WSDL: Web Service Description Language

Outline

- What and why
- WSDL document structure
- Sections and elements
  - types, messages, portTypes, bindings, and services
- Namespaces
- WSDL references

What is WSDL?

- Web Service Description Language
- WSDL is a document written in XML
- The document describes a Web service
- Specifies the location of the service and the methods the service exposes

Why WSDL?

- Without WSDL, calling syntax must be determined from documentation that must be provided, or from examining wire messages
- With WSDL, the generation of proxies for Web services is automated in a truly language- and platform-independent way

Where does WSDL fit?

- SOAP is the envelope containing the message
- WSDL describes the service
- UDDI is a listing of web services described by WSDL

Document Structure

- Written in XML
- Two types of sections
  - Abstract and Concrete
- *Abstract* sections define SOAP messages in a platform- and language-independent manner
- Site-specific matters such as serialization are relegated to the *Concrete* sections

Abstract Definitions

- **Types:** Machine- and language-independent type definitions.
- **Messages:** Contains function parameters (inputs are separate from outputs) or document descriptions.
- **PortTypes:** Refers to message definitions in Messages section that describe function signatures (operation name, input parameters, output parameters).

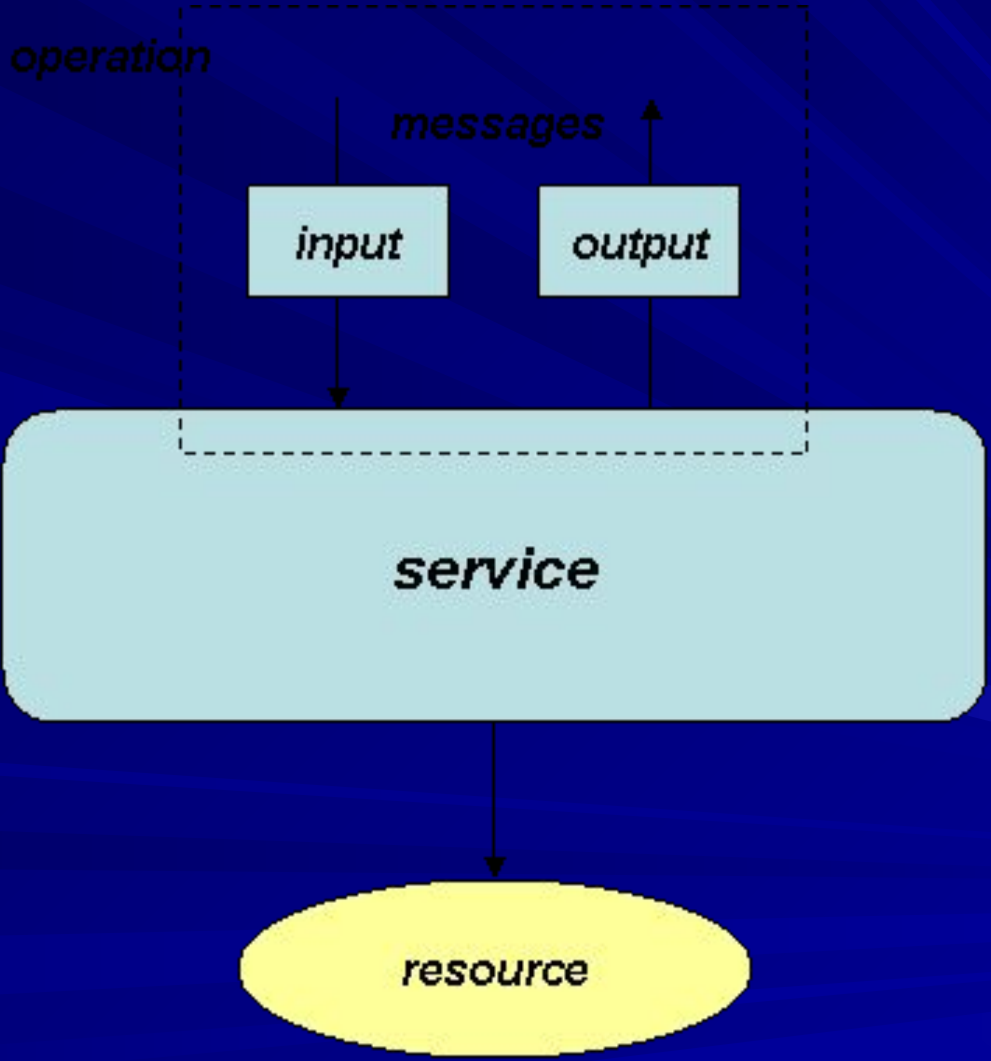
Concrete Descriptions

- **Bindings:** Specifies binding(s) of each operation in the PortTypes section.
- **Services:** Specifies port address(es) of each binding.

Operation

- An *operation* is similar to a function in a high level programming language
- A message exchange is also referred to as an operation
- Operations are the focal point of interacting with the service

Big Picture



An Example

- <?xml version="1.0" encoding="UTF-8" ?>
- This first line declares the document as an XML document.
- Not required, but helps the XML parser determine whether to parse the file or signal an error

Types Section

- The *type* element defines the data types that are used by the web service.
-
```
<xsd:complexType name="PERSON">
   <xsd:sequence>
 <xsd:element name="firstName“ type="xsd:string"/>
 <xsd:element name="lastName" type="xsd:string"/>
<xsd:element name="ageInYears" type="xsd:int"/>
   </xsd:sequence>
</xsd:complexType>
```

Messages Section

- A *message* element defines parameters
- The name of an output message element ends in "Response" by convention

-
  ```
  <message name="Simple.foo">
  <part name="arg" type="xsd:int"/>
  </message>

  <message name="Simple.fooResponse">
  <part name="result" type="xsd:int"/>
  </message>
  ```

PortTypes Section

- Defines a web service, the operations that can be performed, and the messages that are involved.
- <portType name="SimplePortType">
  <operation name="foo" parameterOrder="arg" >
  <input message="wsdlns:Simple.foo"/>
  <outputmessage="wsdlns:Simple.fooResponse"/>
  </operation>
  </portType>

Bindings Section

- The *binding* element defines the message format and protocol details for each port.
-
```
<operation name="foo">
<soap:operation soapAction="http://tempuri.org/action/Simple.foo"/>
<input>
<soap:body use="encoded"
namespace="http://tempuri.org/message/"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</input>
<output>
<soap:body use="encoded“
namespace="http://tempuri.org/message/“
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" />
</output>
</operation>
```

The Port Element

- Each <port> element associates a location with a <binding> in a one-to-one fashion
- <port name="fooSamplePort" binding="fooSampleBinding"> <soap:address location="http://carlos:8080/fooService/foo.asp"/> </port>

Services Section

- A collection of related endpoints, where an endpoint is defined as a combination of a binding and an address
- <service name="FOOSAMPLEService">
  <port name="SimplePort“
  binding="wsdlns:SimpleBinding">
  <soap:address
  location="http://carlos:8080/FooSample/
  FooSample.asp"/>
  </port>
  </service>

An Example

-
  ```
  <message name="Simple.foo">
  <part name="arg" type="xsd:int"/>
  </message>
  <message name="Simple.fooResponse">
  <part name="result" type="xsd:int"/>
  </message>
  <portType name="SimplePortType">
  <operation name="foo" parameterOrder="arg" >
  <input message="wsdlns:Simple.foo"/>
  <output message="wsdlns:Simple.fooResponse"/>
  </operation>
  </portType>
  ```
- The above describes what kind of C/C++ function call?
- int foo(int arg);

Namespaces

- The purpose of namespaces is to avoid naming conflicts.
- Imagine two complimentary web services, named A and B, each with an element named “foo”.
- Each instance of foo can be referenced as A:foo and B:foo
- Example: "xmlns:xsd" defines a shorthand (xsd) for the namespace
- See http://www.w3.org/2001/XMLSchema.

WSDL References [Primary]

- http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwebsrv/html/wsdlexplained.asp

  -a good overview of WSDL
- http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwebsrv/html/understandWSDL.asp

  -another good WSDL description

WSDL References [Secondary]

- http://www.xmethods.com/ve2/Tools.po

  -WSDL analyzer
- http://soap.amazon.com/schemas2/AmazonWebServices.wsdl

  -Amazon’s WSDL document
- http://api.google.com/GoogleSearch.wsdl

  -Google’s WSDL document