PRACTICAL 10

Aim: Write a program to implement business UDDI Registry entry

UDDI stands for Universal Description, Discovery, and Integration. The UDDI Project is an industry initiative aims to enable businesses to quickly, easily, and dynamically find and carry out transactions with one another.

A populated UDDI registry contains cataloged information about businesses; the services that they offer; and communication standards and interfaces they use to conduct transactions.

Built on the Simple Object Access Protocol (SOAP) data communication standard, UDDI creates a global, platform-independent, open architecture space that will benefit businesses.

The UDDI registry can be broadly divided into two categories:

UDDI and Web Services

UDDI and Business Registry

For details about the UDDI data structure, see UDDI Data Structure.

UDDI and Web Services

The owners of Web services publish them to the UDDI registry. Once published, the UDDI registry maintains pointers to the Web service description and to the service.

The UDDI allows clients to search this registry, find the intended service, and retrieve its details. These details include the service invocation point as well as other information to help identify the service and its functionality.

Web service capabilities are exposed through a programming interface, and usually explained through Web services Description Language (WSDL). In a typical publish-and-inquire scenario, the provider publishes its business; registers a service under it; and defines a binding template with technical information on its Web service. The binding template also holds reference to one or several tModels, which represent abstract interfaces implemented by the Web service. The tModels might have been uniquely published by the provider, with information on the interfaces and URL references to the WSDL document.

A typical client inquiry may have one of two objectives:

- To find an implementation of a known interface. In other words, the client has a tModel ID and seeks binding templates referencing that tModel.

- To find the updated value of the invocation point (that is., access point) of a known binding template ID

UDDI and Business Registry

As a Business Registry solution, UDDI enables companies to advertise the business products and services they provide, as well as how they conduct business transactions on the Web. This use of UDDI complements business-to-business (B2B) electronic commerce.

The minimum required information to publish a business is a single business name. Once completed, a full description of a business entity may contain a wealth of information, all of which helps to advertise the business entity and its products and services in a precise and accessible manner.

A Business Registry can contain:

- Business Identification - Multiple names and descriptions of the business, comprehensive contact information, and standard business identifiers such as a tax identifier.
- Categories - Standard categorization information (for example a D-U-N-S business category number).
- Service Description - Multiple names and descriptions of a service. As a container for service information, companies can advertise numerous services, while clearly displaying the ownership of services. The bindingTemplate information describes how to access the service.
- Standards Compliance - In some cases it is important to specify compliance with standards. These standards might display detailed technical requirements on how to use the service.
- Custom Categories - It is possible to publish proprietary specifications (tModels) that identify or categorize businesses or services.

UDDI Data Structure

The data structure within UDDI consists of four constructions: a businessEntity structure, a businessService structure, a bindingTemplate structure and a tModelstructure.

Code:

```
import java.io.*;
importjava.util.*;
```

```
 public class UDDISoapClient
{
   // Default values used if no command line parameters are set
private static final String DEFAULT_HOST_URL =
                        "http://localhost:8080/wasp/uddi/inquiry/";
private static final String DEFAULT_DATA_FILENAME   = "./Default.xml";

   // In the SOAP chapter, we used "urn:oreilly:jaws:samples",
   // but Systinet UDDI requires this to be blank.
private static final String URI                  = "";
private String m_hostURL;
private String m_dataFileName;

publicUDDISoapClient(String hostURL, String dataFileName) throws Exception
{
m_hostURL = hostURL;
m_dataFileName    = dataFileName;

System.out.println(  );
System.out.println("_____________________________________________");
System.out.println("Starting UDDISoapClient:");
System.out.println("    host url       = " + m_hostURL);
System.out.println("    data file      = " + m_dataFileName);
System.out.println("_____________________________________________");
System.out.println(  );
   }

public void sendSOAPMessage(  ) {
try {
        // Get soap body to include in the SOAP envelope from FILE
FileReaderfr = new FileReader (m_dataFileName);
javax.xml.parsers.DocumentBuilderxdb =
org.apache.soap.util.xml.XMLParserUtils.getXMLDocBuilder(  );
org.w3c.dom.Document doc =
xdb.parse (new org.xml.sax.InputSource (fr));
if (doc == null) {
throw new org.apache.soap.SOAPException
              (org.apache.soap.Constants.FAULT_CODE_CLIENT, "parsing error");
        }

        // Create a vector for collecting the body elements
        Vector bodyElements = new Vector(  );

        // Parse XML element as soap body element
```

```
bodyElements.add(doc.getDocumentElement ( ));
        // Create the SOAP envelope
org.apache.soap.Envelope envelope = new org.apache.soap.Envelope( );
envelope.declareNamespace("idoox", "http://idoox.com/uddiface");
envelope.declareNamespace("ua", "http://idoox.com/uddiface/account");
envelope.declareNamespace("config",
          "http://idoox.com/uddiface/config");
envelope.declareNamespace("attr", "http://idoox.com/uddiface/attr");
envelope.declareNamespace("fxml", "http://idoox.com/uddiface/formxml");
envelope.declareNamespace("inner", "http://idoox.com/uddiface/inner");
envelope.declareNamespace("", "http://idoox.com/uddiface/inner");
envelope.declareNamespace("uddi", "urn:uddi-org:api_v2");
        //
        // NO SOAP HEADER ELEMENT AS SYSTINET WASP DOES NOT REQUIRE IT
        //
        // Create the SOAP body element
org.apache.soap.Body body = new org.apache.soap.Body( );
body.setBodyEntries(bodyElements);
envelope.setBody(body);
        // Build and send the Message.
org.apache.soap.messaging.Messagemsg =
neworg.apache.soap.messaging.Message( );
msg.send (new java.net.URL(m_hostURL), URI, envelope);
System.out.println("Sent SOAP Message with Apache HTTP SOAP Client.");
        // Receive response from the transport and dump it to the screen
System.out.println("Waiting for response....");
org.apache.soap.transport.SOAPTransportst = msg.getSOAPTransport ( );
BufferedReaderbr = st.receive ( );

if(line == null) {
System.out.println("HTTP POST was unsuccessful. \n");
        } else {
while (line != null) {
System.out.println (line);
line = br.readLine( );
            }
        }
      /////
      // Version in examples has XML pretty printing logic here.
      ////
      } catch(Exception e) {
e.printStackTrace( );
      }
  }
```

```
    //
    // NOTE: the remainder of this deals with reading arguments
    //
    /** Main program entry point. */
public static void main(String args[]) {
     // Not Relevant
    }
}
```

Output:

```
Starting UDDISoapClient:
hosturl      = http://localhost:8080/wasp/uddi/inquiry/
data file     = Ch6_FindBusiness.xml
Sent SOAP Message with Apache HTTP SOAP Client.
Waiting for response....
 <?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelopexmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
     <SOAP-ENV:Body>
          <businessListxmlns="urn:uddi-org:api_v2" generic="2.0" operator="SYSTINET">
                    <businessInfos>
                        <businessInfobusinessKey="892ac280-c16b-11d5-85ad-801eef208714">
                             <name xml:lang="en">
                                          Demi Credit
                             </name>
                           <description xml:lang="en">
                               A smaller demo credit agency used for illustrating UDDI inquiry.
                           </description>
                         <serviceInfos>
                           <serviceInfo serviceKey="860eca90-c16d-11d5-85ad-801eef208714"
                              businessKey="9a26b6e0-c15f-11d5-85a3-801eef208714">
                               <name xml:lang="en">
                                    DCAmail
                               </name>
                           </serviceInfo>
                         </serviceInfos>
                    </businessInfo>
               </businessInfos>
          </businessList>
     </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```