CoreJava | JSP | Servlets | JDBC | Struts | Spring | Hibernate

Projects | FAQs | Sample Programs | eBooks | Certification Stuff

Question Banks | Communities | Tutorials | Softwares | Sample Resumes

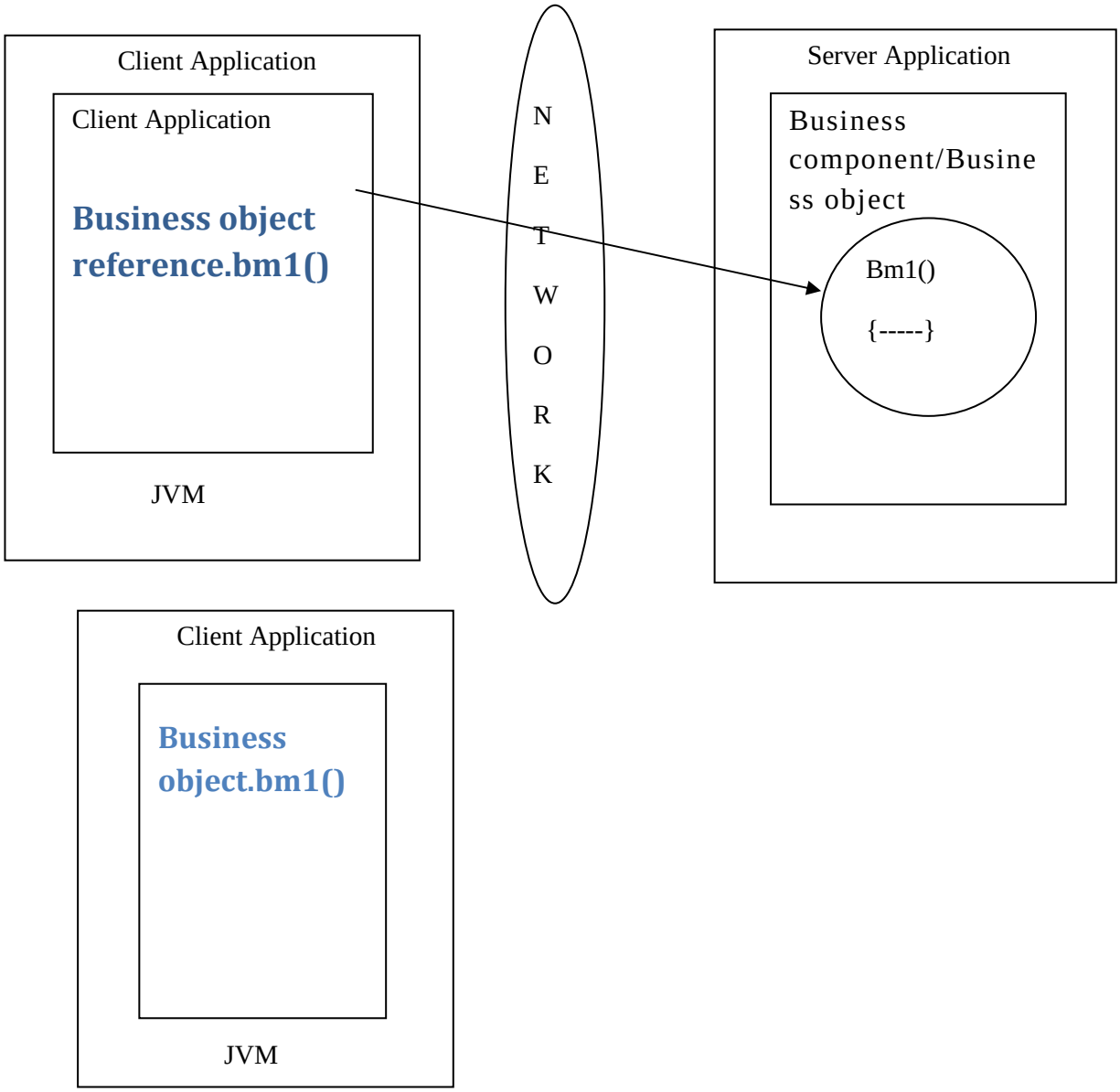
Interview Tips | Forums | Discussions | Online Test Engines | Jobs

www.JavaEra.com

A Perfect Place For All Java Resources

Web Services: web-services is a distributed technology which allows to develop distributed applications.

Understanding Traditional client-server approach: In traditional client-server approach the client application interact with business object of Server application directly.



Note: If there is any change in the location of the Server application then we must modify the code of client application to interact with Server application that is changed to new Location this is called **Location Dependency.**

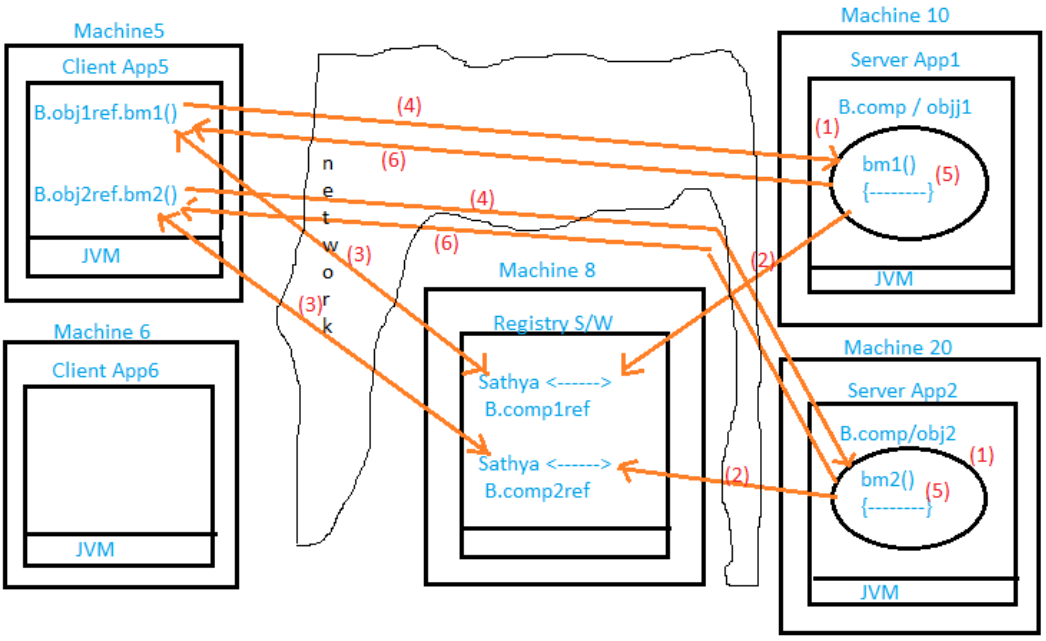
EX: Socket programming based client-server applications are two-tier applications or traditional client-server applications.

Drawbacks of Traditional client-server Approach:

1. These applications are location dependent applications.
2. Keeping multiple components in one Server then application increases burden on the Server Application.

3. But keeping multiple components in multiple Server applications makes each client taking with multiple Server application having location dependency.

Note: To solve above problem we have to work with Distributed technology.



Understanding Distributed Technology:

1. The client-server application with Location Transparency utilizing Registry software as mediator that client-server application is called as Distributed application.
2. In Distributed Applications multiple Server applications will be there maintaining multiple business components to distribute burden/work.
3. In Distributed applications development more client applications will be there, single registry will be there and one or more Server applications will be there. Registry/JNDI register provides global visibility to object/object reference.
4. In one registry we can maintain one or more object/object references nick names or Alias names.
5. With respect to diagram multiple Server applications will be developed in multiple machine having business components.
6. These multiple business objects business objects reference will be bound with single Registry having nick names.
7. Client application gathers one or more business object reference from Registry through Lookup operation.

8. Client application calls business method on business object reference.
9. In the business method of business objects executes in Server application.
10. Results gathered for business methods come to Client application from Server application.

***Note:**

1. In Distributed application Registry software must be there to find location but client-server applications may change their locations.

2. Distributed application's client talk with Server application only after taking with business object reference from Registry. So any change in location of Server application, we just need to inform to Registry software there is no need of informing to Client Application and modifying their code. This is called Location Transparency.

Advantages of Distributed Applications:

1. Distributed applications are Location Transparency applications.
2. Distributed Application allows distribute/share the burden in multiple Server applications.
3. Suitable for Large scale/complex applications which contains huge amount of Clients.

List of Distributed Technologies to develop Distributed Applications:

1. RCP → Open Community
2. RMI → SUN Micro Systems(Oracle)
3. DCOM → Microsoft
4. Remoting → Microsoft 3
5. EJB → from SUN Microsystems 2
6. Http Invoker → from Interface 21 4
7. CORBA → from OML
8. Web services → from Open Community 1 (rating)

The plan of manufacturing computer is called as Architecture of computers.

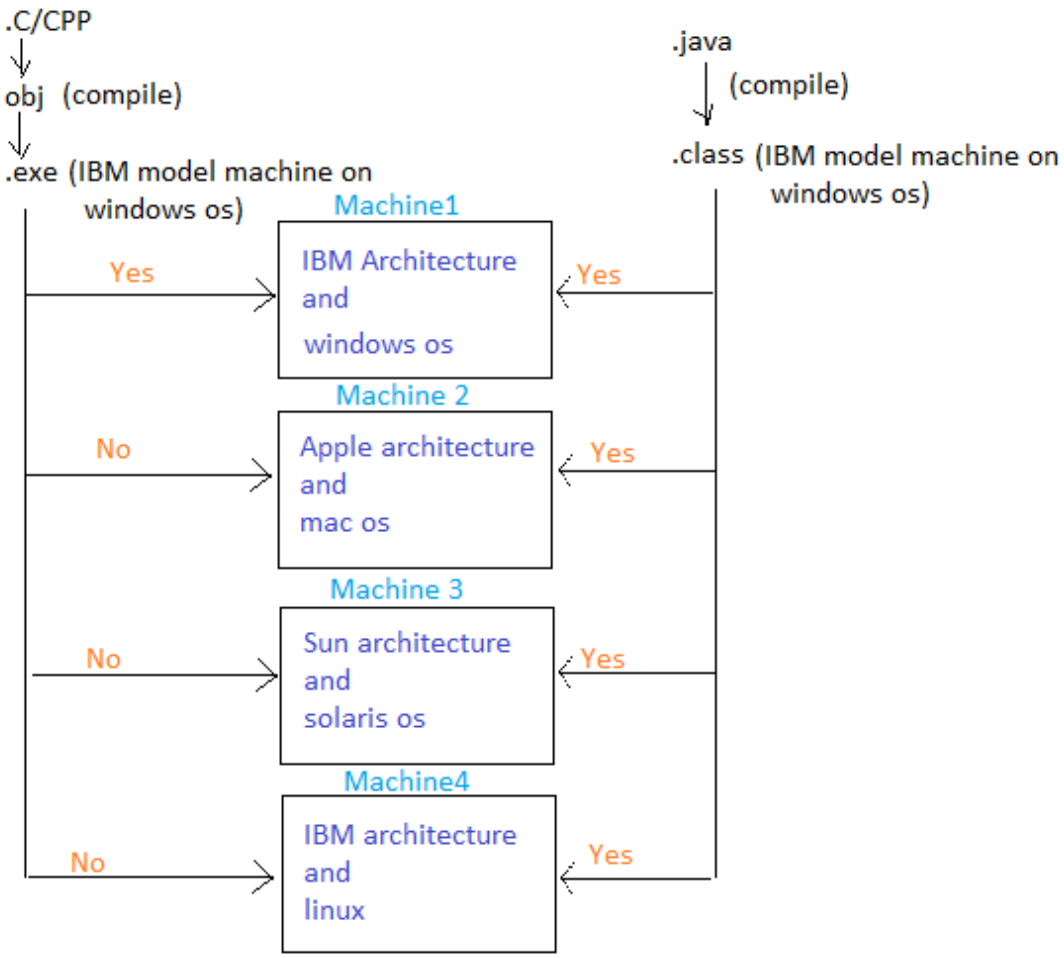
There are 3 popular architectures to manufacture the computers

1. IBM Architecture
2. Apple Architecture
3. Sun Architecture

→C, C++ is platform dependent and Architecture dependent because the .exe file contains platform (operating system) and Architecture specific instructions.

→Java is platform independent and Architecture independent because .class file does not contain operating system and computer Architecture related instructions.

→All our regular computers are IBM model computers.



RPC (Remote procedure Calls): -

1. Applications must be developed in C, C++ technologies.
2. Platform dependent (operating system dependent).
3. Architecture dependent.
4. Language dependent. (Both client and server applications must be developed same language (C or C++))
5. Out dated technology.

RMI (Remote Method Invocation): -

1. From Sun ms (Java based)
2. Platform independent.
3. Architecture independent.
4. Language dependent (Both client and server applications must be developed in java)
5. Not suitable for large scale applications.
6. Can't use internet network as communication channel.

7. Does not provide built-in middleware services.

DCOM (Distributed Component Object Model): -

1. From Microsoft (Microsoft Technology based)
2. Platform dependent
3. Architecture dependent
4. Language independent
5. Suitable for working with Microsoft dependent technologies.

CORBA (Common Language Request Broker Architecture)

1. From OMC (Object management group) 800+ companies except Microsoft.
2. CORBA is specification having rules and guidelines to develop distributed technologies.
3. CORBA based technologies are IDL(Interface Definition Language) from SUN Microsystems.
4. According to CORBA specification
    i. Applications are Platform, Architecture, Language independent
    ii. The Allowed languages are C, C++, Java, Objective, C, ---------
5. CORBA is Strong in specification wise but filed in implementation

EJB:

1. From SUN Microsystems (Java based)
2. Enhancement of RMI having maximum features that CORBA
3. Platform and architecture independent
4. Language dependent (both client and server applications are should be developed in Java).
5. Allows to use internet network as communication channels
6. Allows to use the Application Server supplied middleware services
7. EJB is a specification having rules and guidelines to develop EJB contains software.
8. EJB is industry standard to develop business components in Java based applications.

Web services:

1. Form open community
2. It is web –based distributed technology(the Server application of Distributed application must be a web application)
3. Language independent
4. Platform independent and Architecture independent
5. Allows to convert other technologies (any) based application to web service application.
6. Uses http and SOAP application protocol(Simple Object Application protocol)
7. In any technology base, distributed applications development the following resources are occurred.

The plan of manufacturing computer is called as architecture of the Computer. There are three popular architectures of computer. They are

1. IBM Architecture
2. Apple Architecture
3. SUN Architecture

C, C++ are platform independent and architectural dependent because the .exe file contains platform (OS) and architecture instructions.

i. **Service Provider Application:** It is Server Application (one or more) having the business logic component and business objects.
   In web service environment it can be Java/.NET application.
ii. **Service Client:** The Client Application that calls business method of business components belonging to Server Application.
iii. In Web Service environment Application can be developed in Java or .NET.

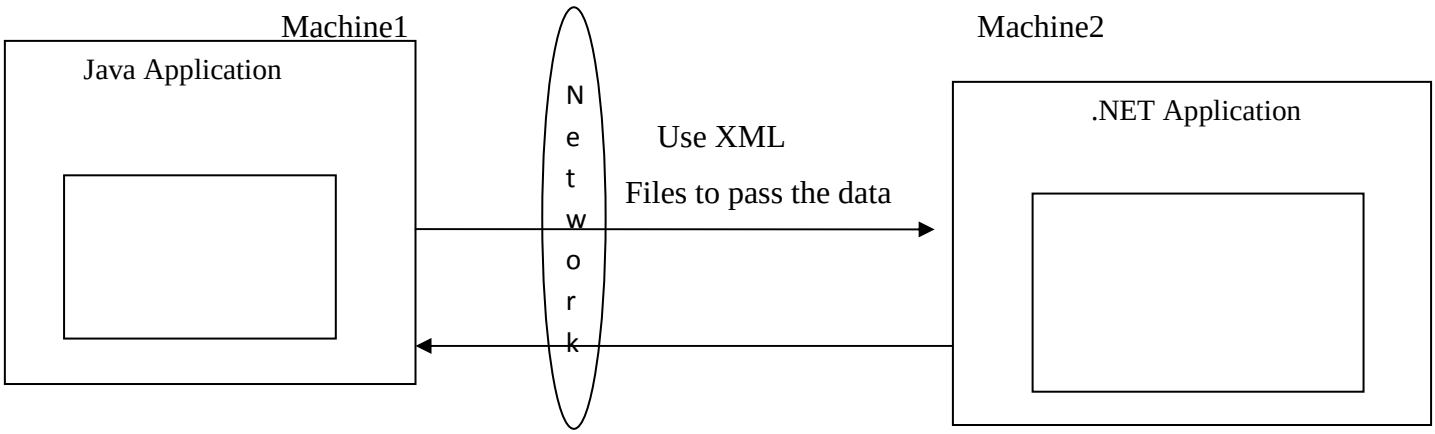
Service or Interface: The common understanding difference between Service provider and Service Client applications containing the declaration of methods.

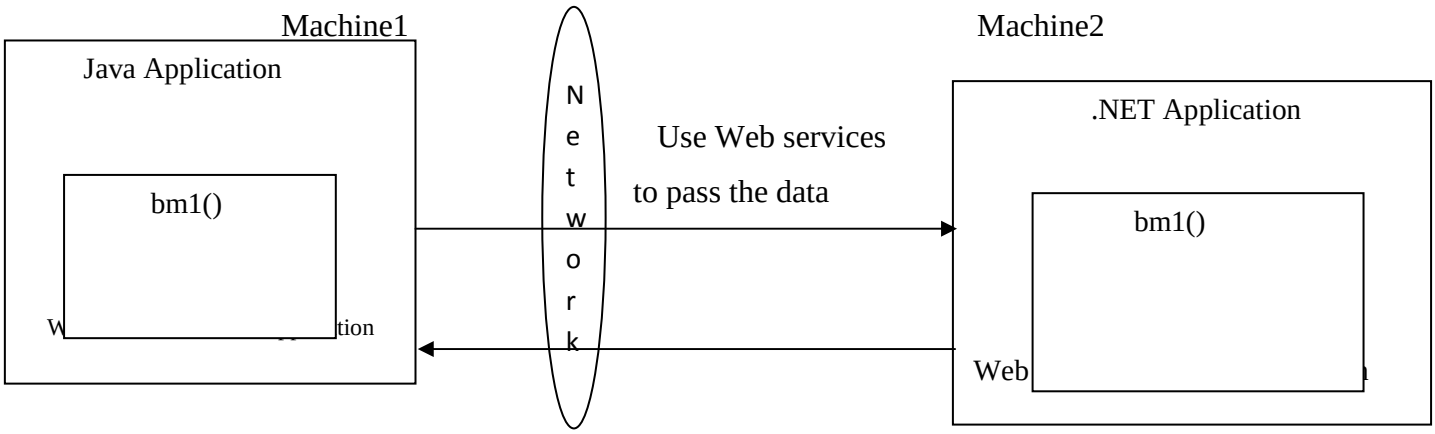
Note: In web service environment this is WSDL (Web service Decrypting Language)

Registry: The mediator between Service client and Service Provider having business component details

Note: In web service environment registry is UDDI.

If we want to pass data between to incompatible applications then pass that data in the form of XML Files.



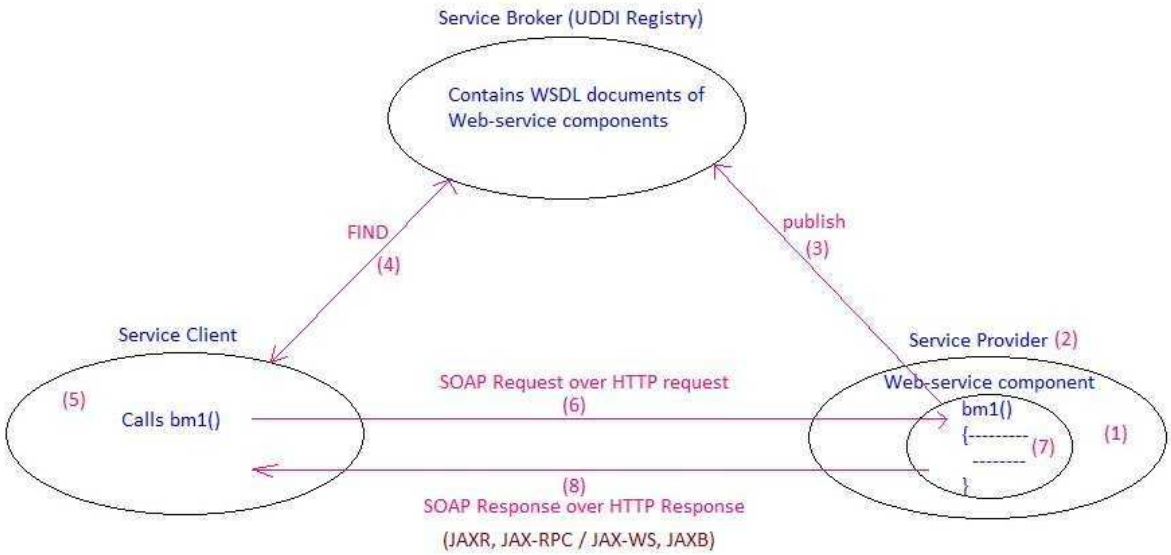


If we want to see the interaction between to incompatible applications then they must be developed as "web-services enabled applications".

In Web service environment

4. If we want to see the interaction between two incompatible applications then they must be developed as web service enabled application.
5. If we want to pass the data to incompatible applications then pass that data with the form of XML files.

Overview diagram of web services:



SOAP response over HTTP Response

JAXR, JAX-RPC/JAX-WS, JAXB

Web Services Definition: Web services is an XML, SOAP, HTTP based distributed technology that allows us to develop interoperable distributed applications by using technology or programmer choice.

i. All software companies are interoperable works everywhere without working about compatibility.
ii. To develop interoperable software components use web services that is we can use .NET components in Java and Vice-Versa

With respect to diagram.

1. Service provider develops web service component as web application in his choice technology(.NET, Java) having business methods
2. Service provider generates web document for web service component having the details about components and business methods(XML)
3. Service provider publishes the web document in UDDI registry for global visibility.
4. Service client gathers the WSDL document in UDDI registry for global visibility.
5. Service Client understands WSDL document and develops client Application in his choice technology. This client application calls the business methods of web service component.
6.

7.
8. Web service client Application calls the business methods of the web services component and gathers the result by using SOAP over HTTP protocol.

To develop all the resources of Web services enabled Applications in JAVA:

a. Use JAXP API makes service provider and Client interacting with UDDI Registry.
b. Use JAXRPC/JAX-WS to develop web service component and clients.
c. Use JAXB to get the interaction between Client and Component.

Note: As of now Java/JEE, .NET are the two major technologies to work with web services.

Web Services Advantages:

1. **Enables Interoperability of legacy and heterogeneous applications:** Here Interoperability means the web service model is that to permits different distributed services to run on a variety of software platforms and architectures, and allows them to be written in different programming language.
2. **Enables Just-in-time integration:** It means dynamic discovery and invocation(publish, find, bind).
3. Business Service through web
4. **Integration with existing Systems:** For example assume that there are some computer systems which are having lot of data in old fashion that means old programming format etc., So if we replace these systems it will become expensive. So by using Web Services we can interact them easily.
5. **Freedom of Choice:** It means in an organization people can select different configuration systems and programming languages.

While working with XML Schema we need to deal with Schema Namespaces.

Schema Namespace is a library that contains set of xml tags. Every Namespace is identified through its uri/url.

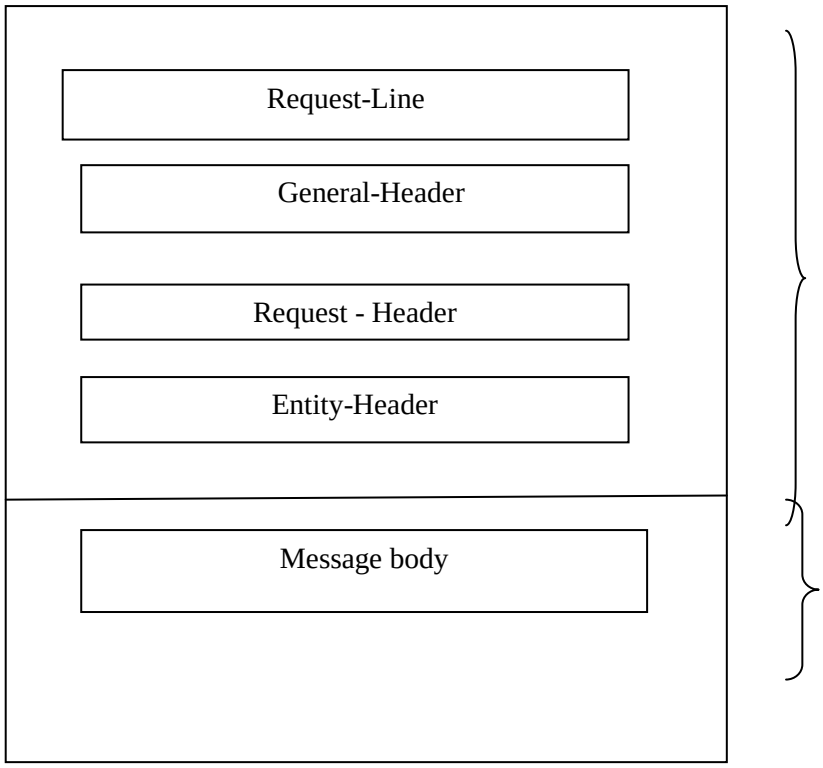
Examples:

www.example.org

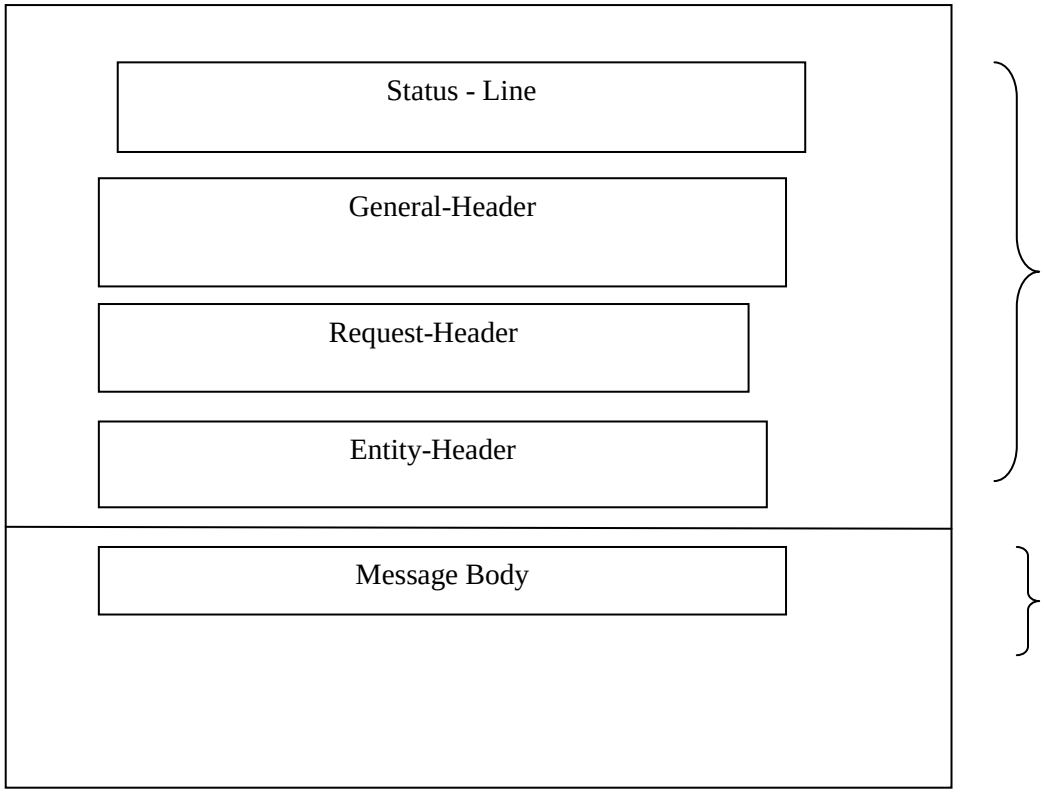
www.w3schools.com/2001/abc

1. In SOAP messages and WSDL documents lot of XML tags will be defined by specifying their namespace uris.
2. Web service client interacts with web service component by using the “SOAP over http protocols”.
3. For basics of http protocol please refer page numbers 9 to 11.

Parts of HTTP Request:



Parts of HTTP Response:



For related on SOAP (Simple Object Access protocol) refer 23-31

In plain HTTP Request, Response the body will be normal content.

In SOAP over HTTP Request, SOAP over HttpResponse the body will be the XML tags based SOAP message as whon in page no:27 of the handout

WSDL (Web Service Description Language): For info on WSDL please refer 33 to 38 of material.

UDDI (Universal Dictionary Description Language): For info UDDI Registry refer 39 to 41

In resume of Web services keep in

Service Oriented Architecture (SOA)

To develop Web services components and Client applications we can use the following APIs and Frameworks of Java Environment

JAX-RPC → API from SUN Micro systems

JAX – WS → API from SUN Micro systems

AXIS → Framework Apache (based JAX-WS)

Metro → Framework from open community (based on JAX-WS)

As of Now there are two methodologies to develop Web services (both Client and Component)

1. SOAP based web services
2. REST FULL Web services (REST→ Representational state)

- In SOAP based Web services SOAP over HTTP protocol will be used.
- In RESTFULL Web services protocol HTTP will be used directly.
- REST is not a protocol it is methodology to develop web services without using SOAP protocol.

The two ways of developing SOAP based Web services components:

1. **Bottom-up Approach (Business first Applications):** In this Approach first the Web service business components will be developed and they will be used to generate WSDL docs.
2. **Top-Down Approach (Contract first):** In this Approach WSDL document will be developed and that will be used as base document to develop Web service components.

- In large scale environment it is recommended to use Top-down Approach where multiple web service components can be developed based on single WSDL documents.
- In small and medium scale projects we can use Bottom-Up Approach where the components will be developed quickly.

Q) What are the differences between JAX-RPC and JAX-WS?

JAX-RPC	**JAX-WEB**
1. Uses SOAP 1.1 version 2. Supports WSDL 1.1 3. Given based on JDK1.4 4. Given based on J2EE1.4 5. Does not support Annotations 6. Supports only Synchronous	1. Uses SOAP 1.2 version and also 1.1 2. Supports WSDL 1.1,1.1,2.X 3. Given based on JDK1.5 4. Given based on Java EE5 5. Supports Annotations 6. Supports Synchronous, Asynchronous

communication between Client and Component. 7. Does not support messages based communication. 8. Uses SOAP to transfer XML data. 9. Not a industry recommended standard.	communication between Client and Component 7. Supports messages based communication. 8. Uses SOAP to transfer XML/HTTP binding to transfer XML data. 9. Not a industry recommended standard.

Note: JAX-WS is built-in API from JDK1.6 onwards

JAX-WS API is

Javax.jws, javax.jws.soap

Annotations of JAX-WS API:

@webService(Makes the Java class as Web service component)

@webMethod(Makes Java method as Web service operations nothing but business methods)

@webParam(To continue the parameters of Business methods)

@webResult(to specify return type of business method)

@oneWay (useful when business method takes input but does not generate output)

@handlerChain(useful in Asynchronous communication) and etc.,

Setup that is required for JAX-WS based Web service components and Client applications:

JDK1.6+

Any Server that is compatable with JDK1.6+ (GlassFish 2.X/3.X and Weblogic 10.X)

Procedure to develop JAX-WS 2.0 based web service component by following Bottom-up approach (Business First Approach)

1. Install JDK 1.6 +, Glassfish 2.X softwares in your computers.

   (Domain creation in Glassfish 2.x

   Please refer **(domain creation in DomainCreationInGlassFish)**

2. Develop the following web application as web service component.

Deployment Directory Structure

C:\apps

|----->webservices

|----->application1

|----->component

|----->WEB-INF

|------->classes

| |------>Calculator.java

| |------>p1

| |--->Calculator.class

| |--->web.xml

|--->lib

|--->ojdbc14.jar (oracle10g)

|--->ojdbc6.jar (oracle11g)

----------Calculator.java----------

```
//Calculator.java
package p1;
import javax.jws.*;
import java.sql.*;

@WebService
public class Calculator
{
	@WebMethod
	public int add(int a, int b)
	{
		int sum=a+b;
		System.out.println("Calculator:add(-,-).....");
		return sum;
	}//add(-,-)
	@WebMethod
	public int getEmpCount()
	{
	  int cnt=0;
	  try
```

```
            {
              Class.forName("oracle.jdbc.driver.OracleDriver");
              Connection con = DriverManager.getConnection
              ("jdbc:oracle:thin:@localhost:1521:orcl11g","scott","tiger"); Statement
              st=con.createStatement();
              ResultSet rs=st.executeQuery("select count(*) from emp");
              if(rs.next())
                    cnt=rs.getInt(1);
               rs.close();
               st.close();
               con.close();
             }//try
             catch (Exception e)
             {
                    e.printStackTrace();
              }//catch
              return cnt;
          }//getEmpCount()
}//class
```

Compilation:

E:\apps\webservices\application1\component\WEB-INF\classes>javac –d . Calculator.java

3. Prepare war file representing the above web service component.

E:\apps\webservices\application1\component>**jar cfv calc.war .**

4. Start mydomain1 domain server of Glassfish 2.X

C:\sun\appserver\bin>asadmin start-domain mydomain1 (enter)

5. Deploy the calc.war file in mydomain1 domain server.

Copy calc.war to C:\Sun\AppServer\domains\mydomian1\autodeploy folder

NOTE: - Glassfish comes with a built-in UDDI registry (private registry).

6. View the WSDL document generated for the above web service component.

Open browser window

|

V

http://localhost:8080/calc/CalculatorService?WSDL

| |

V V

(War file name) (Logical name of web

Service component (class name+Service)

Observe the following window:

Note: To view url of this browser window just observe right side of your msword window and increase size of the window100% to some what high

Note: Every IDE softwares supporting JAX-WS based web services development:

MyEclipse 8.X

NetBeans 6.7.7.1

1. SOA (Service oriented Architecture) is a specification containing rules and guidelines to develop XML, HTTP based interoperable distributed technologies.
2. Web service is the technology whose implementation can be done either in Java or .NET based specification as distributed technology.

Procedure to develop java based JAX-WS web service client application for the above web service component

1. Create a separate folder to develop and execute client application

E:\apps

|----->webservices

|----->application

|----->clientapplication

2. Use **wsimport** tool to generate the client side proxy classes that are required to interact with web Service component from web service client application

E:\apps\webservices\application1\clientapplication>

wsimport –p nithya –keep http://localhost:8080/calc/CalculatorService?wsdl

NOTE: -

i. wsimport is built-in tool of jdk1.6

ii. Make sure that Glassfish domain server is in running mode while working with this tool.

iii. The above step generates package ‘nithya’ having lot of proxy classes.

3. Develop JAX-WS based web service client application by using above generated proxy classes.

```
import nithya.*;
public class TestClient
{
```

```
public static void main(String args[])throws Exception
{
//locate webservice comp through proxy classes
CalculatorService cs=new CalculatorService();
Calculator stub=cs.getCalculatorPort();
//call B.methods
int sum=stub.add(10,20);
System.out.println("the sum is:"+sum);
int cnt=stub.getEmpCount();
System.out.println("Emp count is:"+cnt);
}//main
}//class
```

4. Compile and execute the client application

Compilation:

E:\apps1\webservices1\application1\clientapplication>java Testclient.java

Execution:

C:\apps1\webservices1\application1\clientapplication>java Testclient

Output:

The sum is: 30

Emp count is: 14

For complete program please refer Java Programs/apps folder

Procedure to develop Web Service component(JAX-WS) by using NetBeans IDE:

For complete process please refer D:\Frameworks\web services\NetBeans 6.7.1 projects\First Application.

Procedure JAX-WS based client application for the previous .NET web service component:

1. Make sure that .NET Web Service component is in running mode.
2. Create web project in NetBeans IDE.
3. Add Web service client reference to the project → Right click on project → new → Web service client → WSDLurl → (do not enter any thing here)

Client style: JAXWS → Finish → Delete the existing code of index.jsp and drag and drop the hello world to index.jsp from Web Services references → service1 → service1 SOAP12
Run the project

Note:

1. When Web Service component is deployed in Server then the Servers used wsgen tool internally to convert Annotations based class into Web service component.
2. An IDE generate proxy classes required for client application development while working with JAXWS Environment.

Developing EJB Component by using NetBeans IDE.Working with MyEclipse IDE for JAXWS:

For complete process please refer webservices\NetBeans6.7.1 Projects\EJB-WebServices.docx

Note: While developing this application some problems will arise in IDE. Some times it executes fine. Later it won't execute these types of problems araise. So that I didn't keep this application related project here.

Procedure to develop Web Services Component and Client in MyEclipse IDE

In every JAXWS based Web Service component one Front Controller Servlet(WSService) will be generated automatically to trap the all http requests coming from clients and top pass then to the class acting as Web Service component.

For complete process please refer MyEclipse(Tomcat)WebServices document.

For complete program please refer web services\MyEclipse 8.X projects\First App folder. In this folder two folders are there those are the project folders of MyEclipse IDE.

Bottom-Up Approach	Top-Down Approach
1. It is called as Java WSDL/business first approach. 2. The developed Web Service document is not stable and reliable document it contains the effect of underlying technology. 3. Good for converting existing components into Web Services.	1. It is called WSDL-Java/Contract first Approach. 2. WSDL document is stable, reliable and does not contain underlying technology effect. 3. Good to develop the Web Service components from scratch level.

4. Strong knowledge of XML Schema, WSDL is not required. 5. Good for exposing overloaded business components. 6. Gives less control on WSDL document. 7. After developing Web Service component we need to develop Client Application. 8. We can take more support of tools	4. Strong knowledge of XML Schema, WSDL is required. 5. Complex for exposing overloaded business components. 6. Gives more control on WSDL document. 7. Based on WSDL document we can develop Client and Web Service components parallel. 8. Tools support is limited.

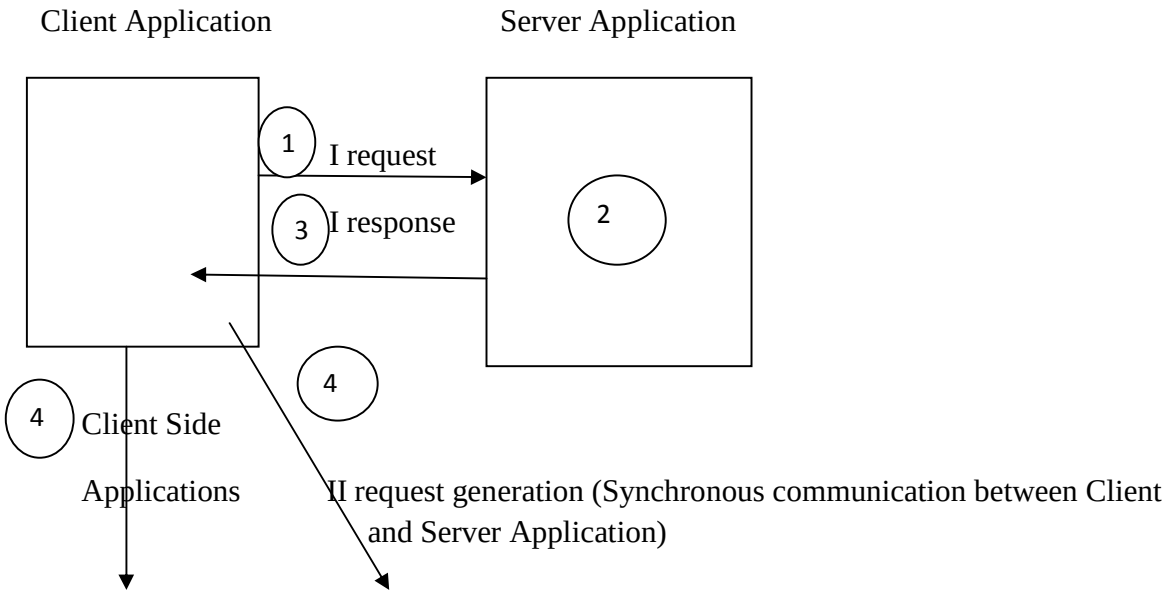
Note: All components that we have developed so for comes under Bottom-Up Approach based Web Service component.

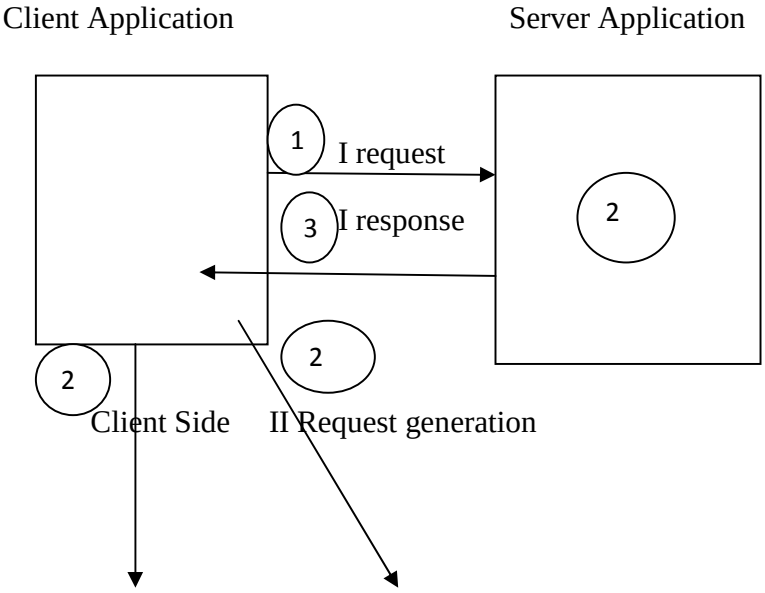
Procedure to develop WSDL document base (Top-Down Approach) Web Service component by using MyEclipse IDE:

For complete process and program please refer D:\Frameworks\web services\MyEclipse 8.X projects\second APP (TopDown Approach) folder.

In NetBeans IDE: For Complete project D:\Frameworks\web services\NetBeans 6.7.1 projects\Second Application.

Asynchronous Communication: Using JAX-RPC we can develop only Synchronous communication Web Component where as by using JAXWS we can develop both Synchronous and Asynchronous communication based Web Service component.





1. In Synchronous communicaton Client Application must wail for given request response from Server Application to generate next request or to perform same Client side operations.
2. In Asynchronous communication Client is free to generate next request to Server Application without waiting for given request related response.

Using JAXWS we can develop Asynchronous based Web Services in two Approaches:

1. Uisng Polling Mechanism
2. Using Handler Mechanism

1. **Polling Mechanism:** Internally someone continuously verifies whether response is arrived or not form component. If response once arrived it will be given to Client Application till that time Client Application free to generate next request or to perform Client side operations.

2. **Handler Mechanism:** Here handler method will be executed when response is overrided from Server till that time Client is free to perform Client side operations or generate next request.

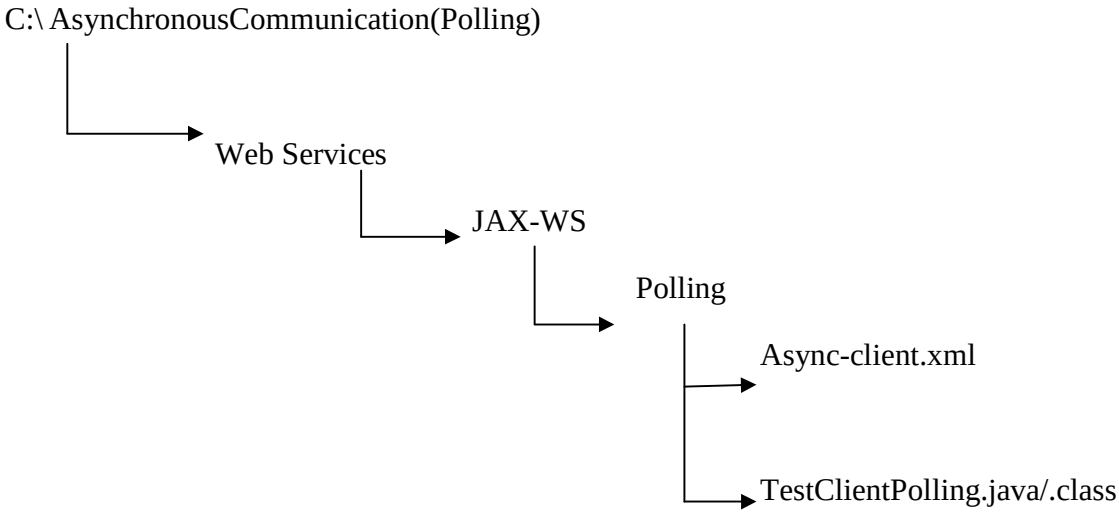
Note: Business method call done in Client Application generates request to Web Service component. The result given by business method execution comes to Client Application as response.

Example Web Service Component and Client Application development to demonstrate JAXWS based Asynchronous Communication:

Keep calc.war file of Day-1 first application JAXWS ready and deployed in Glassfish Server.

Polling based Web Service Client Application development:

To generate Proxy and Helper classes supporting Asynchronous Communication:



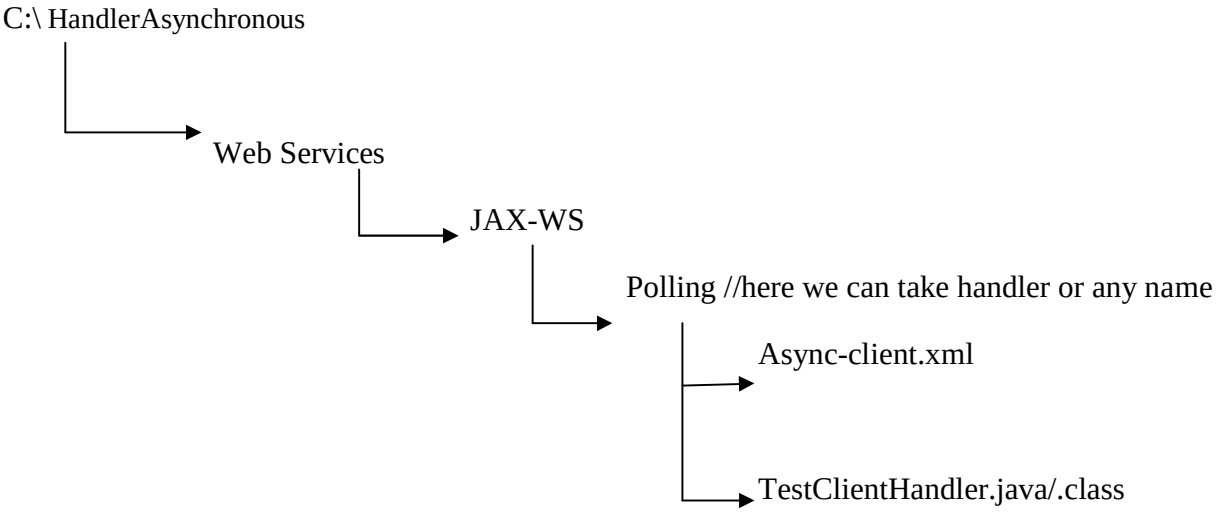
C:\AsynchronousCommunication(Polling)\webservices\jax-ws\polling>wsimport -p mypack -keep http://localhost:2020/calc/CalculatorService?WSDL -b async-client.xml

Now mypack folder(package) will generate automatically this will give Proxy classes by default.

Now develop Client application.

For complete program please refer web services\Programs\AsynchronousCommunication(Polling) folder

Procedure to develop Handler based Asynchronous Web Service Client application for the above Web Service component.



For complete program please refer web services\Programs\HandlerAsynchronous folder

Consuming Web Service components which are available in publicly:

www.service-repository.com, www.uddi.com and etc., are public repositories containing free and publicly available Web Service components in our Web Service client applications.

Procedure to consume Calculator Web Service component of service-repository.com registry:

Note: Ensure that Internet connection is available or not properly otherwise we will get blank screen with lines in browser window.

For complete procedure please refer D:\Frameworks\web services\NetBeans 6.7.1 projects\Consuming Web Service components\Consuming public web service component (calculator) document

For complete program please refer web services\NetBeans 6.7.1 projects\Consuming Web Service components\ConsumingWebServiceComponent(Calculator).

Consuming weather Web Service components which are available in publicly:

For complete process please refer web services\NetBeans 6.7.1 projects\Weather Report\weather.docx.

For complete program please refer web services\NetBeans 6.7.1 projects\Weather Report\WeatherReport.

Authentication

Checking the identity of a user is called authentication checking the access presentation of user on certain resources of the application is called **authentication**

Application Security = Authentication + Authorization

Programmer is not responsible for network security but this is responsible for application level security.

By working with Web Service the client applications can send additional data along with request as user defined request header values

In Authentication based example Web Service client applications sends user name and password details as user defined Http request header values.

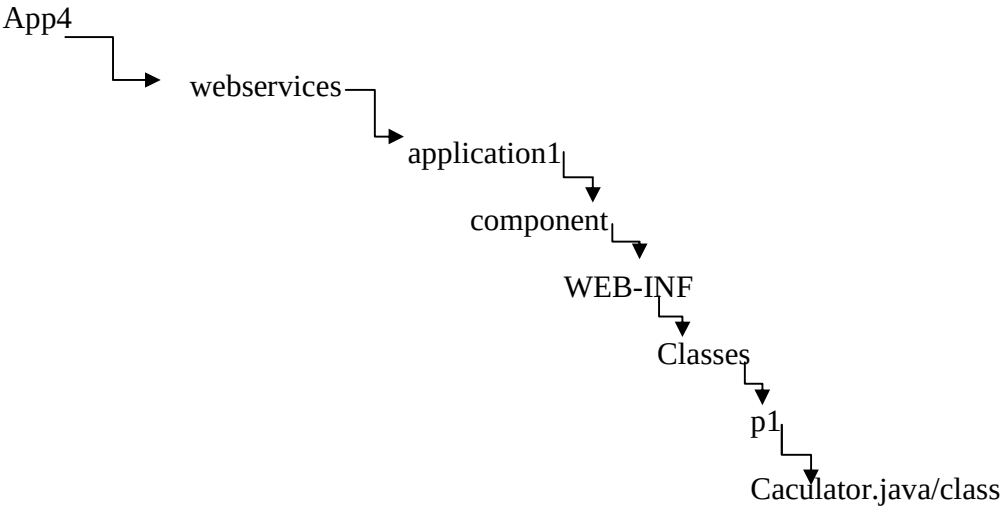
Collection.singleton(-), collection.singleton, Collection.SingletonMap(-,-) return immutable Set, List, Map object object respectively.

Web Services

@Resource Annotation of javax.annotation package makes underlying container/ Runtime environment to inject the resource to specified field.

Procedure:

1. First develop following structure



2. Now we have to create WAR file.

C:\WebServicesPrograms\Apps4\webservices\application1\component> jar cfv calc.war .

3. Now we have to generate proxy classes. To generate proxy classes go to C:\WebServicesPrograms\Apps4\webservices\application1\component location in command prompt and then type

C:\WebServicesPrograms\Apps4\webservices\application1\component> wsimport –p mypack –keep http://localhost:2222/calc/CalculatorService?wsdl.

4. Now proxy classes generated with mypack package. We have to develop right not client application(TestClient) in the following location

C:\WebServicesPrograms.

5. Now compile and run the application

   javac –d . *.java

   java TestClient

For complete program please refer web services\Programs\WebService security\Apps4 folder (or)

web services\NetBeans 6.7.1 projects\Web service security\Apps4 folder.

SO4: It is a specification containing rules and guidelines to develop XML, HTTP based interoperable distributed technology. Web Services is the technology whose implementation can be done either in Java or .NET based SO4 specification distributed technology.

Axis2 Framework to develop Web Services Components and Clients:

Name: AXIS2

Version: 2.X (compatible with JDK1.5+)

Vendor: Apache

Type of Software: Open Source

Download Axis2-1.6.2.zip and also download Axis2-1.6.2.war.zip these two files are available in www.axis.apahe.org/axis2/java/core/download.cgi#a1_5_1

For complete procedure please refer webservices\Programs\AXIS Web Services\AXIS2 installation procedure.docx document.

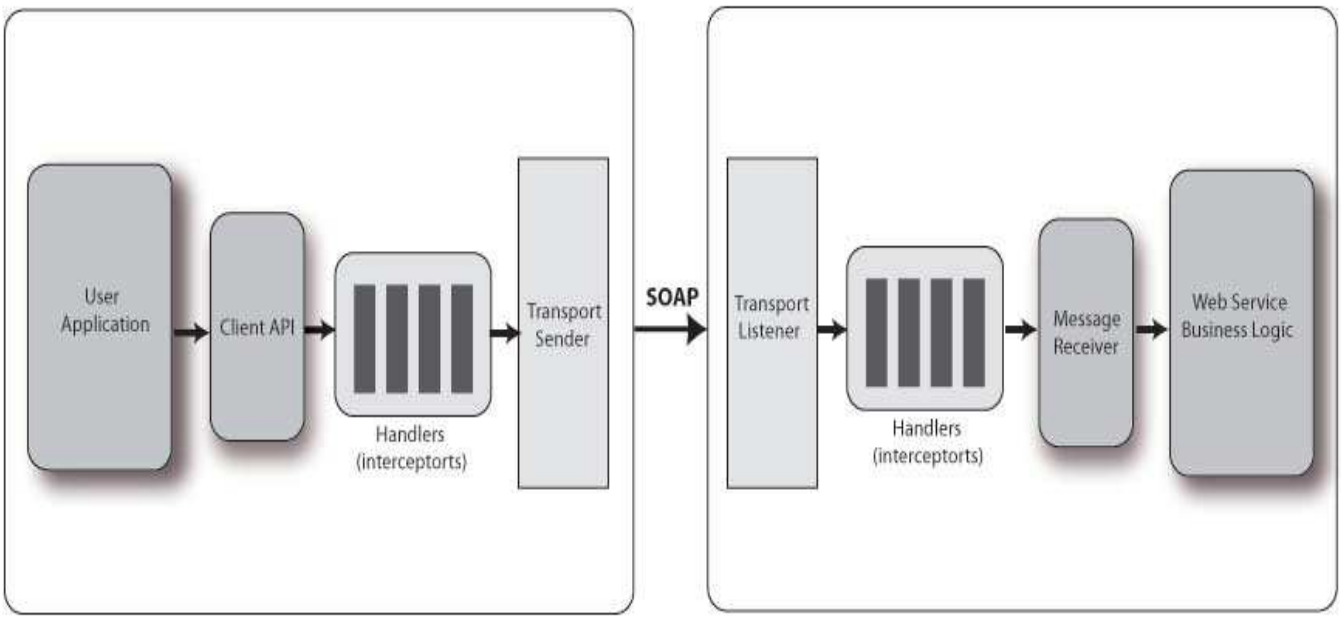
By using Axis2 we can perform following operations.

For Apache documentation for Axis2 please refer webservices\Programs\AXIS Web Services\AXIS2 documentation.docx document

1. Send SOAP messages
2. Receive SOAP messages and process SOAP messages
3. Create a Web Service component from a plain Java class.
4. Create a Web Service component from a WSDL document.
5. Generate WSDL document for existing component and etc.,

Diagram that shows SOAP messages flow in AXIS2 Application:

Web Services

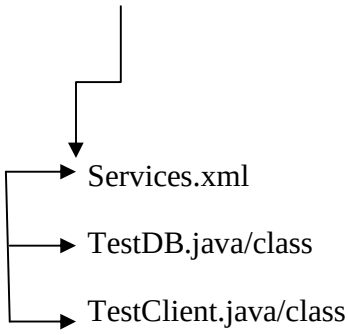


Each AXIS2 Web Service component comes as .aar file (Axis Archive File)

Procedure to develop AXIS2 Component:

Create a Deployment directory structure of Web Service components as shown below

C:\ App5\webservices\Axis2\Application1\Component\META-INF



For complete program please refer D:\Frameworks\webservices\Programs\AXIS Web Services\Apps5 folder

Now start Tomcat Server and ensure Axis2 is available or not. Just type
http://localhost:5050/Axis2

Now make one .aar file.

To make .aar file just type:
WebServicesPrograms\Apps5\webservices\axis2\application1\component>jar cfv bsrservice2.aar .

Now .aar file will come. Place the .aar file in C:\Program Files\Apache Software Foundation\Tomcat 6.0\webapps\axis2\WEB-INF\services location.

Now open browser type http://localhost:5050/axis2/services/BsrComponent?wsdl

Now you will get one WSDL document on browser.

Developing Client Application

1. Set following path environmental variables.

Paths:

C:\AXIS\axis2-1.6.2 (Axis_Home)

C:\AXIS\axis2-1.6.2\bin

C:\Program Files\Java\jdk1.6.0_18 (Java_Home)

2. Procedure to generate Proxy classes

C:\ WebServicesPrograms\App5\webservices\Axis2\Application1

BsrClientComponent

C:\WebServicesPrograms\Apps5\webservices\axis2\application1>wsdl2java –uri http://localhost:5050/axis2/services/BsrComponent?wsdl -0 BsrClientComponent

3. Make sure that Tomcat Server is running mode or not. The above tool generate some proxy classes in BsrClientComponent folder
4. Develop the Client application for above Web Service component.
5. Add following 15 jar files to classpath

i. axiom-api-1.2.13
ii. axiom-impl-1.2.13
iii. axis2-adb-1.6.2
iv. axis2-kernel-1.6.2
v. axis2-metadata-1.6.2
vi. axis2-transport-http-1.6.2
vii. axis2-transport-local-1.6.2
viii. commons-codec-1.3
ix. commons-httpclient-3.1
x. commons-logging-1.1.1

xi. httpcore-4.0
xii. mail-1.4
xiii. neethi-3.0.2
xiv. wsdl4j-1.6.2
xv. XmlSchema-1.4.7

Compile the Client application and run it.

Note1: We can develop .NET based Client applications for Axis2 Web Service Component.

Note2: For Axis based Component we can develop Client application in JAXWS.

Note3: webservice<component classname>+ stub is the proxy class name for every business method.

Procedure to develop Axis Web Service Component and Client by using Eclipse Helios or MyEclipse 9+:

For complete procedure please refer webservices\Eclipse Helios Projects\AXIS2 Project Document.docx

For Component please refer webservices\Eclipse Helios Projects\AxisComponent2

For Client Application please refer D:\Frameworks\webservices\Eclipse Helios Projects\ClientProj2.

Procedure to develop Axis Web Service Component and Client(JDBC Application) by using Eclipse Helios or MyEclipse 9+:

For Component please refer webservices\Eclipse Helios Projects\AxisComponent2 folder

For Client program please refer D:\Frameworks\webservices\Eclipse Helios Projects\ClientProj2 folder.

For complete procedure please refer D:\Frameworks\webservices\Programs\AXIS Web Services\Eclipse Helios Projects\Procedure to develop BscClientComponent1 and ClientApplication.docx

RESTFUL Web Services:

REST is a not protocol, REST is a methodology to develop Web Services by directly appending XML data to http request, http response with out using SOAP.

RE - represent Represental

ST – State

In SOAP based Web Services, the http Request, Response body contains SOAP request, response message respectively but the message contains more data and tags satisfying http protocol structure. This improves the burden on body of **http request, http response.** To see the difference between SOAP request, SOAP message and REST style XML message.

REST Style:

<weather>

<city> Hyderabad</city>

<date>11-10-12</date>

<high>85</high>

<low>70</low>

</weather>

SOAP based

<SOAP-ENV:Envelope xmlns:xsd=http://www.w3.org/2001/XMLSchema

Xmlns:SOAP-ENV=http://schemas.xml.soap.org/soap/envelope/

Xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance>

<SOAP-ENV:Body>

<nsl:getCityWeather xmlns:nsl=”Weather”>

<opl xsi:type=”xsd:string”>Hyderabad</opl>

</nsl:getCityWeather>

</SOAP-ENV:Body>

</SOAP-ENV:Envelope>

REST Basics:

Overview: REST is an architectural style which is based on web-standards and the HTTP protocol. REST was first described by Roy Fielding in 2000.

Web Services

1. In REST based architecture everything is a response. A resource is accessed via a common interface based on the HTTP standard methods, In an REST architecture we typically have a REST server which provides access to the resources and a REST client which accesses and modify REST resources. Every resource should support the HTTP common operations. Resources are identified by global ID's (Which are typically URIs)
2. REST allows that resources have different responsibilities, ex: text, XML, JSON etc., The REST Client can be ask for the specific representation via the HTTP protocol(Content Negotation)

HTTP Methods:

I. **GET:** Defines reading access of the resource without side-effects. The resource is never changed via a GET method request. **Ex:** The request has no side effects.
II. **PUT:** Create new resources. This operation must also be idempotent.
III. **DELETE:** It removes the resources.
IV. **POST:** Updates existing resources or creates a new resource.

Annotations	**Description**
1. @PATH(our_path)	Sets the path to base URL+ /your path. The base URL is based on your application name, the servlet and the URL pattern from the web.xml configuration file.
2. @POST	Indicates that following method will answer to a HTTP POST request.
3. @GET	Indicates that following method will answer to a HTTP GET request.
4. @PUT	Indicates that following method will answer to a HTTP PUT request.
5. @DELETE	Indicates that following method will answer to a HTTP DELTE request.
6. @Produces(MediaTypes.TEXT_PLAN [,more –types])	It defines which MIME type is delivered by a method annotated with @GET. In the example text("text/plain") is produced. Other examples would be "application/xml" or "application/json"
7. @Consumes(type[, more-types])	It defines MIME type is consumed by this method.
8. @PathParam	Used to inject values from the URL into a

	method parameter. This way we inject for example the ID of a resource into the method to get the correct object.

Q) What is the difference between SOAP style Web Service and REST style Web Service.

SOAP Style	REST Style
1. Uses SOAP over Http Protocol 2. Http Request, Response body contains SOAP message based XML 3. Suitable for complex Web Services 4. Allows to work with Middleware Services 5. Use JAXWS, JAXRPC, AXIS and etc., APIs to develop these Web Services 6. Can generate only XML response. 7. **EX:** Stock broker component available in sharekhan.com 8. Service interface is WSDL document.	1. Uses Http Protocol 2. Http Request, Response body contains direct XML 3. Suitable for small and medium Web Services 4. Does not Allows to work with Middleware Services 5. Use JAXRS API to develop these Web Services 6. Can generate only XML,HTML,JSON response. 7. **EX:** Components available in Amazon, ebay etc., 8. Suitable interface is WADL

Procedure to develop RESTFUL Web Service component in NetBeans IDE:

For complete procedure please refer webservices\NetBeans 7.1 Projects\RESTFUL Applications\RESTFUL Web Services Component and Client creation.docx

For Component creation please refer webservices\NetBeans 7.1 Projects\RESTFUL Applications\RESTComponentBhargav folder

For Client creation please refer webservices\NetBeans 7.1 Projects\RESTFUL Applications\RESTClientBhargav folder.

When multiple methods are there in RESTful component we can identify them either by using base path or separate path. Base path will be given on the top of the class. Separate path will be given on the top of the business method using @Path Annotation.

EX: RESTful Web Service Component:

@Path(“mypath”)

public class TestRS

```
{
@GET
@Procedures(“text/html”);
pubic String bm1()
{
-----
------
}
@GET
@Produces(“application/xml”);
@Path(“{n1}”);
public String bm1(@PathParam(“n1”) int a)
{
-------
-------
}
@GET
@Procedures(“application/xml”);
@Path(“m1/{p1}/{p2}”);
public String bm1@PathParam(“{n1}”) int a,@PathParam(“{n2}”) int b)
{
----------
----------
}
@POST
```

@Produces(“application/json”);

@Path(“{m2}”);

public String bm2()

{

}

@POST

@Produces(“application/json”);

@Path(“m2/{n1}”)

public String bm2@PathParam(“n1”) int a)

{

}

}

urls from Client application

http://localhost:8888/RestProj1/resources/mypath

Calls bm1()

http://localhost:8888/RestProj1/resources/mypath/20

Calls bm1(-)

http://localhost:8888/RestProj1/resources/mypath/m1/20/30

Calls bm1(-,-)

http://localhost:8888/RestProj1/resources/mypath/m2

Calls bm2()

http://localhost:8888/RestProj1/resources/mypath/m2/56

Calls bm2(-)

For complete program please refer D:\Frameworks\webservices\NetBeans 7.1 Projects\RESTFUL Applications\RESTfulUrlClientComponent folder.

Procedure to develop RESTful Web Service Component dealing with XML content by using MyEclipse IDE.

For complete procedure please refer webservices\RESTFUL WebService Dealing with XML.docx.

For Component creation Program please refer webservices\MyEclipse 8.X Projects\RESTComponentDealingWithXML folder

For complete Client Application please refer D:\Frameworks\webservices\MyEclipse 8.X Projects\ClientProjectForRESTComponentDealingWithXml folder

Procedure to develop RESTful Web Service Component dealing with JSON content by using MyEclipse IDE.

For component Application please refer webservices\MyEclipse 8.X Projects\RESTComponent10

For Client Application please refer webservices\MyEclipse 8.X Projects\ClientBhargavProj1 folder

Spring Web Services:

Spring Web Services is the concept of Spring JEE module which specifies the process of developing Spring Web Services component and Client Application development.

Spring Web Services gives org.springframework.remoting.jaxws.SimepleJaxWSServiceExporterBean to convert given components into Web Service component similarly the org.springframework.remoting.jaxws.JaxWsProxyFactoryBean is given to give business object reference to Client application using which business method is called.

Jars required for complete this project

1. aopalliance-1.0.jar""C:\Spring jars\Jars\commons-logging.jar
2. org.springframework.aop-sources-3.0.5.RELEASE.jar
3. org.springframework.asm-sources-3.0.5.RELEASE.jar
4. org.springframework.beans-sources-3.0.5.RELEASE.jar

5. org.springframework.context.support-sources-3.0.5.RELEASE.jar
6. C:\Spring jars\Jars\org.springframework.context-sources-3.0.5.RELEASE.jar
7. C:\Spring jars\Jars\org.springframework.core-sourc3.0.5.RELEASE.jar
8. C:\Spring jars\Jars\org.springframework.expression-sources-3.0.5.RELEASE.jar
9. C:\Spring jars\Jars\org.springframework.web.portlet-sources-3.0.5.RELEASE.jar
10. C:\Spring jars\Jars\org.springframework.web.servlet-sources-3.0.5.RELEASE.jar
11. C:\Spring jars\Jars\org.springframework.web-sources-3.0.5.RELEASE.jar
12. spring.jar
13. commons-logging.jar. } These two only available in Spring2.X, remaining are available in Spring3.X software location for 3.X jars is spring-framework-3.0.5.RELEASE\dist

For example Application on Component creation Please refer webservices\MyEclipse 8.X Projects\SpringWebServicesProjectComponent folder.

Client application

Client Application execution is still pending.