Published on The O'Reilly Network (http://www.oreillynet.com/)

http://www.oreillynet.com/pub/a/webservices/2002/02/12/webservicefaqs.html
See this if you're having trouble printing code examples

Top Ten FAQs for Web Services

by Ethan Cerami, author of Web Services Essentials
02/12/2002

Web services represent an important evolutionary step in building distributed applications. But, what exactly is a Web service? What is the Web service protocol stack? And, does the World Wide Web Consortium support any Web service standards?

Below are answers to the top ten most frequently asked questions (FAQs) about Web services. Together, they provide an overview of the Web services landscape as well as links to additional resources for more in-depth material.

1. What is a Web service?

Many people and companies have debated the exact definition of Web services. At a minimum, however, a Web service is any piece of software that makes itself available over the Internet and uses a standardized XML messaging system.

XML is used to encode all communications to a Web service. For example, a client invokes a Web service by sending an XML message, then waits for a corresponding XML response. Because all communication is in XML, Web services are not tied to any one operating system or programming language--Java can talk with Perl; Windows applications can talk with Unix applications.

Beyond this basic definition, a Web service may also have two additional (and desirable) properties:

- First, a Web service can have a public interface, defined in a common XML grammar. The interface describes all the methods available to clients and specifies the signature for each method. Currently, interface definition is accomplished via the Web Service Description Language (WSDL). (See FAQ number 7.)

- Second, if you create a Web service, there should be some relatively simple mechanism for you to publish this fact. Likewise, there should be some simple mechanism for interested parties to locate the service and locate its public interface. The most prominent directory of Web services is currently available via UDDI, or Universal Description, Discovery, and Integration. (See FAQ number 8.)

Web services currently run a wide gamut from news syndication and stock-market data to weather reports and package-tracking systems. For a quick look at the range of Web services currently available, check out the XMethods directory of Web services.

2. What is new about Web services?

People have been using Remote Procedure Calls (RPC) for some time now, and they long ago discovered how to send such calls over HTTP.

So, what is really new about Web services? The answer is

XML.

XML lies at the core of Web services, and provides a common language for describing Remote Procedure Calls, Web services, and Web service directories.

Prior to XML, one could share data among different applications, but XML makes this so much easier to do. In the same vein, one can share services and code without Web services, but XML makes it easier to do these as well.

By standardizing on XML, different applications can more easily talk to one another, and this makes software a whole lot more interesting.

3. I keep reading about Web services, but I have never actually seen one. Can you show me a real Web service in action?

If you want a more intuitive feel for Web services, try out the IBM Web Services Browser, available on the IBM Alphaworks site. The browser provides a series of Web services demonstrations. Behind the scenes, it ties together SOAP, WSDL, and UDDI to provide a simple plug-and-play interface for finding and invoking Web services. For example, you can find a stock-quote service, a traffic-report service, and a weather service. Each service is independent, and you can stack services like building blocks. You can, therefore, create a single page that displays multiple services--where the end result looks like a stripped-down version of *my.yahoo* or *my.excite*.

4. What is the Web service protocol stack?

The Web service protocol stack is an evolving set of protocols used to define, discover, and implement Web services. The core protocol stack consists of four layers:

- Service Transport: This layer is responsible for transporting messages between applications. Currently, this includes HTTP, SMTP, FTP, and newer protocols, such as Blocks Extensible Exchange Protocol (BEEP).

- XML Messaging: This layer is responsible for encoding messages in a common XML format so that messages can be understood at either end. Currently, this includes XML-RPC and SOAP.

- Service Description: This layer is responsible for describing the public interface to a specific Web service.

Currently, service description is handled via the WSDL.

- Service Discovery: This layer is responsible for centralizing services into a common registry, and providing easy publish/find functionality. Currently, service discovery is handled via the UDDI.

Beyond the essentials of XML-RPC, SOAP, WSDL, and UDDI, the Web service protocol stack includes a whole zoo of newer, evolving protocols. These include WSFL (Web Services Flow Language), SOAP-DSIG (SOAP Security Extensions: Digital Signature), and USML (UDDI Search Markup Language). For an overview of these protocols, check out Pavel Kulchenko's article, Web Services Acronyms, Demystified, on *XML.com*.

Fortunately, you do not need to understand the full protocol stack to get started with Web services. Assuming you already know the basics of HTTP, it is best to start at the XML Messaging layer and work your way up.

5. What is XML-RPC?

XML-RPC is a protocol that uses XML messages to perform Remote Procedure Calls. Requests are encoded in XML and sent via HTTP POST; XML responses are embedded in the body of the HTTP response.

More succinctly, XML-RPC = HTTP + XML + Remote Procedure Calls.

Because XML-RPC is platform independent, diverse applications can communicate with one another. For example, a Java client can speak XML-RPC to a Perl server.

To get a quick sense of XML-RPC, here is a sample XML-RPC request to a weather service (with the HTTP Headers omitted):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodCall>
   <methodName>weather.getWeather</methodName>
   <params>
      <param><value>10016</value></param>
   </params>
</methodCall>
```

The request consists of a simple `<methodCall>` element, which specifies the method name (`getWeather`) and any method parameters (`zip code`).

Here is a sample XML-RPC response from the weather service:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodResponse>
   <params>
      <param>
         <value><int>65</int></value>
      </param>
   </params>
</methodResponse>
```

The response consists of a single `<methodReponse>` element, which specifies the return value (the current temperature). In this case, the return value is specified as an integer.

In many ways, XML-RPC is much simpler than SOAP, and therefore represents the easiest way to get started with Web services.

The official XML-RPC specification is available at XML-RPC.com. Dozens of XML-RPC implementations are available in Perl, Python, Java, and Ruby. See the XML-RPC home page for a complete list of implementations.

6. What is SOAP?

SOAP is an XML-based protocol for exchanging information between computers. Although SOAP can be used in a variety of messaging systems and can be delivered via a variety of transport protocols, the main focus of SOAP is Remote Procedure Calls (RPC) transported via HTTP. Like XML-RPC, SOAP is platform independent, and therefore enables diverse applications to communicate with one another.

To get a quick sense of SOAP, here is a sample SOAP request to a weather service (with the HTTP Headers omitted):

```
<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://www.w3.org/2001/09/soap-envelope"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <SOAP-ENV:Body>
      <ns1:getWeather
         xmlns:ns1="urn:examples:weatherservice"
         SOAP-ENV:encodingStyle=" http://www.w3.org/2001/09/soap-encoding
         <zipcode xsi:type="xsd:string">10016</zipcode>
      </ns1:getWeather>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

As you can see, the request is slightly more complicated than XML-RPC and makes use of both XML namespaces and XML Schemas. Much like XML-RPC, however, the body of the request specifies both a method name (`getWeather`), and a list of parameters (`zipcode`).

Here is a sample SOAP response from the weather service:

```
<?xml version='1.0' encoding='UTF-8'?>
<SOAP-ENV:Envelope
   xmlns:SOAP-ENV="http://www.w3.org/2001/09/soap-envelope"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <SOAP-ENV:Body>
      <ns1:getWeatherResponse
         xmlns:ns1="urn:examples:weatherservice"
         SOAP-ENV:encodingStyle="http://www.w3.org/2001/09/soap-encoding">
         <return xsi:type="xsd:int">65</return>
      </ns1:getWeatherResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The response indicates a single integer return value (the current temperature).

The World Wide Web Consortium (W3C) is in the process of creating a SOAP standard. The latest working draft is designated as SOAP 1.2, and the specification is now broken into two parts. Part 1 describes the SOAP messaging framework and envelope specification. Part 2 describes the SOAP encoding rules, the SOAP-RPC convention, and HTTP binding details.

7. What is WSDL?

The Web Services Description Language (WSDL) currently represents the service description layer within the Web service protocol stack.

In a nutshell, WSDL is an XML grammar for specifying a public interface for a Web service. This public interface can include the following:

- Information on all publicly available functions.
- Data type information for all XML messages.
- Binding information about the specific transport protocol to be used.
- Address information for locating the specified service.

WSDL is not necessarily tied to a specific XML messaging system, but it does include built-in extensions for describing SOAP services.

Below is a sample WSDL file. This file describes the public interface for the weather service used in the SOAP example above. Obviously, there are many details to understanding the example. For now, just consider two points.

> First, the `<message>` elements specify the individual XML messages that are transferred between computers. In this case, we have a `getWeatherRequest` and a `getWeatherResponse`. Second, the `<service>` element specifies that the service is available via SOAP and is available at a specific URL.

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="WeatherService"
   targetNamespace="http://www.ecerami.com/wsdl/WeatherService.wsdl"
   xmlns="http://schemas.xmlsoap.org/wsdl/"
   xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
   xmlns:tns="http://www.ecerami.com/wsdl/WeatherService.wsdl"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema">

   <message name="getWeatherRequest">
      <part name="zipcode" type="xsd:string"/>
   </message>
   <message name="getWeatherResponse">
      <part name="temperature" type="xsd:int"/>
   </message>

   <portType name="Weather_PortType">
      <operation name="getWeather">
         <input message="tns:getWeatherRequest"/>
         <output message="tns:getWeatherResponse"/>
      </operation>
   </portType>

   <binding name="Weather_Binding" type="tns:Weather_PortType">
      <soap:binding style="rpc"
         transport="http://schemas.xmlsoap.org/soap/http"/>
      <operation name="getWeather">
         <soap:operation soapAction=""/>
         <input>
            <soap:body
               encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
               namespace="urn:examples:weatherservice"
               use="encoded"/>
         </input>
         <output>
            <soap:body
               encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
               namespace="urn:examples:weatherservice"
               use="encoded"/>
         </output>
      </operation>
   </binding>

   <service name="Weather_Service">
      <documentation>WSDL File for Weather Service</documentation>
      <port binding="tns:Weather_Binding" name="Weather_Port">
         <soap:address
            location="http://localhost:8080/soap/servlet/rpcrouter"/>
      </port>
   </service>
</definitions>
```

Using WSDL, a client can locate a Web service, and invoke any of the publicly available functions. With WSDL-aware tools, this process can be entirely automated, enabling applications to easily integrate new services with little or no manual code. For example, check out the GLUE platform from the Mind Electric.

WSDL has been submitted to the W3C, but it currently has no official status within the W3C. See this W3C page for the latest draft.

8. What is UDDI?

UDDI (Universal Description, Discovery, and Integration) currently represents the discovery layer within the Web services protocol stack.

UDDI was originally created by Microsoft, IBM, and Ariba, and represents a technical specification for publishing and finding businesses and Web services.

At its core, UDDI consists of two parts.

- First, UDDI is a technical specification for building a distributed directory of businesses and Web services. Data is stored within a specific XML format, and the UDDI specification includes API details for searching existing data and publishing new data.

- Second, the UDDI Business Registry is a fully operational implementation of the UDDI specification. Launched in May 2001 by Microsoft and IBM, the UDDI registry now enables anyone to search existing UDDI data. It also enables any company to register themselves and their services.

The data captured within UDDI is divided into three main categories:

- White Pages: This includes general information about a specific company. For example, business name, business description, and address.

- Yellow Pages: This includes general classification data for either the company or the service offered. For example, this data may include industry, product, or geographic codes based on standard taxonomies.

- Green Pages: This includes technical information about a Web service. Generally, this includes a pointer to an external specification, and an address for invoking the Web service.

You can view the Microsoft UDDI site, or the IBM UDDI site. The complete UDDI specification is available at uddi.org.

Beta versions of UDDI Version 2 are available at:

- Hewlett Packard
- IBM
- Microsoft
- SAP

9. How do I get started with Web Services?

The easiest way to get started with Web services is to learn XML-RPC. Check out the XML-RPC specification or read my book, Web Services Essentials. O'Reilly has also recently released a book on Programming Web Services with XML-RPC by Simon St.Laurent, Joe Johnston, and Edd Dumbill.

Once you have learned the basics of XML-RPC, move onto SOAP, WSDL, and UDDI. These topics are also covered in *Web Services Essentials*. For a comprehensive treatment of SOAP, check out O'Reilly's Programming Web Services with SOAP, by Doug Tidwell, James Snell, and Pavel Kulchenko.

10. Does the W3C support any Web service

standards?

The World Wide Web Consortium (W3C) is actively pursuing standardization of Web service protocols. In September 2000, the W3C established an XML Protocol Activity. The goal of the group is to establish a formal standard for SOAP. A draft version of SOAP 1.2 is currently under review, and progressing through the official W3C recommendation process.

On January 25, 2002, the W3C also announced the formation of a Web Service Activity. This new activity will include the current SOAP work as well as two new groups. The first new group is the Web Services Description Working Group, which will take up work on WSDL. The second new group is the Web Services Architecture Working Group, which will attempt to create a cohesive framework for Web service protocols.

For an overview of the new W3C Web Service activity, check out Edd Dumbill's recent *XML.com* article, Welcome Web Services Activity.

O'Reilly & Associates will soon release (February 2002) Web Services Essentials.

- You can also look at the Full Description of the book.
- For more information, or to order the book, click here.