The Web Service Concept, Architecture & Surrounding Technologies

Lecture 5

University of Technology, Mauritius
Service Oriented Architecture
WAT 5101C

G.Suddul

Outline

- Introduction
- What are Web Services
- Browser interaction v/s. Web Services Interaction
- Web Service Promise
- Core Web Services Standards:
  - SOAP
  - WSDL
  - XML
  - UDDI
- Tool & Vendors
- Who Manages the Web Services Specifications?
  - W3C
  - OASIS
  - WS-I

Introduction

- Web services represent a new architecture for creating applications that can be accessed from a different computer.
- This lecture will improve your understanding of the topic at an executive summary level, covering the following:
  - The definition of Web services
  - The promise of Web services
  - The key specifications that comprise Web services
  - The tools that are commonly used to create Web services
  - The challenges that face Web services developers
  - The specifications that defines Web services

3

What are Web Services?

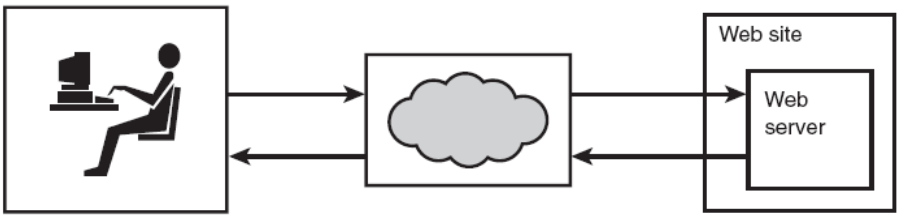
- A Web service is a software application that can be accessed remotely using different *XML-based languages.*
- Normally, a Web service is identified by a URL, just like any other Web site.
- What makes Web services different from ordinary Web sites is the type of interaction that they can provide.

4

Basic Request/Response

- Most Web sites are designed to provide a response to a request from a user. (see diagram on next slide)
  - The user either types in the URL of the site or clicks on a hyperlink to create the request.
  - This request takes the form of a text document that contains some fairly simple instructions for the server.
  - These instructions are limited to the name of a document to be returned or a call to a server-side program, along with a few parameters.

Basic Request/Response



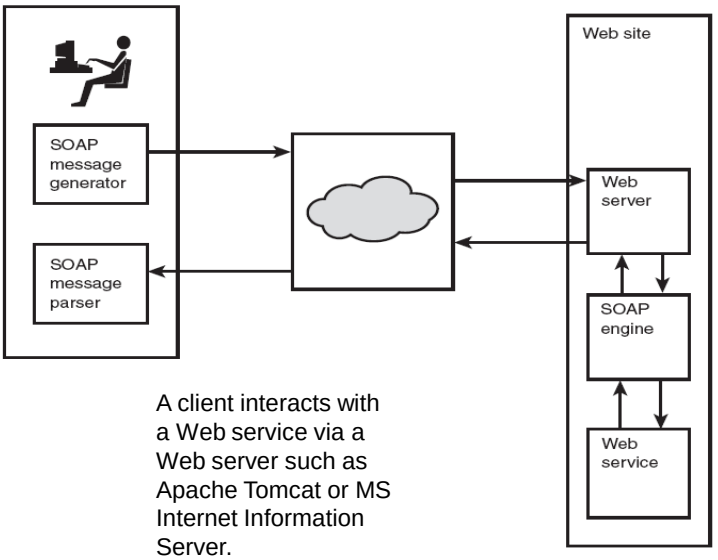
A browser interacts with a Web server to make requests.

Web Service Interaction

- The difference lies in the content of what is sent in the request from the client to the service.
- The Web Service client sends an XML document.
- The format of this XML document is defined by the SOAP specification.
- A SOAP message can contain method call with its parameters, and it is more complex that a browser interaction.

7

Web Service Client Interaction



A client interacts with a Web service via a Web server such as Apache Tomcat or MS Internet Information Server.

8

The Web Service's Promise

- The promise of **interoperability**.
- Web Services architecture is based on sending XML messages in a specific SOAP format.
- XML can be represented as plain *ASCII characters,* which can be transferred easily from computer to computer.
  - The implications of the above are significant.

Interoperability means

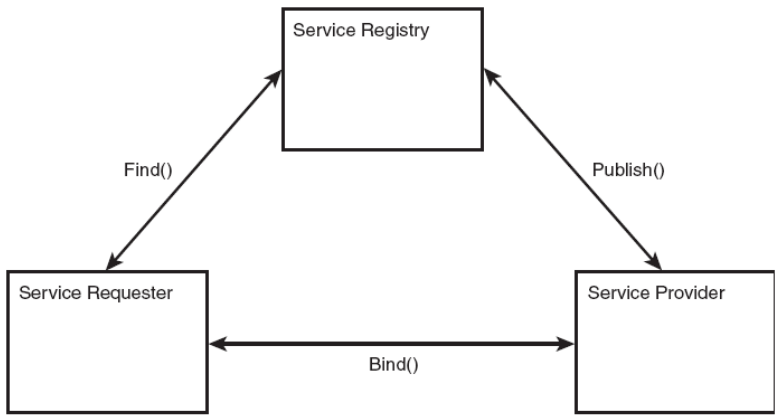
- It doesn't matter what kind of computer sends the SOAP message or on what operating system it is running.
- It doesn't matter where in the world the client is sending the message from.
- It doesn't matter what language the software that the client is running on was written in.
- There is no need for the client to know what type of SOAP processor is running on the server.

The Holy Grail of Computing …

- Web Services are also considered the Holy Grail of computing:
  - Every software application in the world can potentially talk to every other software application in the world.
  - This communication can take place across all the old boundaries of location, operating system, language, protocol, and so on.

Web Services Architecture

- It is a simple model that contains three entities and three operations.



Core Web Services Standards: XML

- Extensible Markup Language (XML) is the language that all the Web services Languages are built on.
- XML is a tool for constructing self-describing documents:
  - Create grammars that describe in XML schemas what tags are allowed and the relationships between the elements defined by these tags.
- SOAP, WSDL, and UDDI are all XML-based grammars defined for Web Services.

13

Core Web Services Standards: SOAP

- Originally stood for Simple Object Access Protocol (no longer, since 1.2).
- It is a specification that defines an XML grammar for both sending messages and responding to messages.
- The goal of SOAP is to describe a message format that is not bound to any hardware or software architecture.
  (*one that carries a message from any platform to any other platform in an unambiguous fashion*).

14

Brief Overview of SOAP

- The SOAP standard contains two parts:
  - Header
    - carries processing instructions
  - Body that contains the ***payload***
    - Two types of soap documents (XML documents and RPCs)
    - XML documents
      - Payload contains data that is moved from one computer to another.
    - RPC – Remote procedure Call
      - Payload contains a method call that is intended to be executed on another computer.

15

The SOAP Grammar

- The two strongest features of SOAP are:
  - its simplicity, and
  - the fact that everyone has agreed to use it.
- A SOAP message is composed of two mandatory parts:
  1) The SOAP envelope.
  2) The SOAP body.
- It is also composed of one optional part:
  3) The SOAP header.

16

Analogy with a physical letter

- Think of a SOAP message as an actual letter:
- **The envelope:**
  - defines what the message is (an envelope in your mail box contains a letter).
- **Header:**
  - to know who this message is for (usually have the senders details as well).
- **Letter:**
  - Encoded Message.
- Likewise, when an object receives a SOAP envelope, it expects an encoded XML letter.

17

Skeleton of a SOAP Message

```
<?xml version="1.0"?>
<soap:Envelope
  xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
  soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

    <soap:Header>
      ...
      ...
    </soap:Header>

    <soap:Body>
      ...
            <soap:Fault>
              ...
              ...
            </soap:Fault>

    </soap:Body>

</soap:Envelope>
```

18

The SOAP Envelope

- The required SOAP Envelope element is the root element of a SOAP message.
  - It defines the XML document as a SOAP message.
  - Note the use of the ***xmlns:soap*** namespace, which points to: http://www.w3.org/2001/12/soap-envelope.
  - The SOAP **encodingStyle** attribute is used to define the data types used in the document.
    - This attribute may appear on any SOAP element, and it will apply to that element's contents and all child elements.
    - A SOAP message has no default encoding.

Headers

- There are two different types of headers SOAP uses for its messages.
  - The first is a standard **HTTP header.**
    - Also used for retrieving or sending HTML pages.
    - This is appropriate as SOAP messages are also relayed via HTTP.
  - The second one is the **SOAP header.**
- **Example of an HTTP Header** (POST):

  POST /MyWebService HTTP/1.1
  Host: MyHost
  Content Type: text/xml; charset="utf-8"
  Content-length: xxx

HTTP-POST

- The HTTP header (described previously) is typical when using an HTTP-POST to send a SOAP message.
- The server would respond with a header as follows:

  HTTP/1.1 200 OK

  Content-Type: text/xml; charset="utf-8"

  Content-Length: xxx

21

SOAP HTTP Binding

- A SOAP method is an HTTP request/response that complies with the SOAP encoding rules.
  - HTTP + XML = SOAP
- A SOAP request could be an HTTP POST or an HTTP GET request.
- The HTTP POST request specifies at least two HTTP headers:
  - Content-Type.
  - Content-Length.

22

Content-Type

- The Content-Type header for a SOAP request and response defines the MIME type for the message and the character encoding (optional) used for the XML body of the request or response.

- **Syntax:**
  - Content-Type: MIMEType; charset=character-encoding

- **Example:**

  POST /item HTTP/1.1

  Content-Type: application/soap+xml; charset=utf-8

23

Content-Length

- The Content-Length header for a SOAP request and response specifies the number of bytes in the body of the request or response

- **Syntax:**
  - Content-Length: bytes

- **Example:**

  POST /item HTTP/1.1

  Content-Type: application/soap+xml; charset=utf-8

  Content-Length: 250

24

SOAP Header

POST /MyWebService HTTP/1.1
Host: MyHostContent Type: text/xml; charset=”utf-8”
Content-length: xxx…

<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Header>

<m:Trans xmlns:m="http://www. example.com/transaction/“
soap:mustUnderstand="1">234</m:Trans>

</soap:Header>
…
</soap:Envelope>

SOAP Header

- Is used to convey additional information that is not included in the body of the message or HTTP header.

- Example:

<SOAP:Envelope>
<SOAP:Header mustUnderstand>
<authHeader>
password
<authHeader>
<SOAP:Header>

<SOAP:BODY>
…

SOAP Header

- SOAP header tag goes inside the envelope.
- In previous example:
  - Header has a ***authHeader*** element, which supplies a password.
  - This password can be used by the receiving object to verify the sender's object permissions.
  - ***mustUnderstand*** attribute tells the receiving object that it must process the header(it cannot ignore it)

27

SOAP Body

- The SOAP body contains the xml message intended for the ultimate end-point.
- The example below requests the price of apples.
  - *Note that the m:GetPrice and the Item elements above are application-specific elements. They are not a part of the SOAP standard.*

```
<soap:Body>
  <m:GetPrice xmlns:m="http://www.exmaple.com/prices">
    <m:Item>Apples</m:Item>
  </m:GetPrice>
</soap:Body>
```

28

A SOAP Request Explained

- Inside the <SOAP:Body>:
  - Element named *GetPrice*
  - This could be the name of the method that calculates the of an article.
  - In this case: price of apples.
- In the SOAP body, the xml web service is told to execute a function (we can also supply values as arguments)

The SOAP Response

```
<soap:Envelope
    xmlns:soap="http://www.w3.org/2001/12/soap-envelope">
    soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding" >
    <soap:Body>
        <m:GetPriceResponse
    xmlns:m="http://www.example.com/prices">
                        <m:Price>1.90</m:Price>
        </m:GetPriceResponse>
    </soap:Body>
</soap:Envelope>
```

The SOAP Response Explained

- The SOAP response is very simple:
  - It always follows a pattern.
  - Name of the function that was executed followed by the word response (***GetPriceResponse***).
  - Inside the ***GetPriceResponse*** element there is a sub element which contains the price (1.90).

31

Core Web Services Standards: HTTP ?

- Developed to facilitate the transfer of requests from a browser to a Web server.
- Web services takes advantage of this protocol to move SOAP messages and WSDL documents from one computer to another.
- NOTE: HTTP is purely request-response MEP.
- Other transport mechanisms such as FTP, SMTP, and JMS can be used to perform this same function. (Supported by Axis Server 1.4, released 2006)

32

Core Web Services Standards: WSDL

- Web services Description Language (WSDL) is a specification that describes a piece of software in terms of the method calls that it responds to.
  - Methods are described in an abstract way that is independent of programming language and operating system.
- It also contains the various details of how to actually make a connection to the service.
  - E.g. using http, which port is the service binded to, etc..
- Versions: 1.1 & 2.0.

33

WSDL: Description of a Web Service

- Consider the case in which you are told about a Web service that provides some information or processing that you would like to access.
- Your first questions would be:
  - What method calls does it accept?
  - What parameters will it accept?
  - What responses will it send back?
  - What protocols can it process?
  - What data format specifications can it accept?
  - What are the URLs for the service?

34

WSDL

- We can write software that can generate messages based on the logic in the program combined with the information in the WSDL.
- Normally, a potential Web service consumer would obtain the WSDL first.
- Using the WSDL
  - This would-be client could either have a programmer create software to this WSDL ,or
  - use software that is capable of generating a program to do the communications part of the client processing.

35

Four Abstract Elements

- The four abstract XML elements that can be defined in a WSDL are as follows:
  - <wsdl:types>
  - <wsdl:message>
  - <wsdl:operation>
  - <wsdl:portType>

36

Three Concrete Elements

- There are three concrete XML elements in a WSDL:
  - <wsdl:service>
  - <wsdl:port>
  - <wsdl:binding>

37

The types Element

- The *<wsdl:types>* element is used to indicate that a WSDL type is being declared.
- The types element describes all the **data types** used between the client and server.
  - WSDL is not tied exclusively to a specific typing system, but it uses the W3C XML Schema specification as its default choice.
  - If the service uses only XML Schema built-in simple types, such as strings and integers, the types element is not required.

38

A Customer Definition

- The following snippet shows us a customer definition that we can use to illustrate how this works:

```
<customer>
    <customerID>1001</customerID>
    <lastName>Maddox</lastName>
    <firstName>Greg</firstName>
    <address> 123 First Street</address>
    <city>Atlanta</city>
    <state>GA</state>
    <zip>30003</zip>
</customer>
```


XML Schema

```
<xsd:element name="customer">
    <xsd:complexType >
        <xsd:sequence>
            <xsd:element name="customerID" type="xsd:string"/>
            <xsd:element name="lastName" type="xsd:string"/>
            <xsd:element name="firstName" type="xsd:string"/>
            <xsd:element name="address" type="xsd:string"/>
            <xsd:element name="city" type="xsd:string"/>
            <xsd:element name="state" type="xsd:string"/>
            <xsd:element name="zip" type="xsd:string"/>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

Overview of the Schema

- The root element is called “customer”:
  - <xsd:element name=”customer”>
- A complexType is a user-defined type (*class in java*)
  - <xsd:complexType name=”customerType”>
- Each one of the fields that we want to include gets its own element definition.
  - The name will be the name of the data field, and the type will be the simple data type of that field.

Creating the <wsdl:types> section

- Now that we have a schema, we can create the <wsdl:types> section of the WSDL document that we are building.
- As you can see (next slide), the wsdl:types element is created directly by adding the <wsdl:types> and </wsdl:types> tags to the schema definition for this user-defined data type.
- The reason for doing this is simple.
  - The XML schema provides exactly the data needed to communicate the format of a data type.

Creating the <wsdl:types> section

```
<wsdl:types>
    <xsd:schema targetNamespace="http://www.xyz.com/customerType.xsd"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema">
        <xsd:element name="customer">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="customerID" type="xsd:string"/>
                    <xsd:element name="lastName" type="xsd:string"/>
                    <xsd:element name="firstName" type="xsd:string"/>
                    <xsd:element name="address" type="xsd:string"/>
                    <xsd:element name="city" type="xsd:string"/>
                    <xsd:element name="state" type="xsd:string"/>
                    <xsd:element name="zip" type="xsd:string"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
    </xsd:schema>
</wsdl:types>
```

43

The message Element

- *Messages* are one-way communications from one computer to another.
- For example:
  - In the typical request/response scenario, one message is sent and a second message is received.
- The message element is considered an abstract element because it describes the message logically instead of physically.

44

addCustomer Message

- If we create a message called *addCustomer*, we would add a message element to the WSDL called *addCustomer*:

```
<wsdl:message name="addCustomer">
    <wsdl:part name="customerInfo" element="tns:customer"/>
</wsdl:message>
```

- This message is going to add a customer to the Web service by sending it an instance of the customer element that we defined in the types element.

45

Response and Exception Messages

- This message will send an integer back that confirms that the addCustomer message successfully completed.

```
<wsdl:message name="confirmation">
    <wsdl:part name="response" element="xsd:integer"/>
</wsdl:message>
```

- This message will send an integer back that tells the client that the addCustomer message did not successfully complete.

```
<wsdl:message name="exceptionMessage">
    <wsdl:part name="badResult" element="xsd:integer"/>
</wsdl:message>
```

46

The operation Element

- The operation element is analogous to a method call in Java or a subroutine call in Visual Basic.
- Only three messages are allowed in an operation:
  - The Input Message:
    - Defines the data that the Web service expects to receive.
  - The Output Message:
    - Defines the data that the Web service expects to send in response.
  - The Fault Message:
    - Defines the error messages that can be returned by the Web service.

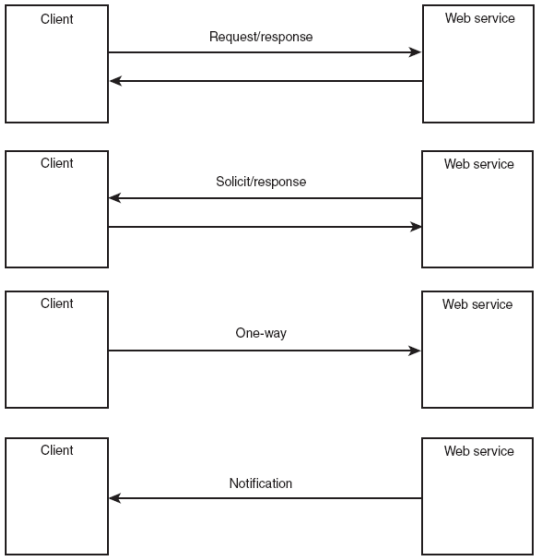
47

Types of Operations

- Several types of operations can be declared in a WSDL document:
  - **Request/Response:**
    - A client makes a request, and the Web service responds to it.
  - **Solicit/Response:**
    - A Web service sends a message to the client, and the client responds.
  - **One-way:**
    - A client sends a message to the Web service but expects no response.
  - **Notification:**
    - A Web service sends a message to the client but expects no response.

48

Types of Operations



49

Operation Syntax

- The syntax of an operation is simple. If the operation is a request/response type, the format will be as shown here:

```
<wsdl:operation name="createNewCustomer">
    <wsdl:input message="addCustomer">
    <wsdl:output message="confirmation">
    <wsdl:fault message="exceptionMessage">
</wsdl:operation>
```

50

The portType Element

- A port, in Web services jargon, is a single Web service.
  - The portType is the set of all operations that one Web service can accept.
  - It provides a one-stop point where a client can obtain information on all the operations that a Web service provides.

```
<wsdl:portType name="newCustomerPortType">
    <wsdl:operation name="createNewCustomer">
        <wsdl:input message="addCustomer"/>
        <wsdl:output message="confirmation"/>
        <wsdl:fault message="exceptionMessage"/>
    </wsdl:operation>
</wsdl:portType>
```


The binding Element

- The binding element serves two purposes:
  - It serves as the link between the abstract elements and the concrete elements in the WSDL.
  - It provides a container for information such as the protocol and the address of the Web service.

Parts of a Binding

```
…
<wsdl:binding name="newCustomerBinding"
                    type="newCustomerPortType">

<soap:binding style="rpc"
                     transport="http://schemas.xmlsoap.org/soap/http" />

<wsdl:operation name="createNewCustomer">
    ….
```


Binding With PortType

- The first line connects the binding with the portType that we created earlier.
  - The same portType can appear in more than one binding element (SMTP, HTTP,…).
- Our binding is designated to be a SOAP binding.
- In addition, it is going to use the HTTP to send the SOAP documents:

  <soap:binding style="document"
  transport="http://schemas.xmlsoap.org/soap/http" />

The port Element

- The only piece of information that is now missing is the actual IP address and port of the Web service that is represented by this WSDL.
- The *port element is* where this information is located.
- Syntax:

```
<wsdl:port binding="newCustomerBinding" name="newCustomerPort">
    <soap:address
    location="http://www.aicalecture.mu:1776/soap/servlet/rpcrouter">
</wsdl:port>
```

55

The service Element

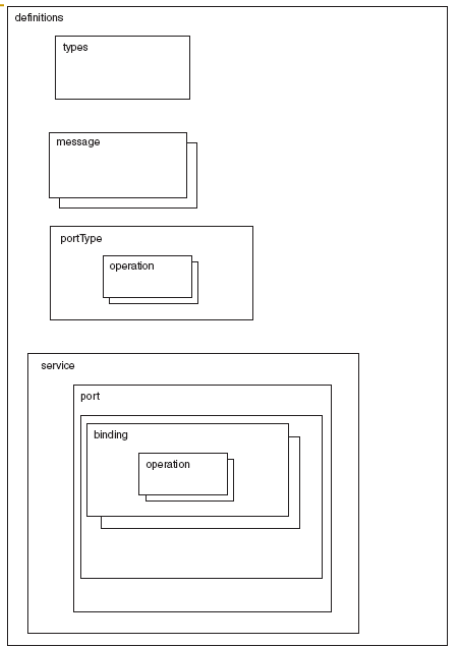
- Many Web services are gathered together and called a service.
  - **This element is a container for all ports that are represented by a WSDL document.**
  - The ports within a service can't be chained so that the output of one port is the input to another. As a result, the service tag is of limited value, but is required by the specification.

56

The definitions Element

- The root element in a WSDL document is:
  - *<wsdl:definitions>*
- *It contains elements* to specify the targetNameSpace, as well as a number of ordinary namespaces to help keep out naming conflicts.
- The rest of the WSDL document appears under this element. The end of the document is indicated by:
  - *</wsdl:definitions>*

57



58

Core Web Services Standards: UDDI

- Universal Discovery, Description, and Integration (UDDI) specification:
  - Describes how a potential customer of a Web service could learn about its capabilities and obtain the basic information needed to make the initial contact with the site.
  - Normally, this contact includes a download of the WSDL.
  - UDDI registries can be public, private, or semiprivate.

Tools & Vendors

- Web services can be developed using any programming language that supports sockets:
  - You could write a client that generated its own SOAP messages.
  - You could then open a socket and send the message to a Web service listening on that socket.
  - That Web service could do its own SOAP parsing, make its own method calls, write to its own logs, and prepare its own SOAP response message.
  - Finally, it could open a response socket and send the return message to the client, who could then display the results.

Existing Tools

- Tools range from special Java classes that know how to create SOAP messages to full-blown development environments:
  - **Apache Axis**
    - Apache Software Foundation coordinates the creation of open source projects. One of its projects is a SOAP engine that is normally used with its Tomcat server.
  - **Java**
    - Sun Microsystems has created a set of optional packages that can access UDDI registries, generate WSDLs, and so on.
  - **Visual Studio .NET**
    - Microsoft's new way to create Web services is to use this product in conjunction with any one of the Visual Studio languages such as Visual Basic, Visual C++, or C#.

Existing Tools

- Web Services .NET Clients
  - Clients created using .NET that can interact with any Web service.
- BEA WebLogic Workshop
  - BEA is a leading *J2EE vendor that has created a* user-friendly way to create Web services by using an elaborate IDE.
- IBM WebSphere Studio Application Developer (WSAD)
  - IBM has made the creation of Web services a part of its comprehensive package called WSAD.
- Other Important Products
  - Many lesser-known companies have quality entries with Other Toolkits: Iona XMLbus, The Mind Electric GLUE, PocketSOAP, and SOAP::Lite.

Who Manages the Web Services Specifications?

- **The World Wide Web Consortium**
  - The most important of these organizations is the World Wide Web Consortium (W3C).
  - The W3C states the following:
    - The World Wide Web Consortium (W3C) develops interoperable technologies (specifications, guidelines, software, and tools) to lead the Web to its full potential. W3C is a forum for information, commerce, communication, and collective understanding.
  - The W3C manages the SOAP, WSDL, XML, XML Schema, and HTTP specifications, among others.
  - In addition, the W3C manages the WS-Architecture document.

63

Who Manages the Web Services Specifications?

- **OASIS**
  - Another important organization in the Web services world is the Organization for the Advancement of Structured Information Standards (OASIS).
  - The OASIS Web site, www.oasis-open.org, states the following:
    - OASIS is a not-for-profit, global consortium that drives the development, convergence, and adoption of e-business standards.
  - OASIS manages UDDI, WS-Security, and SAML specifications, among others.

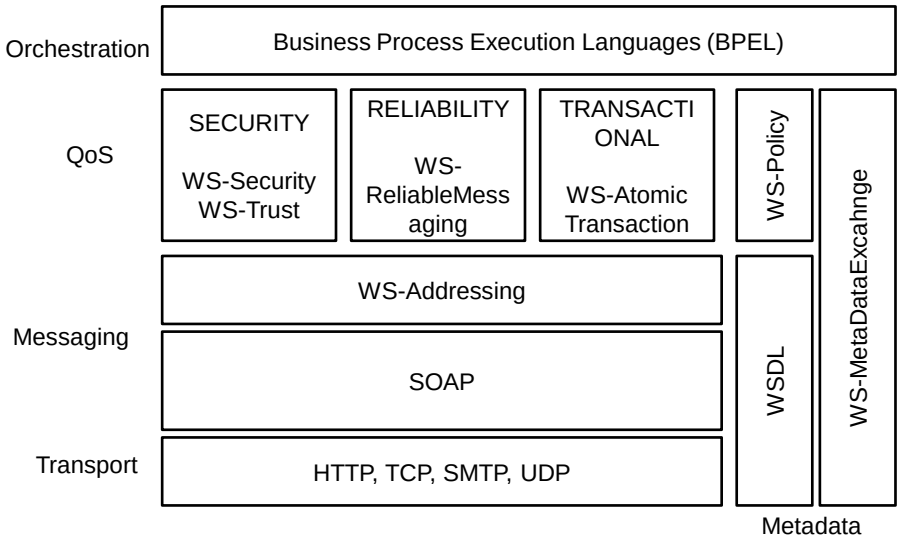
64

Who Manages the Web Services Specifications?

- **WS-I**
  - The Web Services Interoperability Organisation (WS-I) is now under OASIS
  - WS-I promotes Web services interoperability across platforms, operating systems, and programming languages.
  - The organisation works across the industry and standards organisations to respond to customer needs by providing guidance, best practices, and resources for developing Web services solutions.
  - The WS-I published profiles, testing tools, and sample applications that guarantee, as nearly as possible, that the different standards work together.

65

Web Services Platform – A complete Framework



66

REST vs. SOA (Web Services)

- REST
  - Representational State Transfer.
  - A Software architectural (style) for the WWW.
  - Requests and responses are built around the transfer of representations of resources.