appropriate WSDL file for that service. Therefore, it is not necessary for developers to understand the syntax of WSDL fully when building and deploying Web services.

2.7.4 UDDI (Universal Description, Discovery and Integration)

The third major Web services standard, UDDI, enables developers and businesses to publish and locate Web services on the Internet. Originally designed by Microsoft, IBM, Ariba and 50 other companies, UDDI began as a way for users of B2B exchanges to share information about their businesses and business processes with potential partners and affiliates.[8] As its name implies, the specification allows companies to describe their own services and electronic processes, discover those of other companies and integrate others' services into their systems. Although UDDI is a relatively new standard (the first version was published in September 2000), it has acquired significant industry backing. An industry consortium called the UDDI project oversees UDDI's development, but plans eventually to turn control of the specification over to a standards body. The most recent version of UDDI, UDDI 2.0, was released in June 2001; UDDI 3.0 is under development.[9]

UDDI defines an XML-based format in which companies can describe their electronic capabilities and business processes; the specification also provides a standardized method of registering and locating the descriptions via the Internet. Part of the information that companies can supply is data regarding available Web services. Companies can store their information either in private UDDI registries, which are accessible only to approved business partners, or in public UDDI registries, which any interested party can access. The largest, most comprehensive public UDDI registry is the *UDDI Business Registry* (*UBR*), which was developed by the UDDI project to facilitate the formation of new business relationships. Four implementations of the UBR—hosted by Microsoft, IBM, Hewlett-Packard and SAP—are in operation. The four host companies have synchronized their registries, so information entered in one is replicated in the other three.

The structure of UDDI registries is modeled on that of the phone book. Registries contain "white pages," where companies list contact information and textual descriptions of themselves; "yellow pages," which provide classification information about companies and details on companies' electronic capabilities; and "green pages," which list technical data relating to services and business processes.[10] Information regarding businesses and services is highly categorized, enabling companies to search for desired partners or services. Then, IT staffs can use the technical information in the registries to link electronically to

other businesses. In this manner, UDDI simplifies the process of creating B2B relationships and connecting electronic systems to exchange data and services.

Vendor interest in UDDI registries is already strong—most Web services players support UDDI and have incorporated the standard into their products. NTT Communications of Tokyo and other companies are in the process of building additional implementations of the UDDI Business Registry. However, businesses have been slow to enter information in the public registries. This has not surprised industry experts, who believe that companies will begin by building private registries shared among partners. Large organizations also can create private registries to organize their own Web services and make the services available to other departments. Most agree that, once the technology has matured and users are comfortable with it, public exchanges will become more popular.

It is difficult for large organizations to change the ways in which they communicate, form partnerships, locate clients and transact business. Some companies are hesitant to abandon older B2B communications mechanisms, whereas others are concerned about the security issues raised by exposing corporate data or applications on the Web. However, organizations are slowly realizing that technologies such as UDDI can improve business processes and provide competitive advantages.

WORKS CITED

1. J. Borck, “Leaders of the Web Services Pack,” *InfoWorld* 17 September 2001: 52.
2. P. Fingar, “Web Services Among Peers,” *Internet World* January 2002: 21.
3. P. Flessner, “XML Web Services: More Than Protocols and Acronyms,” *Software Development Times* 15 August 2001: 31.
4. W. Oellermann, Jr., *Architecting Web Services* (Berkeley, CA: Apress, 2001) 602.
5. B. Howerton, “Keeping Promises,” *Intelligent Enterprise* 1 January 2002: 46.
6. S. Hildreth, “Web Services: The Next Generation of Distributed Computing,” 9 April 2001 <`e-serv.ebizq.net/wbs/hildreth_1.html`>.
7. P. Cauldwell, et al., *Professional XML Web Services* (Birmingham, UK: Wrox Press 2001) 21.
8. P. Korzeniowski, “A Little Slice of the UDDI Pie,” *eWeek* 4 February 2002: 50.
9. P. Korzeniowski, “A Little Slice of the UDDI Pie,” *eWeek* 4 February 2002: 51.
10. T. Wilson, “UDDI Promises Link to Web Services,” *Internet Week* 26 November 2001: 26.