第5章 chapter 5

WSDL与UDDI

本章学习目标

- 理解WSDL的作用
- 熟悉WSDL的文档结构
- 掌握WSDL的文档元素
- 了解UDDI的作用
- 了解UDDI的实现机制
- 了解UDDI的数据结构
- 了解UDDI的API
- 理解WSDL到UDDI数据结构的映射

5.1 WSDL概述

WSDL(Web Service Description Language,Web服务描述语言)是一种用来描述Web服务功能特征的语言。2001年3月,WSDL 1.1被IBM、Microsoft公司作为一个W3C记录(W3C note)提交到有关XML协议的W3C XML活动,用于描述网络服务。2002年7月,W3C发布了第一个WSDL 1.2工作草案。WSDL描述了Web服务的接口和语义等信息,Web服务的使用者可以通过WSDL了解这个Web服务支持哪些功能调用,以及如何调用这些功能,进而可以向这个Web服务发送SOAP(或其他协议类型)请求,最终使用这个Web服务。

5.1.1 WSDL的基本概念

Web服务是一种定义在Web上的对象,Web服务的开发者需要对服务的调用方式进行某种结构化的说明,以便服务的调用者能够正确地使用这些服务。WSDL就是专门用来描述Web服务的一种语言,其规定了一套基于XML的语法,能够提供以下关于Web服务的4个方面的重要信息。

- 描述服务功能的信息；
- 描述这些功能的传入(请求)和传出(响应)消息的类型信息；
- 描述服务的协议绑定信息；
- 描述用户查找特定服务的地址信息。

WSDL将Web服务定义为端口的集合，一个端口代表一个服务访问点。WSDL把服务访问点和消息的抽象化描述与具体的服务部署和数据格式的绑定分离，从而使对服务的抽象定义可以方便地重用。

WSDL文档包含以下8个关键的构成元素。

1. ＜definitions＞

＜definitions＞元素是WSDL文档的根元素，用来定义Web服务的名称，并声明WSDL中使用的命名空间。

2. ＜types＞

＜types＞元素用于描述Web服务与调用者之间传递消息时所使用的数据类型。WSDL支持任何类型的系统，默认采用XML Schema类型系统。

3. ＜message＞

＜message＞元素是Web服务与调用者之间传递的消息的逻辑定义。一个消息可能包含多个部分，每一部分用＜part＞元素表示，可以使用＜types＞元素中定义的数据类型来定义每个＜part＞元素类型。

4. ＜operation＞

＜operation＞元素是Web服务中所支持的操作的抽象定义。通常一个＜operation＞元素描述一个服务访问点的请求/响应消息对。

5. ＜portType＞

＜portType＞元素是某个访问点所支持的所有操作的抽象定义。

6. ＜binding＞

＜binding＞元素定义了特定＜portType＞元素定义的操作和消息的格式、协议之间的绑定。

7. ＜port＞

＜port＞元素定义了Web服务的绑定地址。

8. ＜service＞

＜service＞元素描述了相关的服务访问点的集合。

其中，<types>元素、<message>元素、<operation>元素和<protType>元素描述了调用Web服务的抽象定义，它们与具体Web服务部署细节无关，这些抽象定义是可以重用的，相当于IDL描述的对象接口标准。但是这些抽象定义的对象到底用哪种语言实现，遵从什么平台的细节规范，被部署在什么机器上则是由<binding>元素、<port>元素和<service>元素所描述的。

<service>元素描述的是服务所提供的所有访问入口的部署细节。一个服务可以包含多个服务访问入口<port>元素(<port>元素描述的是一个服务访问入口的部署细节，port=url+bingding)。调用模式则使用<binding>元素来表示，<binding>元素定义了某个<portType>元素与某一种具体的网络传输或消息传输协议的绑定(binding=portType+具体传输协议和数据格式规范)。在这一层中，描述的内容就与具体服务的部署相关了。

WSDL被设计成与语言和平台无关的一种描述语言，其可被用于描述任何语言实现的、部署在任何平台上的Web服务。

5.1.2 一个简单的WSDL实例

【例5.1】 一个提供IP地址来源的Web服务的WSDL文档，其代码如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions xmlns:wsdl="http://xxxs.xmlsoap.org/wsdl/" targetNamespace=
"http://WebXml.com.cn/" xmlns:xsd="http://www.wxxx3.org/2001/XMLSchema"
xmlns:tns="http://Wxxxl.com.cn/" xmlns:soap="http://sxxxs.xmlsoap.org/wsdl/
soap/">
<!--这部分是服务的抽象定义，包括对<types>、<message>、<operation>和<portType>等
元素的定义-->
    <wsdl:types>
        <xsd:schema targetNamespace="http://Wxxxl.com.cn/" elementFormDefault=
        "qualified">
            <xsd:element name="getCountryCityByIp">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="theIpAddress" type="xsd:string"
                        maxOccurs="1" minOccurs="0"/>
                    </xsd:sequence>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="getCountryCityByIpResponse">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="getCountryCityByIpResult" type=
                        "tns:ArrayOfString" maxOccurs="1" minOccurs="0"/>
                    </xsd:sequence>
                </xsd:complexType>
            </xsd:element>
```

```
            <xsd:complexType name="ArrayOfString">
                <xsd:sequence>
                    <xsd:element name="string" type="xsd:string" maxOccurs=
                    "unbounded" minOccurs="0" nillable="true"/>
                </xsd:sequence>
            </xsd:complexType>
            <xsd:element name="ArrayOfString" type="tns:ArrayOfString"
            nillable="true"/>
            <xsd:element name="string" type="xsd:string" nillable="true"/>
        </xsd:schema>
    </wsdl:types>
    <wsdl:message name="getCountryCityByIpSoapIn">
        <wsdl:part name="parameters" element="tns:getCountryCityByIp"/>
    </wsdl:message>
    <wsdl:message name="getCountryCityByIpSoapOut">
        <wsdl:part name="parameters"
        element="tns:getCountryCityByIpResponse"/>
    </wsdl:message>
    <wsdl:portType name="IpAddressSearchWebServiceSoap">
        <wsdl:operation name="getCountryCityByIp">
            <wsdl:input message="tns:getCountryCityByIpSoapIn"/>
            <wsdl:output message="tns:getCountryCityByIpSoapOut"/>
        </wsdl:operation>
    </wsdl:portType>
<!--这部分将服务的抽象定义与 SOAP/HTTP 绑定,描述如何通过 SOAP/HTTP 来访问前面描述
的访问入口。其中规定了在具体 SOAP 调用时,应当使用的 soapAction 是 http://Wxxxl.
com.cn/getCountryCityByIp-->
    <wsdl:binding name="IpAddressSearchWebServiceSoap" type="tns:
    IpAddressSearchWebServiceSoap">
        <soap:binding transport="http://sxxxs.xmlsoap.org/soap/http"/>
        <wsdl:operation name="getCountryCityByIp">
            <soap:operation style="document" soapAction="http://Wxxxl
            .com.cn/getCountryCityByIp"/>
            <wsdl:input>
                <soap:body use="literal"/>
            </wsdl:input>
            <wsdl:output>
                <soap:body use="literal"/>
            </wsdl:output>
        </wsdl:operation>
    </wsdl:binding>
<!--这部分是具体的 Web 服务的定义。在这个名为 IpAddressSearchWebService 的 Web 服
务中,提供了一个服务访问入口,访问的地址是“http://www.wxxxl.com.cn/WebServices/
IpAddressSearchWebService”,使用的消息模式是由前面的<binding>元素所定义的,访问
入口和<binding>元素一起构成一个端口。-->
    <wsdl:service name="IpAddressSearchWebService">
        <wsdl:port name="IpAddressSearchWebServiceSoap"
        binding="tns:IpAddressSearchWebServiceSoap">
            <soap:address location="http://www.wxxxl.com.cn/WebServices/
```

```
            IpAddressSearchWebService "/>
        </wsdl:port>
      </wsdl:service>
</wsdl:definitions>
```

5.2 WSDL文档结构

WSDL是一种用于描述Web服务的XML语言，它以一种结构化的方式将Web服务描述为一组对消息进行操作的网络端点，并将服务定义为网络终端或端口的集合。在WSDL中，抽象定义与具体的网络部署或数据绑定是分开的，这样就可以重用抽象定义。WSDL没有引入新的类型定义语言，而是把XML Schema当作它的类型系统。另外，WSDL也允许通过扩展使用其他类型定义语言。

在WSDL文档中，<types>元素、<message>元素、<part>元素、<operation>元素和<protType>元素描述了Web服务的抽象接口。<protType>元素本质上是一个抽象接口(类似于Java接口定义)，由<operation>元素和<message>元素定义组成。每一个<message>元素定义描述了消息的有效信息，这些消息既可以是由Web服务向外发送的消息，也可以是它所接收的消息。消息由<part>元素表示了一个类型的实例。通过<protType>元素可以声明<operation>元素，每一个<operation>元素都包含了许多<message>元素定义，这些<message>元素定义描述了它的输入输出参数以及任何出错的情况。其具体关系如图5.1所示。

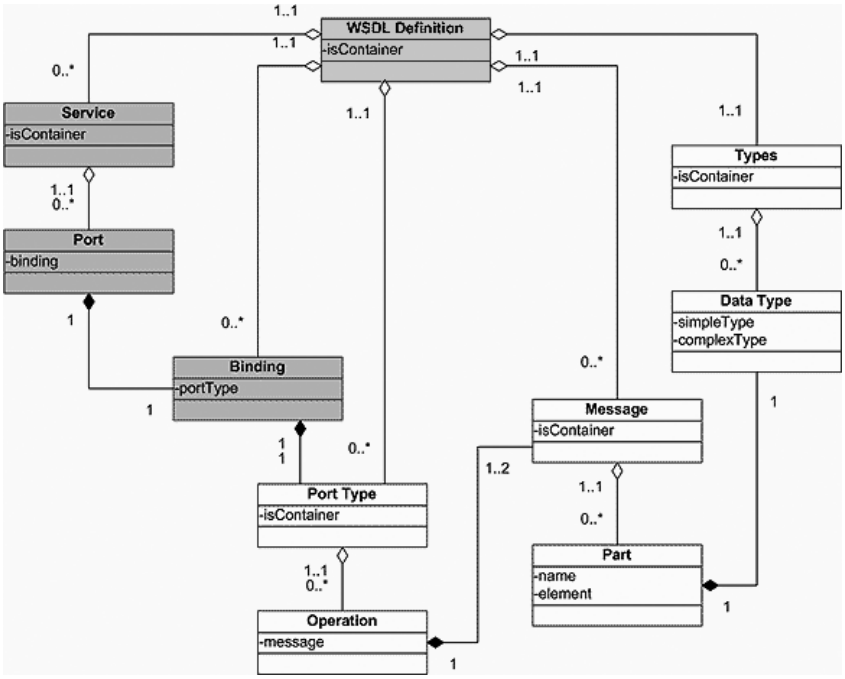

图5.1 WSDL文档结构中主要元素之间的关系

1. <definitions>

<definitions>元素用来定义 WSDL 文档的名称，并且引入需要的 XML 命名空间。下面的代码是一个<definitions>元素的结构：

```
<wsdl:definitions xmlns:wsdl="http://sxxxs.xmlsoap.org/wsdl/" targetNamespace=
"http://Wxxxl. com. cn/"  xmlns: xsd =" http://www. wxxx3. org/2001/XMLSchema "
xmlns:tns="http://Wxxxl.com.cn/" xmlns:soap="http://sxxxs.xmlsoap.org/wsdl/
soap/">
```

上述代码中声明了 3 个必需的命名空间 WSDL、SOAP 和 XSD；targetNamespace 指定了 WSDL 文档中出现的新元素与属性的命名空间，xmlns:tns=" http://Wxxxl.com.cn/"指定了这个命名空间的前缀为 tns。

2. <types>

<types>(类型)元素规定了与消息相关的数据类型的定义。为了获得最大程度的平台中立性，WSDL 使用 XML Schema 语法来定义数据类型。这些数据类型用来定义 Web 服务方法的参数和返回值。下面的代码是一个<types>元素的结构：

```
<wsdl:types>
    <xsd:schema targetNamespace="http://Wxxxl.com.cn/" elementFormDefault=
    "qualified">
        <xsd:element name="getCountryCityByIp">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="theIpAddress" type="xsd:string"
                    maxOccurs="1" minOccurs="0"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="getCountryCityByIpResponse">
            <xsd:complexType>
                <xsd:sequence>
                     <xsd:element name="getCountryCityByIpResult" type="tns:
                     ArrayOfString" maxOccurs="1" minOccurs="0"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:complexType name="ArrayOfString">
            <xsd:sequence>
                <xsd:element name="string" type="xsd:string" maxOccurs=
                "unbounded" minOccurs="0" nillable="true"/>
            </xsd:sequence>
```

```
        </xsd:complexType>
        <xsd:element name="ArrayOfString" type="tns:ArrayOfString"
        nillable="true"/>
        <xsd:element name="string" type="xsd:string" nillable="true"/>
    </xsd:schema>
</wsdl:types>
```

上述代码中,使用<types>元素定义了 getCountryCityByIp 和 getCountryCityByIpResponse 两个复杂类型,其中 getCountryCityByIp 包含一个子元素 theIpAddress,表示要查询的 IP 地址的具体信息;getCountryCityByIpResponse 包含一个子元素 getCountryCityByIpResult,表示得到的 IP 的城市信息。

3. <message>

<message>(消息)元素定义了传递的消息的数据结构。一个<message>元素由一个或多个<part>(消息片段)元素构成,<part>元素可以使用<types>元素中定义的元素类型。下面的代码是一个<message>元素的结构:

```
<wsdl:message name="getCountryCityByIpSoapIn">
    <wsdl:part name="parameters" element="tns:getCountryCityByIp"/>
</wsdl:message>
<wsdl:message name="getCountryCityByIpSoapOut">
    <wsdl:part name="parameters" element="tns:getCountryCityByIpResponse"/>
</wsdl:message>
```

上述代码中,定义了 getCountryCityByIpSoapIn 和 getCountryCityByIpSoapOut 两个<message>元素,分别代表后续的 getCountryCityByIp 操作的输入和输出消息。<message>元素包含的<part>元素是操作的参数和返回值的名称和类型,即 getCountryCityByIpSoapIn 消息定义了 getCountryCityByIp 操作的输入参数名为 parameters,类型为 getCountryCityByIp;getCountryCityByIpSoapOut 消息定义了 getCountryCityByIp 操作的返回值名称为 parameters,类型为 getCountryCityByIpResponse。如果操作使用多个参数或多个返回值,则<message>元素中可以定义多个<part>元素。

4. <portType>

<portType>(端口类型)元素定义了 Web 服务的抽象接口。该接口类似 Java 的接口,都是定义了一个抽象类型和方法,没有定义实现。一个<portType>中可以定义多个<operation>,一个<operation>可以看作是一个方法。下面的代码是一个<portType>元素的结构:

```
<wsdl:portType name="IpAddressSearchWebServiceSoap">
    <wsdl:operation name="getCountryCityByIp">
        <wsdl:input message="tns:getCountryCityByIpSoapIn"/>
```

```
        <wsdl:output message="tns:getCountryCityByIpSoapOut"/>
    </wsdl:operation>
</wsdl:portType>
```

上述代码中<portType>元素定义了服务的调用模式的类型，这里包含一个操作getCountryCityByIp方法，同时包含<input>元素和<output>元素表明该操作是一个请求/响应模式，请求消息是前面定义的getCountryCityByIpSoapIn消息，响应消息是前面定义的getCountryCityByIpSoapOut消息。

WSDL支持以下4种类型的操作。

One-way：单向操作，此类型的操作可以接收消息，但不会返回响应，通常由一个<input>元素定义。

Request-response：请求/响应操作，此类型的操作可以接收请求消息并返回响应消息，通常由一个<input>元素和一个<output>元素定义，还可以包含一个可选的<fault>元素，用于定义错误消息的抽象数据格式。

Solicit-response：恳求/响应操作，此类型的操作可以接收请求消息并接收响应消息，通常由一个<input>元素和一个<output>元素定义，还可以包含一个可选的<fault>元素，用于定义错误消息的抽象数据格式。

Notification：通知操作，此类型的操作可以发送消息，通常由一个<output>元素定义。

5. <binding>

<binding>（绑定）元素将一个抽象的<portType>元素映射到一组具体的协议（SOAP或者HTTP）、消息传递样式（RPC或者document）以及编码样式（literal或者SOAP encoding）。<binding>元素有两个属性：name属性和type属性。name属性定义<binding>元素的名称，type属性指向用于<binding>元素的端口<types>元素、<message>元素和<portType>元素标签处理抽象的数据内容，而<binding>元素标签把前3部分的抽象定义具体化。下面的代码是一个<binding>元素的结构：

```
<wsdl:binding name="IpAddressSearchWebServiceSoap" type="tns:
IpAddressSearchWebServiceSoap">
    <soap:binding transport="http://sxxxs.xmlsoap.org/soap/http"/>
    <wsdl:operation name="getCountryCityByIp">
        <soap:operation style="document" soapAction="http://Wxxxl.com.cn/
        getCountryCityByIp"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
```

在上述代码中，使用了SOAP命名空间下的<soap:binding>、<soap:operation>和<soap:body>这3个元素完成了绑定。

<soap:binding>元素指定了用于传输SOAP消息的Internet协议以及operation缺省的消息类型(RPC还是文档类型)，transport="http://sxxxs.xmlsoap.org/soap/http"表示采用的是HTTP的传输方式，不同的值代表不同的传输方式，如HTTPS、SMTP、FTP等。

style="document"表示消息类型是面向文档(document)的。在面向RPC的通信中，消息包含参数和返回值；在面向文档的通信中，消息包含文档，文档的格式没有限制，通常应该是消息的发送者和接收者达成一致的XML文档。

<soap:operation>元素指定了消息传递样式以及SOAPAction字段的值。

<soap:body>元素有四个属性use、namespace、part和encodingStyle，对于WS-I use的属性值必须是literal，意味着不是用编码的方式，所以永远不会用到encodingStyle属性，在RPC样式中，必须用一个有效的URI指定的namespace属性。此URI可以与WSDL文档的targetNampspce相同；而在document样式中不能使用namespace，XML文档样式的命名空间派生于它的XML文档。

6. <service>

<service>元素包含一个或者多个<port>元素，其中每个<port>元素表示一个不同的Web服务。<port>元素将URL赋给一个特定的<binding>元素，甚至可以使两个或者多个<port>元素将不同的URL赋值给相同的<binding>元素。下面的代码是一个<service>元素的结构：

```
<wsdl:service name="IpAddressSearchWebService">
    <wsdl:port name="IpAddressSearchWebServiceSoap" binding="tns:
    IpAddressSearchWebServiceSoap">
        <soap:address location="http://www.wxxxl.com.cn/WebServices/
        IpAddressSearchWebService "/>
    </wsdl:port>
</wsdl:service>
```

使用<port>元素定义了一个地址，因为在<binding>元素中使用了SOAP的绑定方式，所有<port>元素中使用元素<soap:address location="http://www.wxxxl.com.cn/WebServices/IpAddressSearchWebService"/>指定了服务的地址为“http://www.wxxxl.com.cn/WebServices/IpAddressSearchWebService”。

5.3 WSDL 绑 定

在WSDL中，绑定指的是将协议、数据格式与<message>元素、<operation>元素、<portType>元素等抽象实体进行关联的过程。WSDL支持的4种操作类型(单向、请

求/响应、恳求/响应、通知)都是抽象的定义,而绑定描述了这些抽象概念的具体关联。因此,必须为特定的操作类型定义绑定才能够成功执行该类型的操作。

WSDL允许在WSDL命名空间定义的元素中使用用户自定义的元素,这些元素称为扩展元素。扩展元素为WSDL提供了强大的扩展机制,WSDL规范中定义了3种绑定扩展:SOAP绑定、HTTP GET POST绑定和MIME绑定,其中SOAP绑定是最常见的一种方式。

WSDL1.1针对SOAP1.1定义了绑定扩展,在例5.1中一个提供IP地址来源的Web服务的WSDL文档中就使用了SOAP绑定,关于SOAP绑定的代码如下:

```
<wsdl:binding name="IpAddressSearchWebServiceSoap" type="tns:
IpAddressSearchWebServiceSoap">
    <soap:binding transport="http://sxxxs.xmlsoap.org/soap/http"/>
    <wsdl:operation name="getCountryCityByIp">
        <soap:operation style="document" soapAction="http://Wxxxl.com.cn/
        getCountryCityByIp"/>
        <wsdl:input>
            <soap:body use="literal"/>
        </wsdl:input>
        <wsdl:output>
            <soap:body use="literal"/>
        </wsdl:output>
    </wsdl:operation>
</wsdl:binding>
```

上述代码中,使用了SOAP命名空间的<soap:binding>、<soap:operation>、<soap:body>这3个元素完成了绑定。

<soap:binding>元素规定了采用SOAP协议格式的绑定,该元素不提供有关消息格式或编码的信息,但是只要服务使用SOAP绑定,就必须声明此元素。

<soap:binding>元素的style属性指定了绑定的操作是面向RPC还是面向文档(document)的。在面向RPC的通信中,消息包含参数和返回值;在面向文档的通信中,消息包含文档,文档的格式没有限制,通常应该是消息的发送者和接收者达成一致的XML文档。

<soap:binding>元素的transport属性指定了SOAP的传输类型,如上述代码中的transport属性值http://sxxxs.xmlsoap.org/soap/http代表了HTTP传输。不同的值可以代表不同类型的传输,如SMTP、FTP等。

<soap:operation>元素定义了SOAP操作,其中soapAction属性指定了SOAP消息报头值。

<soap:body>元素定义了抽象的<part>元素在SOAP消息中的具体外观。其中use属性指定了消息片段使用特定的编码规则还是已定义的具体模式,对应值为encoded和literal。