Kaunas University of technologY

Department of Informatics

Distributed Systems Lab 1 Part B

Create a distributed system for data search using Webservice

|  |  |
| --- | --- |
| **Student:** | **Muruganantham Jaisankar** I M. Sc student  **Faculty of Informatics**  [muruganantham.jaisankar@ktu.edu](mailto:name.surname@ktu.edu)  en.ktu.edu |

**22-11-2021, Kaunas.**

**Web Service/SOAP**

Aim of the lab to distribute the data to different servers and retrieving the data for requested Client.

**Architecture to implement**

**Diagram

Description automatically generated**

Each servers having 5 numbers, if server 1 don’t have the number then it will go to another server 2, if server 2 having the number then it will return to server 1, then the server 1 send to the client.

**Java Server pages File**

**Client server number distribution**

<%@page contentType=*"text/html;charset=UTF-8"*%><HTML>

<HEAD>

<TITLE>Client-Server number distribution </TITLE>

</HEAD>

<BODY>

<H1>Client-Server number distribution </H1>

<UL>

<LI><A HREF=*"Input.jsp?method=13"* TARGET=*"inputs"*> Ask Number</A></LI>

</UL>

</BODY>

</HTML>

**Ask server**

<%@page contentType=*"text/html;charset=UTF-8"*%>

<HTML>

<HEAD>

<TITLE>Enter the number to ask from Server</TITLE>

</HEAD>

<BODY>

<H1>Enter the number to ask from Server</H1>

<%

String method = request.getParameter("method");

**int** methodID = 0;

**if** (method == **null**) methodID = -1;

**boolean** valid = **true**;

**if**(methodID != -1) methodID = Integer.parseInt(method);

**switch** (methodID){

**case** 13:

valid = **false**;

%>

<FORM METHOD=*"POST"* ACTION=*"Result.jsp"* TARGET=*"result"*>

<INPUT TYPE=*"HIDDEN"* NAME=*"method"* VALUE=*"*<%=org.eclipse.jst.ws.util.JspUtils.markup(method)%>*"*>

<TABLE>

<TR>

<TD COLSPAN=*"1"* ALIGN=*"LEFT"*>a:</TD>

<TD ALIGN=*"left"*><INPUT TYPE=*"TEXT"* NAME=*"a16"* SIZE=*20*></TD>

</TR>

</TABLE>

<BR>

<INPUT TYPE=*"SUBMIT"* VALUE=*"Invoke"*>

<INPUT TYPE=*"RESET"* VALUE=*"Clear"*>

</FORM>

<%

**break**;

}

**if** (valid) {

%>

Select a method to test.

<%

}

%>

</BODY>

</HTML>

**Result**

<%@page contentType=*"text/html;charset=UTF-8"*%>

<% request.setCharacterEncoding("UTF-8"); %>

<HTML>

<HEAD>

<TITLE>Result</TITLE>

</HEAD>

<BODY>

<H1>Result</H1>

<jsp:useBean id=*"sampleServiceProxyid"* scope=*"session"* class=*"com.ServiceProxy"* />

<%

**if** (request.getParameter("endpoint") != **null** && request.getParameter("endpoint").length() > 0)

sampleServiceProxyid.setEndpoint(request.getParameter("endpoint"));

%>

<%

String method = request.getParameter("method");

**int** methodID = 0;

**if** (method == **null**) methodID = -1;

**if**(methodID != -1) methodID = Integer.parseInt(method);

**boolean** gotMethod = **false**;

**try** {

**switch** (methodID){

**case** 13:

gotMethod = **true**;

String a\_1id= request.getParameter("a16");

**int** a\_1idTemp = Integer.parseInt(a\_1id);

java.lang.String func13mtemp = sampleServiceProxyid.func(a\_1idTemp);

**if**(func13mtemp == **null**){

%>

<%=func13mtemp %>

<%

}**else**{

String tempResultreturnp14 = org.eclipse.jst.ws.util.JspUtils.markup(String.valueOf(func13mtemp));

%>

<%= tempResultreturnp14 %>

<%

}

**break**;

}

} **catch** (Exception e) {

%>

Exception: <%= org.eclipse.jst.ws.util.JspUtils.markup(e.toString()) %>

Message: <%= org.eclipse.jst.ws.util.JspUtils.markup(e.getMessage()) %>

<%

**return**;

}

**if**(!gotMethod){

%>

result: N/A

<%

}

%>

</BODY>

</HTML>

**WSDL file**

**Server1**

**package** groupA;

**public** **interface** Service **extends** java.rmi.Remote {

**public** java.lang.String func(**int** a) **throws** java.rmi.RemoteException;

}

**ServerProxy**

**package** groupA;

**public** **class** ServiceProxy **implements** com.Service {

**private** String \_endpoint = **null**;

**private** com.Service service = **null**;

**public** ServiceProxy() {

\_initServiceProxy();

}

**public** ServiceProxy(String endpoint) {

\_endpoint = endpoint;

\_initServiceProxy();

}

**private** **void** \_initServiceProxy() {

**try** {

service = (**new** com.ServiceServiceLocator()).getservice();

**if** (service != **null**) {

**if** (\_endpoint != **null**)

((javax.xml.rpc.Stub)service).\_setProperty("javax.xml.rpc.service.endpoint.address", \_endpoint);

**else**

\_endpoint = (String)((javax.xml.rpc.Stub)service).\_getProperty("javax.xml.rpc.service.endpoint.address");

}

}

**catch** (javax.xml.rpc.ServiceException serviceException) {}

}

**public** String getEndpoint() {

**return** \_endpoint;

}

**public** **void** setEndpoint(String endpoint) {

\_endpoint = endpoint;

**if** (service != **null**)

((javax.xml.rpc.Stub)service).\_setProperty("javax.xml.rpc.service.endpoint.address", \_endpoint);

}

**public** com.Service getService() {

**if** (service == **null**)

\_initServiceProxy();

**return** service;

}

**public** java.lang.String func(**int** a) **throws** java.rmi.RemoteException{

**if** (service == **null**)

\_initServiceProxy();

**return** service.func(a);

}

}

**Server1 Service**

**package** com;

**public** **interface** ServiceService **extends** javax.xml.rpc.Service {

**public** java.lang.String getserviceAddress();

**public** com.Service getservice() **throws** javax.xml.rpc.ServiceException;

**public** com.Service getservice(java.net.URL portAddress) **throws** javax.xml.rpc.ServiceException;

}

**Server Service Locator**

**package** com;

**public** **class** ServiceServiceLocator **extends** org.apache.axis.client.Service **implements** com.ServiceService {

**public** ServiceServiceLocator() {

}

**public** ServiceServiceLocator(org.apache.axis.EngineConfiguration config) {

**super**(config);

}

**public** ServiceServiceLocator(java.lang.String wsdlLoc, javax.xml.namespace.QName sName) **throws** javax.xml.rpc.ServiceException {

**super**(wsdlLoc, sName);

}

// Use to get a proxy class for service

**private** java.lang.String service\_address = "http://localhost:91/client/services/service";

**public** java.lang.String getserviceAddress() {

**return** service\_address;

}

// The WSDD service name defaults to the port name.

**private** java.lang.String serviceWSDDServiceName = "service";

**public** java.lang.String getserviceWSDDServiceName() {

**return** serviceWSDDServiceName;

}

**public** **void** setserviceWSDDServiceName(java.lang.String name) {

serviceWSDDServiceName = name;

}

**public** com.Service getservice() **throws** javax.xml.rpc.ServiceException {

java.net.URL endpoint;

**try** {

endpoint = **new** java.net.URL(service\_address);

}

**catch** (java.net.MalformedURLException e) {

**throw** **new** javax.xml.rpc.ServiceException(e);

}

**return** getservice(endpoint);

}

**public** com.Service getservice(java.net.URL portAddress) **throws** javax.xml.rpc.ServiceException {

**try** {

com.ServiceSoapBindingStub \_stub = **new** com.ServiceSoapBindingStub(portAddress, **this**);

\_stub.setPortName(getserviceWSDDServiceName());

**return** \_stub;

}

**catch** (org.apache.axis.AxisFault e) {

**return** **null**;

}

}

**public** **void** setserviceEndpointAddress(java.lang.String address) {

service\_address = address;

}

/\*\*

\* For the given interface, get the stub implementation.

\* If this service has no port for the given interface,

\* then ServiceException is thrown.

\*/

**public** java.rmi.Remote getPort(Class serviceEndpointInterface) **throws** javax.xml.rpc.ServiceException {

**try** {

**if** (com.Service.**class**.isAssignableFrom(serviceEndpointInterface)) {

com.ServiceSoapBindingStub \_stub = **new** com.ServiceSoapBindingStub(**new** java.net.URL(service\_address), **this**);

\_stub.setPortName(getserviceWSDDServiceName());

**return** \_stub;

}

}

**catch** (java.lang.Throwable t) {

**throw** **new** javax.xml.rpc.ServiceException(t);

}

**throw** **new** javax.xml.rpc.ServiceException("There is no stub implementation for the interface: " + (serviceEndpointInterface == **null** ? "null" : serviceEndpointInterface.getName()));

}

/\*\*

\* For the given interface, get the stub implementation.

\* If this service has no port for the given interface,

\* then ServiceException is thrown.

\*/

**public** java.rmi.Remote getPort(javax.xml.namespace.QName portName, Class serviceEndpointInterface) **throws** javax.xml.rpc.ServiceException {

**if** (portName == **null**) {

**return** getPort(serviceEndpointInterface);

}

java.lang.String inputPortName = portName.getLocalPart();

**if** ("service".equals(inputPortName)) {

**return** getservice();

}

**else** {

java.rmi.Remote \_stub = getPort(serviceEndpointInterface);

((org.apache.axis.client.Stub) \_stub).setPortName(portName);

**return** \_stub;

}

}

**public** javax.xml.namespace.QName getServiceName() {

**return** **new** javax.xml.namespace.QName("http://com", "serviceService");

}

**private** java.util.HashSet ports = **null**;

**public** java.util.Iterator getPorts() {

**if** (ports == **null**) {

ports = **new** java.util.HashSet();

ports.add(**new** javax.xml.namespace.QName("http://com", "service"));

}

**return** ports.iterator();

}

/\*\*

\* Set the endpoint address for the specified port name.

\*/

**public** **void** setEndpointAddress(java.lang.String portName, java.lang.String address) **throws** javax.xml.rpc.ServiceException {

**if** ("service".equals(portName)) {

setserviceEndpointAddress(address);

}

**else**

{ // Unknown Port Name

**throw** **new** javax.xml.rpc.ServiceException(" Cannot set Endpoint Address for Unknown Port" + portName);

}

}

/\*\*

\* Set the endpoint address for the specified port name.

\*/

**public** **void** setEndpointAddress(javax.xml.namespace.QName portName, java.lang.String address) **throws** javax.xml.rpc.ServiceException {

setEndpointAddress(portName.getLocalPart(), address);

}

}

**Server SOAP Binding Stub**

**package** com;

**public** **class** ServiceSoapBindingStub **extends** org.apache.axis.client.Stub **implements** com.Service {

**private** java.util.Vector cachedSerClasses = **new** java.util.Vector();

**private** java.util.Vector cachedSerQNames = **new** java.util.Vector();

**private** java.util.Vector cachedSerFactories = **new** java.util.Vector();

**private** java.util.Vector cachedDeserFactories = **new** java.util.Vector();

**static** org.apache.axis.description.OperationDesc [] *\_operations*;

**static** {

*\_operations* = **new** org.apache.axis.description.OperationDesc[1];

*\_initOperationDesc1*();

}

**private** **static** **void** \_initOperationDesc1(){

org.apache.axis.description.OperationDesc oper;

org.apache.axis.description.ParameterDesc param;

oper = **new** org.apache.axis.description.OperationDesc();

oper.setName("func");

param = **new** org.apache.axis.description.ParameterDesc(**new** javax.xml.namespace.QName("http://com", "a"), org.apache.axis.description.ParameterDesc.*IN*, **new** javax.xml.namespace.QName("http://www.w3.org/2001/XMLSchema", "int"), **int**.**class**, **false**, **false**);

oper.addParameter(param);

oper.setReturnType(**new** javax.xml.namespace.QName("http://www.w3.org/2001/XMLSchema", "string"));

oper.setReturnClass(java.lang.String.**class**);

oper.setReturnQName(**new** javax.xml.namespace.QName("http://com", "funcReturn"));

oper.setStyle(org.apache.axis.constants.Style.*WRAPPED*);

oper.setUse(org.apache.axis.constants.Use.*LITERAL*);

*\_operations*[0] = oper;

}

**public** ServiceSoapBindingStub() **throws** org.apache.axis.AxisFault {

**this**(**null**);

}

**public** ServiceSoapBindingStub(java.net.URL endpointURL, javax.xml.rpc.Service service) **throws** org.apache.axis.AxisFault {

**this**(service);

**super**.cachedEndpoint = endpointURL;

}

**public** ServiceSoapBindingStub(javax.xml.rpc.Service service) **throws** org.apache.axis.AxisFault {

**if** (service == **null**) {

**super**.service = **new** org.apache.axis.client.Service();

} **else** {

**super**.service = service;

}

((org.apache.axis.client.Service)**super**.service).setTypeMappingVersion("1.2");

}

**protected** org.apache.axis.client.Call createCall() **throws** java.rmi.RemoteException {

**try** {

org.apache.axis.client.Call \_call = **super**.\_createCall();

**if** (**super**.maintainSessionSet) {

\_call.setMaintainSession(**super**.maintainSession);

}

**if** (**super**.cachedUsername != **null**) {

\_call.setUsername(**super**.cachedUsername);

}

**if** (**super**.cachedPassword != **null**) {

\_call.setPassword(**super**.cachedPassword);

}

**if** (**super**.cachedEndpoint != **null**) {

\_call.setTargetEndpointAddress(**super**.cachedEndpoint);

}

**if** (**super**.cachedTimeout != **null**) {

\_call.setTimeout(**super**.cachedTimeout);

}

**if** (**super**.cachedPortName != **null**) {

\_call.setPortName(**super**.cachedPortName);

}

java.util.Enumeration keys = **super**.cachedProperties.keys();

**while** (keys.hasMoreElements()) {

java.lang.String key = (java.lang.String) keys.nextElement();

\_call.setProperty(key, **super**.cachedProperties.get(key));

}

**return** \_call;

}

**catch** (java.lang.Throwable \_t) {

**throw** **new** org.apache.axis.AxisFault("Failure trying to get the Call object", \_t);

}

}

**public** java.lang.String func(**int** a) **throws** java.rmi.RemoteException {

**if** (**super**.cachedEndpoint == **null**) {

**throw** **new** org.apache.axis.NoEndPointException();

}

org.apache.axis.client.Call \_call = createCall();

\_call.setOperation(*\_operations*[0]);

\_call.setUseSOAPAction(**true**);

\_call.setSOAPActionURI("");

\_call.setEncodingStyle(**null**);

\_call.setProperty(org.apache.axis.client.Call.*SEND\_TYPE\_ATTR*, Boolean.*FALSE*);

\_call.setProperty(org.apache.axis.AxisEngine.*PROP\_DOMULTIREFS*, Boolean.*FALSE*);

\_call.setSOAPVersion(org.apache.axis.soap.SOAPConstants.*SOAP11\_CONSTANTS*);

\_call.setOperationName(**new** javax.xml.namespace.QName("http://com", "func"));

setRequestHeaders(\_call);

setAttachments(\_call);

**try** { java.lang.Object \_resp = \_call.invoke(**new** java.lang.Object[] {**new** java.lang.Integer(a)});

**if** (\_resp **instanceof** java.rmi.RemoteException) {

**throw** (java.rmi.RemoteException)\_resp;

}

**else** {

extractAttachments(\_call);

**try** {

**return** (java.lang.String) \_resp;

} **catch** (java.lang.Exception \_exception) {

**return** (java.lang.String) org.apache.axis.utils.JavaUtils.*convert*(\_resp, java.lang.String.**class**);

}

}

} **catch** (org.apache.axis.AxisFault axisFaultException) {

**throw** axisFaultException;

}

}

}

**Server java file**

**Group A**

**Server 1**

**package** groupA;

**public** **class** server1 {

**public** String func(**int** a)

{

**int** array[]={0,1,2,3,4};

**for**(**int** i=0;i<5;i++)

{

**if**(array[i]==a)

{

**return** "From Group A server 1 number is "+a;

}

}

Server2 serv2=**new** Server2();

String str=serv2.func2(a);

**return** str;

}

}

**Server 2**

**package** groupA;

**public** **class** Server2 {

**public** String func2(**int** a)

{

**int** array[]={5,6,7,8,9};

**for**(**int** i=0;i<5;i++)

{

**if**(array[i]==a)

{

**return** "From Group A server 2 number is "+a;

}

}

Server3 serv3=**new** Server3();

String str=serv3.func3(a);

**return** str;

}

}

**Server 3**

**package** groupA;

**import** groupB.service;

**public** **class** Server3 {

**public** String func3(**int** a)

{

**int** array[]={10,11,12,13,14};

**for**(**int** i=0;i<5;i++)

{

**if**(array[i]==a)

{

**return** "From Group A server 3 number is "+a;

}

}

groupB.service groupBserver1=**new** groupB.service();

String str=groupBserver1.func(a);

**return** str;

}

}

**Group B**

**Server 1**

**package** groupB;

**public** **class** server1 {

**public** String func(**int** a)

{

**int** array[]={15,16,17,18,19};

**for**(**int** i=0;i<5;i++)

{

**if**(array[i]==a)

{

**return** "From Group B server 1 number is "+a;

}

}

Server2 serv2=**new** Server2();

String str=serv2.func2(a);

**return** str;

}

}

**Server 2**

**package** groupB;

**public** **class** Server2 {

**public** String func2(**int** a)

{

**int** array[]={20,21,22,23,24};

**for**(**int** i=0;i<5;i++)

{

**if**(array[i]==a)

{

**return** "From Groub B server 2 number is "+a;

}

}

Server3 serv3=**new** Server3();

String str=serv3.func3(a);

**return** str;

}

}

**Server 3**

**package** groupB;

**public** **class** Server3 {

**public** String func3(**int** a)

{

**int** array[]={25,26,27,28,29};

**for**(**int** i=0;i<5;i++)

{

**if**(array[i]==a)

{

**return** "From Groub B server 3 number is "+a;

}

}

**return** "No number From Groub A and Group B servers 1,2,3";

}

}

**Sample output**

Graphical user interface, text, application, Word

Description automatically generated