Web Services

Unit 5

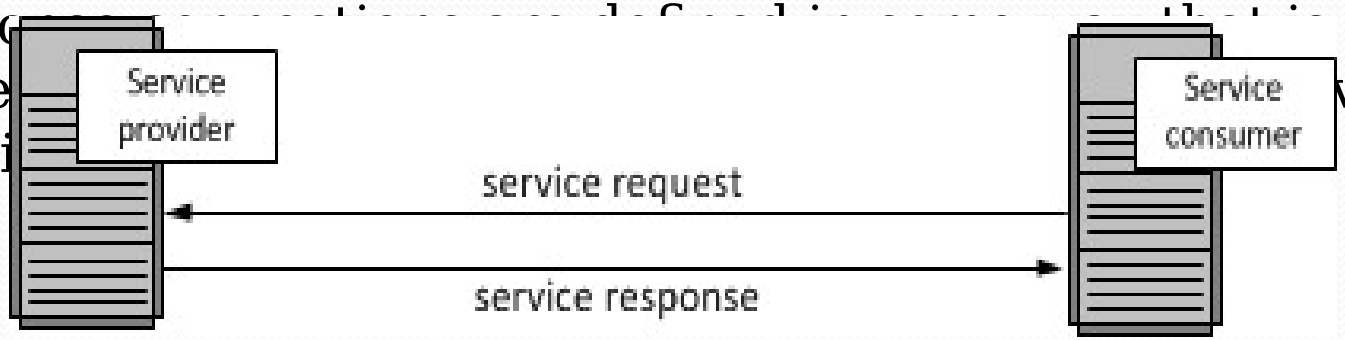
Web Technology-2 Note made by Lecturer
Nabraj Koirala, CCT College, Butwal

Web Service

- Web service is a standardized medium to propagate communication between the client and server applications on the World Wide Web.
- A web service is a software module which is designed to perform a certain set of tasks.
- The web services can be searched for over the network and can also be invoked accordingly.
- When invoked the web service would be able to provide functionality to the client which invokes that web service.

Service-Oriented Architecture (SOA)

- A service-oriented architecture is essentially a collection of services. These services communicate with each other. The communication can involve either simple data passing or it could involve two or more services coordinating some activity. Some means of connecting services to each other is needed.
- The following figure illustrates a basic service-oriented architecture. It shows a service consumer at the right sending a service request message to a service provider at the left. The service provider returns a response message to the service consumer. The request and subsequent respo
unde vice
provi



Web Service Components

There are three major web service components.

- SOAP
- WSDL
- UDDI

SOAP(Simple Object Access Protocol)

- SOAP is an XML-based protocol that lets you exchange info over a particular protocol (can be HTTP or SMTP, for example) between applications. It stands for Simple Object Access Protocol and uses XML for its messaging format to relay the information.
- SOAP is a communication protocol.
- SOAP is for communication between applications.
- SOAP is a format for sending messages.
- SOAP is designed to communicate via Internet.
- SOAP is platform independent.
- SOAP is language independent.
- SOAP is simple and extensible.
- SOAP allows you to get around firewalls.
- SOAP will be developed as a W3C standard.

Why SOAP?

- It is important for web applications to be able to communicate over the Internet.
- The best way to communicate between applications is over HTTP, because HTTP is supported by all Internet browsers and servers. SOAP was created to accomplish this.
- SOAP provides a way to communicate between applications running on different operating systems, with different technologies and programming languages.

SOAP elements (Building Blocks)

A SOAP message is an ordinary XML document containing the following elements:

- An Envelope element that identifies the XML document as a SOAP message
- A Header element that contains header information
- A Body element that contains call and response information
- A Fault element containing errors and status information

Skeleton SOAP Message

```
<?xml version="1.0"?>

<soap:Envelope
xmlns:soap="http://www.w3.org/2003/05/soap-envelope/"
soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">

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

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

</soap:Envelope>
```

The SOAP Envelope Element

- The required SOAP Envelope element is the root element of a SOAP message. This element defines the XML document as a SOAP message.

The SOAP Header Element

- The optional SOAP Header element contains application-specific information (like authentication, payment, etc) about the SOAP message.
- If the Header element is present, it must be the first child element of the Envelope element.

The SOAP Body Element

- The required SOAP Body element contains the actual SOAP message intended for the ultimate endpoint of the message.
- Immediate child elements of the SOAP Body element may be namespace-qualified.

The SOAP Fault Element

- The optional SOAP Fault element is used to indicate error messages.
- The SOAP Fault element holds errors and status information for a SOAP message.
- If a Fault element is present, it must appear as a child element of the Body element. A Fault element can only appear once in a SOAP message.

WSDL:

- WSDL is an acronym for Web Services Description Language.
- WSDL is a xml document containing information about web services such as method name, method parameter and how to access it.
- WSDL is a part of UDDI. It acts as a interface between web service applications.
- WSDL is pronounced as wiz-dull.
- WSDL is written in XML
- WSDL is a W3C recommendation.

Features of WSDL

- WSDL is an XML-based protocol for information exchange in decentralized and distributed environments.
- WSDL definitions describe how to access a web service and what operations it will perform.
- WSDL is a language for describing how to interface with XML-based services.
- WSDL is an integral part of Universal Description, Discovery, and Integration (UDDI), an XML-based worldwide business registry.
- WSDL is the language that UDDI uses.

WSDL Elements

A WSDL document contains the following elements −

- **Definition** − It is the root element of all WSDL documents. It defines the name of the web service, declares multiple namespaces used throughout the remainder of the document, and contains all the service elements described here.
- **Data types** − The data types to be used in the messages are in the form of XML schemas.
- **Message** − It is an abstract definition of the data, in the form of a message presented either as an entire document or as arguments to be mapped to a method invocation.
- **Operation** − It is the abstract definition of the operation for a message, such as naming a method, message queue, or business process, that will accept and process the message.
- **Port type** − It is an abstract set of operations mapped to one or more end-points, defining the collection of operations for a binding; the collection of operations, as it is abstract, can be mapped to multiple transports through various bindings.

- **Binding −** It is the concrete protocol and data formats for the operations and messages defined for a particular port type.
- **Port −** It is a combination of a binding and a network address, providing the target address of the service communication.
- **Service −** It is a collection of related end-points encompassing the service definitions in the file; the services map the binding to the port and include any extensibility definitions.

The WSDL Document Structure

The main structure of a WSDL document looks like this −

```
<definitions>
 <types>
definition of types……..
</types>
<message>
definition of a message….
</message>
 <portType>
 <operation>
definition of a operation…….
</operation>
</portType>
<binding>
definition of a binding….
 </binding>
<service>
definition of a service….
 </service>
</definitions>
```

WSDL Usage

- WSDL is often used in combination with SOAP and XML Schema to provide web services over the Internet. A client program connecting to a web service can read the WSDL to determine what functions are available on the server. Any special data types used are embedded in the WSDL file in the form of XML Schema. The client can then use SOAP to actually call one of the functions listed in the WSDL.

Universal Description Discovery and Integration

- UDDI is an acronym for Universal Description, Discovery and Integration.
- UDDI is a XML based framework for describing, discovering and integrating web services.
- UDDI is a directory of web service interfaces described by WSDL, containing information about web services.
- *UDDI*, is a platform-independent, Extensible Markup Language protocol that includes a XML-based registry by which businesses worldwide can list themselves on the Internet, and a mechanism to register and locate web service applications.
- UDDI was originally proposed as a core Web service standard. It is designed to be interrogated by SOAP messages and to provide access to Web Services Description Language (WSDL) documents describing the protocol bindings and message formats required to interact with the web services listed in its directory.

Structure of UDDI

- A UDDI business registration consists of three components:
- White Pages:
- Yellow Pages:
- Green Pages:

White Pages:-

- White pages give information about the business supplying the service. This includes the name of the business and a description of the business - potentially in multiple languages. Contact information for the business is also provided - for example the businesses address and phone number.

Yellow Pages:-

- Yellow pages provide a classification of the service or business, based on standard taxonomies. Because a single business may provide a number of services, there may be several Yellow Pages (each describing a service) associated with one White Page (giving general information about the business).

Green Pages:-

- Green pages are used to describe how to access a Web Service. It provides technical information about services exposed by the business. Because a Web Service may have multiple bindings , a service may have multiple Green Pages, as each binding will need to be accessed differently.

Representational state transfer

- **Representational State Transfer** (**REST**) is a software architectural style that defines a set of constraints to be used for creating Web services. Web services that conform to the REST architectural style, called *RESTful* Web services (RWS), provide interoperability between computer systems on the Internet. RESTful Web services allow the requesting systems to access and manipulate textual representations of Web resources by using a uniform and predefined set of stateless operations.

Architectural constraints of Restful system are given below:

- **Client-server architecture**
- **Statelessness**
- **Cacheability**
- **Layered system**
- **Code on demand**
- **Uniform interface**

Client-server architecture

- The principle behind the client-server constraints is the separation of concerns. Separating the user interface concerns from the data storage concerns improves the portability of the user interfaces across multiple platforms. It also improves scalability by simplifying the server components. Perhaps most significant to the Web, however, is that the separation allows the components to evolve independently, thus supporting the Internet-scale requirement of multiple organizational domains.

Statelessness

- The client-server communication is constrained by no client context being stored on the server between requests. Each request from any client contains all the information necessary to service the request, and the session state is held in the client. The session state can be transferred by the server to another service such as a database to maintain a persistent state for a period and allow authentication. The client begins sending requests when it is ready to make the transition to a new state. While one or more requests are outstanding, the client is considered to be *in transition*. The representation of each application state contains links that can be used the next time the client chooses to initiate a new state-transition.

Cacheability

- As on the World Wide Web, clients and intermediaries can cache responses. Responses must therefore, implicitly or explicitly, define themselves as cacheable or not to prevent clients from getting stale or inappropriate data in response to further requests. Well-managed caching partially or completely eliminates some client-server interactions, further improving scalability and performance.

Layered system

- A client cannot ordinarily tell whether it is connected directly to the end server, or to an intermediary along the way. Intermediary servers can improve system scalability by enabling load balancing and by providing shared caches. They can also enforce security policies.

Code on demand

- Servers can temporarily extend or customize the functionality of a client by transferring executable code: for example, compiled components such as Java applets, or client-side scripts such as JavaScript.

Uniform interface

- The uniform interface constraint is fundamental to the design of any RESTful system. It simplifies and decouples the architecture, which enables each part to evolve independently. The four constraints for this uniform interface are:

Advantages of RESTful Web Services

- **Fast**: RESTful Web Services are fast because there is no strict specification like SOAP. It consumes less bandwidth and resource.
- **Language and Platform independent**: RESTful web services can be written in any programming language and executed in any platform.
- **Can use SOAP**: RESTful web services can use SOAP web services as the implementation.
- **Permits different data format**: RESTful web service permits different data format such as Plain Text, HTML, XML and JSON.

SOAP vs REST Web Services

SOAP	REST
SOAP is a **protocol**.	REST is an **architectural style**.
SOAP stands for **Simple Object Access Protocol**.	REST stands for **REpresentational State Transfer**.
SOAP **can't use REST** because it is a protocol.	REST **can use SOAP** web services because it is a concept and can use any protocol like HTTP, SOAP.
SOAP **uses services interfaces to expose the business logic**.	REST **uses URI to expose business logic**.
JAX-WS is the java API for SOAP web services.	**JAX-RS** is the java API for RESTful web services.

SOAP	REST
SOAP **defines standards** to be strictly followed.	REST does not define too much standards like SOAP.
SOAP **requires more bandwidth** and resource than REST.	REST **requires less bandwidth** and resource than SOAP.
SOAP **defines its own security**.	RESTful web services **inherits security measures** from the underlying transport.
SOAP **permits XML** data format only.	REST **permits different** data format such as Plain text, HTML, XML, JSON etc.
SOAP is **less preferred** than REST.	REST **more preferred** than SOAP.