CHAPTER - 5

WEB SERVICES

INTRODUCTION TO WEB SERVICES:

Web services are XML-based information exchange systems that use the Internet for direct application-to-application interaction. These systems can include programs, objects, messages, or documents.

A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. This interoperability (e.g., between Java and Python, or Windows and Linux applications) is due to the use of open standards.

A web service enables communication among various applications by using open standards such as HTML, XML, WSDL, and SOAP. A web service takes the help of XML to tag the data, SOAP to transfer a message, WSDL to describe the availability of service. We can build a Java-based web service on Solaris that is accessible from our Visual Basic program that runs on Windows. We can also use C# to build new web services on Windows that can be invoked from our web application that is based on JavaServer Pages (JSP) and runs on Linux.

ADVANTAGES OF WEB SERVICES:

1. **Exposing The Existing Function On The Network:**

A web service is a unit of managed code that can be remotely invoked using HTTP, that is, it can be activated using HTTP requests. Web services allows you to expose the functionality of your existing code over the network. Once it is exposed on the network, other application can use the functionality of your program.

2. **Interoperability:**

Web services allow various applications to talk to each other and share data and services among themselves. Other applications can also use the web services. For example, a VB or .NET application can talk to Java web services and vice versa. Web services are used to make the application platform and technology independent.

3. **Standardized Protocol:**

Web services use standardized industry standard protocol for the communication. All the four layers (Service Transport, XML Messaging, Service Description, and Service Discovery layers) use well-defined protocols in the web services protocol stack. This standardization of protocol stack gives the business many advantages such as a wide range of choices, reduction in the cost due to competition, and increase in the quality.

4. **Low Cost Communication:**

Web services use SOAP over HTTP protocol, so you can use your existing low-cost internet for implementing web services. This solution is much less costly compared to proprietary solutions like EDI/B2B. Besides SOAP over HTTP, web services can also be implemented on other reliable transport mechanisms like FTP.

Web Service Architecture:

There are two ways to view the web service architecture:

- ➢ The first is to examine the individual roles of each web service actor.
- ➢ The second is to examine the emerging web service protocol stack.

Web Service Roles:

There are three major roles within the web service architecture:

☑ **Service Provider:**

This is the provider of the web service. The service provider implements the service and makes it available on the Internet.

☑ **Service Requestor:**

This is any consumer of the web service. The requestor utilizes an existing web service by opening a network connection and sending an XML request.

☑ **Service Registry:**

This is a logically centralized directory of services. The registry provides a central place where developers can publish new services or find existing ones. It therefore serves as a centralized clearing house for companies and their services.

Web Service Protocol Stack:

A second option for viewing the web service architecture is to examine the emerging web service protocol stack. The stack is still evolving, but currently has four main layers.

☑ **Service Transport:**

This layer is responsible for transporting messages between applications. Currently, this layer includes Hyper Text Transport Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP), and newer protocols such as Blocks Extensible Exchange Protocol (BEEP).

☑ **XML Messaging:**

This layer is responsible for encoding messages in a common XML format so that messages can be understood at either end. Currently, this layer includes XML-RPC and SOAP.

☑ **Service Description:**

This layer is responsible for describing the public interface to a specific web service. Currently, service description is handled via the Web Service Description Language (WSDL).

☑ **Service Discovery:**

This layer is responsible for centralizing services into a common registry and providing easy publish/find functionality. Currently, service discovery is handled via Universal Description, Discovery, and Integration (UDDI).

As web services evolve, additional layers may be added and additional technologies may be added to each layer.

Web Service Components:

Over the past few years, three primary technologies have emerged as worldwide standards that make up the core of today's web services technology. These technologies are discussed below.

1. XML-RPC (Extensible Markup Language - Remote Procedure Call):

This is the simplest XML-based protocol for exchanging information between computers.

☑ XML-RPC is a simple protocol that uses XML messages to perform RPCs.
☑ Requests are encoded in XML and sent via HTTP POST.
☑ XML responses are embedded in the body of the HTTP response.
☑ XML-RPC is platform-independent.
☑ XML-RPC allows diverse applications to communicate.
☑ A Java client can speak XML-RPC to a Perl server.
☑ XML-RPC is the easiest way to get started with web services.

2. SOAP (Simple Object Access Protocol):

SOAP is an XML-based protocol for exchanging information between computers.

☑ 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.

2.1. Elements of SOAP:

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

- **Envelope:** Defines the start and the end of the message. It is a mandatory element.
- **Header:** Contains any optional attributes of the message used in processing the message, either at an intermediary point or at the ultimate end-point. It is an optional element.
- **Body:** Contains the XML data comprising the message being sent. It is a mandatory element.
- **Fault:** An optional Fault element that provides information about errors that occur while processing the message.

2.2. SOAP Message Structure:

The following block depicts the general structure of a SOAP message:

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

  <SOAP-ENV:Header>
    ...
    ...
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    ...
    ...
    <SOAP-ENV:Fault>
      ...
      ...
    </SOAP-ENV:Fault>
    ...
  </SOAP-ENV:Body>
</SOAP_ENV:Envelope>
```

3. WSDL (Web Services Description Language):

WSDL is an XML-based language for describing web services and how to access them.

☑ WSDL stands for Web Services Description Language.
☑ WSDL was developed jointly by Microsoft and IBM.
☑ WSDL is an XML based protocol for information exchange in decentralized and distributed environments.
☑ WSDL is the standard format for describing a web service.
☑ WSDL definition describes 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 UDDI, an XML-based worldwide business registry.
☑ WSDL is the language that UDDI uses.
☑ WSDL is pronounced as 'wiz-dull' and spelled out as 'W-S-D-L'.

4. UDDI (Universal Description, Discovery and Integration):

UDDI is an XML-based standard for describing, publishing, and finding web services.

☑ UDDI stands for Universal Description, Discovery, and Integration.
☑ UDDI is a specification for a distributed registry of web services.
☑ UDDI is platform independent, open framework.
☑ UDDI can communicate via SOAP, CORBA, and Java RMI Protocol.
☑ UDDI uses WSDL to describe interfaces to web services.

- ☑ UDDI is seen with SOAP and WSDL as one of the three foundation standards of web services.
- ☑ UDDI is an open industry initiative enabling businesses to discover each other and define how they interact over the Internet.

RESTFul Web Services:

Web services based on REST Architecture are known as RESTful Web Services. These web services use HTTP methods to implement the concept of REST architecture. A RESTful web service usually defines a URI (Uniform Resource Identifier), which is a service that provides resource representation such as JSON and a set of HTTP Methods.

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.

HTTP Methods:

The following HTTP methods are most commonly used in a REST based architecture.

- ☑ **GET:** Provides a read only access to a resource.
- ☑ **PUT:** Used to create a new resource.
- ☑ **DELETE:** Used to remove a resource.
- ☑ **POST:** Used to update an existing resource or create a new resource.
- ☑ **OPTIONS:** Used to get the supported operations on a resource.

Difference between SOAP and REST:

No.	SOAP	REST
1	SOAP is a **protocol**.	REST is an **architectural style**.
2	SOAP stands for **Simple Object Access Protocol**.	REST stands for **REpresentational State Transfer**.
3	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.
4	SOAP **uses services interfaces to expose the business logic**.	REST **uses URI to expose business logic**.
5	**JAX-WS** is the java API for SOAP web services.	**JAX-RS** is the java API for RESTful web services.
6	SOAP **defines standards** to be strictly followed.	REST does not define too much standards like SOAP.

7	SOAP **requires more bandwidth** and resource than REST.	REST **requires less bandwidth** and resource than SOAP.
8	SOAP **defines its own security**.	RESTful web services **inherits security measures**from the underlying transport.
9	SOAP **permits XML** data format only.	REST **permits different** data format such as Plain text, HTML, XML, JSON etc.
10	SOAP is **less preferred** than REST.	REST **more preferred** than SOAP.