Chapter

7.3 SOAP, UDDI and WSDL

SOAP, WSDL and UDDI are the core technologies used in Web services interactions. 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 other applications, WSDL offers a uniform method of describing Web services to other programs and UDDI enables the creation of searchable Web services registries. When deployed together, these technologies allow developers to package applications as services, publish the services on the Web and advertise the services to developers and applications.

Applications typically communicate with Web services via a SOAP messaging framework. SOAP messages encapsulate programmatic instructions that are transported to a SOAP server, which processes the messages and invokes the targeted Web services. The WSDL document associated with a particular Web service informs client applications how to format a SOAP message to send to that service.

Figure 7.1 depicts the general architecture in which Web services operate. The UDDI registry stores the locations of WSDL documents. To publish a Web service, a service provider registers the service’s WSDL document with the UDDI registry. A service consumer discovers the service from the UDDI registry. Once the service consumer knows how to access the Web service, the consumer can communicate with the Web service directly via SOAP messages.

7.4 Universal Description, Discovery and Integration (UDDI)

In September 2000, the UDDI project—led by IBM, Microsoft and Ariba—released Version 1.0 of the UDDI specification. This specification defines a framework for centralized registries that facilitate the storage, discovery and exchange of information about businesses and their Web services. UDDI is used in the publicly accessible UDDI Business Registry (UBR) maintained by Microsoft, IBM, Hewlett-Packard and SAP. Companies also can implement UDDI in private registries, which are accessible only to authorized parties, such as a company’s employees, business partners and suppliers.
In June 2001, the UDDI project released a beta specification of UDDI Version 2.0, which improves on several features of the original specification. One enhancement offers advanced searching capabilities, such as the ability to search using the wildcard character (*). UDDI Version 2.0 also increases the global scope of UDDI registries in that businesses can provide company and product descriptions in various languages, such as Chinese and French. Another new feature enables an organization to provide data regarding its infrastructure, such as details about departments (sales, marketing, research and development, etc.), partners and affiliates. In addition, UDDI Version 2.0 offers support for industry-specific identifiers, such as those of the Standard Industrial Classification (SIC) system, which assigns unique numerical identifiers to industries. For example, 2621 represents Paper Mills, and 7371 represents Computer Programming Services.

At the time of this writing, more than 300 companies belong to the UDDI community. These companies, known as community members, are committed to the enhancement, evolution and world-wide acceptance of the UDDI registry. Community members include many large and influential organizations, such as American Express, Boeing, Fujitsu, Hitachi and Sun Microsystems. A complete list of community members can be found at www.uddi.org.

7.4.1 Operator Nodes and Registrars

An operator node is an organization that hosts an implementation of the UDDI Business Registry (UBR). Four operator nodes—Hewlett-Packard, IBM, Microsoft and SAP—host beta implementations of the UBR that adhere to the UDDI Version 2.0, and two operator nodes—IBM and Microsoft—host implementations of the UBR that adhere to the UDDI Version 1.0. A company needs to register with only one operator node to be listed in the UBR. This is because the UBR is based on the “register once, publish everywhere” principle, which states that information contained in one registry is replicated in the other registries. Replication is the process of updating records so that all instances of those records are identical. Thus, when a company registers with one operator node (known as a custodian), the company’s data appears in the other three registries, as well. Although data is not replicated instantaneously, the operator nodes synchronize their data at least every 12 hours.

At the time of this writing, only the UBR implementations that adhere to UDDI Version 1.0 support replication. However, the UBR implementations that adhere to UDDI Version 2.0 are expected to support replication by fall 2002. [Note: The UDDI project announced that NTT Communications of Tokyo, Japan, will become an operator node in 2002.]

Although replication ensures that all four UBRs contain identical information, a company can update its information only through its custodian. This is because the UDDI Version 2.0 API Specification does not provide a protocol for reconciling disparate or duplicate data. Limiting companies to interaction with only one operator node prevents users from entering multiple versions of data in different operator nodes.

Alternatively, companies can publish information in the UBR through a registrar—an organization that assists companies in creating data, such as business and service descriptions, to be stored in UDDI registries. Note that registrars are not operator nodes, because registrars do not host implementations of the UDDI registry. A complete list of the registrars can be found at www.uddi.org.
7.4.2 Advantages of Registering

Registering in the UBR offers advantages to both service providers and service consumers. For service providers, the UBR is an effective method of advertising Web services. Because the UBR can be accessed from anywhere, service providers gain global visibility, enabling them to communicate and form alliances with organizations located throughout the world. This kind of worldwide exposure helps service providers expand their markets.14

For service consumers, the UBR saves time and simplifies the process of using Web services. The UBR stores technical details about Web services, so service consumers do not have to spend time locating service-related information, such as how to communicate with a particular Web service. By using the UBR, service consumers can integrate their applications with remote services more quickly and efficiently.15

The UBR also can reduce costs for service providers and service consumers. Service providers can advertise their businesses and services for free, and service consumers can locate compatible Web services for free. Without the UBR, organizations might have to pay fees to advertise and find Web services. [Note: Some service providers listed in the UBR charge consumers that access their Web services.]16

7.5 Role of UDDI in Web Services

As stated previously, UDDI registries contain general and technical information about businesses and their Web services. Vendors often compare the UBR’s structure to that of a phone book. In this section, we discuss the components of the UBR’s phone-book structure—white pages, yellow pages and green pages.17 We overview the schema for the UDDI information model, which specifies the XML elements and attributes used to describe a Web service. We also explain the UDDI publishing and inquiry APIs, which define rules for posting and searching registry content, respectively.

The UBR mainly supports indirect discovery, because it is hosted by four intermediaries. However, UDDI also can support direct discovery in private registries. This is because private registries usually are implemented by specific organizations and describe only services offered by those organizations.

7.5.1 Levels of UDDI

The UBR can be categorized into white pages, yellow pages and green pages. The white pages contain general information about a company, such as its name, address, contact information and identifiers. Identifiers are values (alphabetic or numeric) that uniquely distinguish companies. Examples of identifiers are Dun & Bradstreet’s D-U-N-S® (Data Universal Numbering System) classifications, which are nine-digit numbers assigned to businesses.18

The yellow pages divide companies into various categories on the basis of their products or services. For example, a software company might be categorized under computer software or software engineering. The yellow pages allow registry users to search for companies or services that fit a particular category (such as sales, travel or books).19

The green pages contain technical information about a company’s products, services and Web services. This data allows a service client to bind (i.e., establish a communication channel) to a Web service, because the information defines how to invoke the service.20 The green pages usually include references to services’ WSDL documents, which contain information on how to interact with Web services.21
7.5.2 Information Models in UDDI

The UDDI Version 2.0 Data Structure Reference (available at www.uddi.org) stipulates that to transact business, a client company needs access to certain information about a provider’s Web service. This information, known collectively as the UDDI information model, includes the following five components: business information, business-service information, binding information, service-specification information and publisher-assertion information. In this section, we discuss the five components of the UDDI information model.

Each component of the UDDI information model resides within a data structure that consists of XML elements and attributes. These XML elements and attributes describe the components of the information model. The XML representation of the UDDI information model is used when interfacing with a UDDI registry. Because there are five components, there exist five interrelated data structures. Figure 7.2 illustrates the relationships among the UDDI information model’s five data-structure types.

The businessEntity structure encapsulates a business’s general information, such as its name, address and contact information. This structure references the businessService structure, which describes different types of services offered by the company. Technical information about these services resides in the bindingTemplate structure, which contains references to tModel structures. A tModel structure contains information on how to interact with the Web services. The publisherAssertions structure describes the relationships (e.g., partnerships) between two business entities.
Chapter

The *business information* component corresponds to the UDDI white and yellow pages in that it contains general data about a business and the products and services offered by that business. The business information resides in the *businessEntity* top-level structure, which categorizes businesses by their unique identifiers. In this context, a top-level structure encapsulates elements, attributes and other structures that describe a business and its Web services. Figure 7.3 summarizes the elements and attributes of the *businessEntity* structure.

In this structure, *businessKey* is a required attribute that uniquely identifies a business. The custodian assigns a unique identifier to each *businessEntity* structure upon registration. Unique identifiers are referred to as *Universally Unique Identifiers (UUIDs)* and usually consist of hexadecimal values. The *businessServices* element contains zero or more references to the descriptions of services offered by an organization. This element references the *businessService* structure.

The *business service information* component corresponds to the UDDI green pages in that it contains technical data about the products and services offered by a particular business. The business service information resides in the *businessService* structure. Figure 7.4 summarizes the elements of *businessService*. Each *businessService* structure is identified uniquely by two UUIDs—*serviceKey* and *businessKey*. The required *bindingTemplates* structure contains the technical information about a Web service.

The *bindingTemplates* structure includes zero or more references to the *bindingTemplate* structure, which contains the binding information.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>businessKey</td>
<td>A required attribute that contains a unique hexadecimal identifier for the business. The identifier is assigned by the custodian upon registration.</td>
</tr>
<tr>
<td>authorizedName</td>
<td>An optional attribute that contains the name of the person who published the information.</td>
</tr>
<tr>
<td>operator</td>
<td>An optional attribute that contains the name of the operator node with which the business registered.</td>
</tr>
<tr>
<td>discoveryURLs</td>
<td>An optional element that contains URLs to discovery documents.</td>
</tr>
<tr>
<td>name</td>
<td>A required element that contains the name of the business.</td>
</tr>
<tr>
<td>description</td>
<td>An optional element that contains a brief description of the business.</td>
</tr>
<tr>
<td>contacts</td>
<td>An optional element that contains the business’s contact information.</td>
</tr>
<tr>
<td>businessServices</td>
<td>An optional element that lists the services offered by the business.</td>
</tr>
<tr>
<td>identifierBag</td>
<td>An optional element that contains a list of unique identifiers (D-U-N-S® number, stock symbol, etc.) associated with the business.</td>
</tr>
<tr>
<td>categoryBag</td>
<td>An optional element that contains a list of industry, product or geographic classifications.</td>
</tr>
</tbody>
</table>

**Fig. 7.3** *businessEntity* attributes and child elements.
The binding information component also corresponds to the green pages in that it contains technical information pertaining to a Web service. This information specifies how clients can connect to a particular Web service. Figure 7.5 summarizes the elements and attributes of the **bindingTemplate** structure.

Attribute **bindingKey** is a UUID assigned to each **bindingTemplate** structure by the custodian. The **tModelInstanceDetails** structure contains references to one or more **tModelInstanceInfo** structures, which contain the elements and attributes that describe the service-specification information, or “blueprints,” of a Web service. Structure **tModelInstanceInfo** references the **tModel** structure.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>serviceKey</td>
<td>A required attribute that contains a unique, hexadecimal identifier for a</td>
</tr>
<tr>
<td>businessKey</td>
<td>An attribute that references the <strong>businessKey</strong> of the <strong>businessEntity</strong></td>
</tr>
<tr>
<td>name</td>
<td>A required element that contains the name(s) of a service.</td>
</tr>
<tr>
<td>description</td>
<td>An optional element that contains a brief description of a service.</td>
</tr>
<tr>
<td>bindingTemplates</td>
<td>A required structure that contains technical information about a service.</td>
</tr>
<tr>
<td>categoryBag</td>
<td>An optional element that contains a list of industry, product or geographic classifications.</td>
</tr>
</tbody>
</table>

**Fig. 7.4**  **businessService** attributes child elements.

The binding information component also corresponds to the green pages in that it contains technical information pertaining to a Web service. This information specifies how clients can connect to a particular Web service. Figure 7.5 summarizes the elements and attributes of the **bindingTemplate** structure.

Attribute **bindingKey** is a UUID assigned to each **bindingTemplate** structure by the custodian. The **tModelInstanceDetails** structure contains references to one or more **tModelInstanceInfo** structures, which contain the elements and attributes that describe the service-specification information, or “blueprints,” of a Web service. Structure **tModelInstanceInfo** references the **tModel** structure.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bindingKey</td>
<td>A required attribute that contains a unique hexadecimal identifier. The identifier is assigned by the operator node upon registration.</td>
</tr>
<tr>
<td>serviceKey</td>
<td>An attribute that references the <strong>serviceKey</strong> of the <strong>businessService</strong> element. This attribute is required if the <strong>bindingTemplate</strong> structure is not contained in a fully qualified parent that contains another <strong>serviceKey</strong>.</td>
</tr>
<tr>
<td>description</td>
<td>An optional element that contains brief description(s) of Web service(s).</td>
</tr>
<tr>
<td>accessPoint</td>
<td>An element that states where to access a Web service. Valid <strong>accessPoint</strong> types include <strong>mailto</strong>, <strong>http</strong>, <strong>https</strong>, <strong>ftp</strong>, <strong>fax</strong>, <strong>phone</strong> and <strong>other</strong>. This element is required if <strong>hostingRedirector</strong> is not specified.</td>
</tr>
</tbody>
</table>

**Fig. 7.5**  **bindingTemplate** attributes and child elements. (Part 1 of 2)
Whereas the binding information specifies where to access a Web service, the service-specification information component describes how to interact with the Web service. The service-specification information resides in the tModel structure, summarized in Fig. 7.6.

The tModel, or Service Type Registrations, structure contains information that allows service consumers to use a service provider’s Web service.22 In the tModel structure, the tModelKey is a UUID assigned to the structure. The custodian assigns the value to the tModelKey. Structure overviewDoc references documentation that provides information or instructions about the Web service. Usually, overviewDoc references WSDL documents, which contain technical (“blueprint”) information about Web services. This “blueprint” information includes the parameters that a Web service receives, the data formats it accepts (.dat,.txt, etc.) and other application-specific information. This information allows programmers to determine whether a Web service is compatible with their programs.

The publisher-assertion information component indicates a relationship between two companies. To instantiate the relationship, both parties must agree to the relationship by declaring identical assertions (i.e., identical statements that specify a mutual relationship). A relationship is valid only when both parties reciprocate the assertions. The publisher assertion information resides in the publisherAssertion structure. Figure 7.7 summarizes the elements of the publisherAssertion structure.

Publisher-assertion information, a new feature of Version 2.0, allows organizations to acknowledge parent-child, peer-peer and identity relationships. A parent-child relationship indicates that one organization owns another, smaller organization (i.e., the organization identified in the fromKey owns the organization identified in the toKey). A peer-peer relationship states that the organizations identified in the fromKey and the toKey are partners or affiliates. An identity relationship states that the organization identified in the fromKey is the same as the organization identified in the toKey. Identity relationships typically are used to assert an organization’s various divisions, units and departments.23

By providing publisher-assertion capabilities, UDDI allows large corporations to describe aspects of their businesses—such as divisions, departments, partners, affiliates and subsidiaries—to users of the UBR. This information is beneficial to service consumers who want to know how business A relates to business B before accessing the services of either business.
7.5.3 UDDI Publishing and Inquiry APIs

The UDDI Version 2.0 API Specification overviews the publishing and inquiry APIs for creating and searching registry content. The publishing API supports the publish operation, which enables companies to post and update information in the UDDI registry. Access to the publishing API is restricted, and the UDDI project requires that operator nodes implement an authentication protocol that verifies the identity of the individual or organization creating or updating the information. The publishing API consists of commands that service providers can use to create and update information. Access to the publishing API commands is available only via HTTPS (i.e., a variant of HTTP that uses Secure Sockets Layer to establish security).  

© 2002. Prentice Hall. All Rights Reserved.
Chapter

The *inquiry API* supports the *find* operation, which enables service consumers to browse the registry for service providers that offer a certain service or type of service. Anyone can use the inquiry API to perform queries on the UBR. The inquiry API supports three query patterns—browse, drill-down and invocation. The *browse pattern*, which supports the five information-model structures discussed in the previous section, allows service consumers to perform broad searches for businesses, services, templates or *tModels*. This query pattern returns the general, overview information (identification key, name and description) pertaining to the business, service, template or *tModel*.25

To obtain a more detailed description of a business, service, template or *tModel*, the inquiry API provides the *drill-down pattern*. This pattern typically is used in conjunction with the browse pattern, because it requires an identification key (obtained during the browse pattern) to retrieve the necessary information. The identification key is passed as an argument in a drill-down pattern. With the drill-down pattern, users can obtain technical information, such as integration capabilities and scalability.26

The *invocation pattern* queries the *bindingTemplate* structure, which contains information that programmers need to access a particular Web service. Because the location of a Web service might change, the invocation pattern searches the *bindingTemplate* structure for the service’s current location. The access information always resides in the *bindingTemplate* structure; therefore, service consumers typically use automated tools that query the structure for the access information.27

After service consumers discover compatible Web services, the consumers must connect to, and communicate with, the computing systems of other businesses. The process of connecting to, and communicating with, a Web service is referred to as binding.28

**WORKS CITED**

2. P. Korzeniowski, “UDDI: Two Versions Down, One to Go,” eWeek 4 February 2002 <www.eweek.com/article/0,3658,s%253D722%2526a%253D22050,00.asp>.
3. “SIC Division Structure,” <155.103.6.10/cgi-bin/sic/sicser5>.

© 2002. Prentice Hall. All Rights Reserved.