WEB-BASED AIRLINE TICKET BOOKING SYSTEM

Yu Jianming, CSE

Problem in Lieu of Thesis Prepared for the Degree of

MASTER OF SCIENCE

UNIVERSITY OF NORTH TEXAS

August 2004

APPROVED:

Tom Jacob, Major Professor
Robert Brazil, Graduate Coordinator,
Krishna Kavi, Chair of the Computer Science and
Engineering Department
Oscar Garcia, Dean of the College of Engineering
Sandra L. Terrell, Interim Dean of the Robert B.
Toulouse School of Graduate Studies.

Online airline ticket booking system is one of the essential applications of E-commerce. With the development of Internet and security technology, more and more people begin to consume online, which is more convenient and personal than traditional way. The goal of this system is to make people purchase airline tickets easily. The system is written in JAVA™. Chapter 1 will introduce some basic conception of the technologies have been used in this system. Chapter 2 shows how the database and the system are designed. Chapter 3 shows the logic of the Web site. In Chapter 4 the interface of the system will be given. Chapter 5 tells the platform of this system.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>LIST OF TABLES</th>
<th>iii</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRADEMARKED NAMES USED</td>
<td>iv</td>
</tr>
</tbody>
</table>

## Chapter

1. **INTRODUCTION** ................................................................. 1  
   The Problem and Its Purposes .............................................. 1  
   Definition of Terms .......................................................... 1  

2. **THE SYSTEM AND DATABASE DESIGN** .............................. 3  
   System Design ......................................................................... 4  
   Database Design ....................................................................... 5  

3. **THE LOGIC OF THE WEB SITE** ........................................ 7  

4. **THE PLATFORM OF THIS SYSTEM AND CONFIGURATION** .......... 12  

5. **INTERFACE OF THE SYSTEM** ............................................. 13  
   System Management Part ....................................................... 13  
   Ticket Booking System on Web End ....................................... 15  

6. **PROGRAMS** .................................................................... 22  
   Common Object Request Broker Architecture (CORBA) ............ 22  
   Remote Method Invocation (RMI) ............................................ 22  
   Database .................................................................................. 24  
   Java Server Pages (JSP) ........................................................ 25  

| REFERENCE LIST | 28 |
# LIST OF TABLES

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The structure of the Web site</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>The tables of database</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>Main pages of Airline Ticket Booking System</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>Structure of client part of management of Airline Ticket Booking System</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>The server panel of RMI</td>
<td>13</td>
</tr>
<tr>
<td>6</td>
<td>Login form of the client part of RMI</td>
<td>13</td>
</tr>
<tr>
<td>7</td>
<td>The client part of RMI for Employee Information</td>
<td>14</td>
</tr>
<tr>
<td>8</td>
<td>The client part of RMI for Flight Information</td>
<td>14</td>
</tr>
<tr>
<td>9</td>
<td>The client part of RMI for Customer Information</td>
<td>15</td>
</tr>
<tr>
<td>10</td>
<td>Login form of the Java server pages (JSP) part</td>
<td>15</td>
</tr>
<tr>
<td>11</td>
<td>Personal Assistant of JSP part</td>
<td>16</td>
</tr>
<tr>
<td>12</td>
<td>Traveler Information Modification form of JSP part</td>
<td>16</td>
</tr>
<tr>
<td>13</td>
<td>Search form of JSP part</td>
<td>17</td>
</tr>
<tr>
<td>14</td>
<td>The search result of JSP part</td>
<td>18</td>
</tr>
<tr>
<td>15</td>
<td>Search summary of the JSP part</td>
<td>18</td>
</tr>
<tr>
<td>16</td>
<td>Customer input form of the JSP part</td>
<td>19</td>
</tr>
<tr>
<td>17</td>
<td>Credit card information input form of the JSP part</td>
<td>20</td>
</tr>
<tr>
<td>18</td>
<td>Saved itineraries of the JSP part</td>
<td>20</td>
</tr>
<tr>
<td>19</td>
<td>Detail of itinerary of the JSP part</td>
<td>21</td>
</tr>
<tr>
<td>20</td>
<td>Register form of the JSP part</td>
<td>21</td>
</tr>
<tr>
<td>Product Name</td>
<td>Owner</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>Java™, Java Server Pages(JSP™) and JavaScript™</td>
<td>Sun Microsystems, Inc., <a href="http://www.sun.com">www.sun.com</a></td>
<td></td>
</tr>
<tr>
<td>Jbuilder™ 9 Enterprise Trial</td>
<td>Borland Software Corporation, <a href="http://www.borland.com">www.borland.com</a></td>
<td></td>
</tr>
<tr>
<td>Linux®</td>
<td>Linus Torvalds</td>
<td></td>
</tr>
<tr>
<td>Oracle ® 9i.</td>
<td>Oracle Corporation, <a href="http://www.oracle.com">www.oracle.com</a></td>
<td></td>
</tr>
<tr>
<td>Tomcat™4.0</td>
<td>Apache Digital Corporation, Durango, CO, <a href="http://www.apache.com">www.apache.com</a></td>
<td></td>
</tr>
<tr>
<td>UNIX®</td>
<td>The Open Group, <a href="http://www.opengroup.org">www.opengroup.org</a></td>
<td></td>
</tr>
<tr>
<td>Windows® XP™</td>
<td>Microsoft Corporation, <a href="http://www.microsoft.com">www.microsoft.com</a></td>
<td></td>
</tr>
</tbody>
</table>
1. INTRODUCTION

The Problem and its Purposes

Online airline ticket booking system is one of the essential applications of E-commerce. With the development of Internet and security technology, more and more people begin to consume online, which is more convenient and personal than traditional way. The goal of this system is to make people purchase airline tickets easily. The system is written in Java.

Definition of Terms

What is CORBA? The common object request broker architecture (CORBA) allows distributed applications to interoperate (application to application communication), regardless of what language they are written in or where these applications reside. The CORBA specification was adopted by the object management group to address the complexity and high cost of developing distributed object applications. CORBA uses an object-oriented approach for creating software components that can be reused and shared between applications. Each object encapsulates the details of its inner workings and presents a well-defined interface, which reduces application complexity. The cost of developing applications is reduced, because once an object is implemented and tested, it
can be used over and over again. The object request broker (ORB) connects a client application with the objects it wants to use. The client program does not need to know whether the object implementation it is in communication with resides on the same computer or is located on a remote computer somewhere on the network. The client program only needs to know the object's name and understand how to use the object's interface. The ORB takes care of the details of locating the object, routing the request, and returning the result.

Remote method invocation (RMI) enables you to create distributed Java-to-Java applications, in which the methods of remote Java objects can be invoked from other Java virtual machines, possibly on different hosts. A Java program can make a call on a remote object once it obtains a reference to the remote object, either by looking up the remote object in the bootstrap naming service provided by RMI or by receiving the reference as an argument or a return value. A client can call a remote object in a server, and that server can also be a client of other remote objects. RMI uses object serialization to marshal and unmarshal parameters and does not truncate types, supporting true object-oriented polymorphism.

Web Services is about interoperability, which uses Simple Object Access Protocol (SOAP) an Extensible Markup Language (XML) based protocol for exchanging information.
2. THE SYSTEM AND DATABASE DESIGN

Credit Card Server (Web Services)

WebService,CORBA

Web-Server    SQL Server       Management Server(Web Services)

Internet

User         User            Company(Client)   Company(Client)

Figure 1: The structure of the web site
System Design

The Web-based airline ticket booking system uses client/server frame. The customers can use Web browser to access the system and book airline tickets. Because Internet and web browsers are widely used all over the world, there is no need to train customers how to use them. The second advantage is that there is no limit on customers’ operating systems, for example, they can use almost all kinds of popular operating systems such as UNIX, Linux and Window XP. In the server part, we use Java Server Pages (JSP), which is written in Java. So the server programs can also be run on most kinds of platforms, such as UNIX, Linux, or Windows. This increases the flexibility of the system. And the companies who use the system do not need to buy new hardware devices. For the communication between the server part and the third part services such as credit card verification, we use Web Services and common object request broker architecture (CORBA). Those technologies are widely used for distributed systems. And the cost of developing applications is reduced, because the source code can be reused by other applications.

After the user send a request by JSP to the Web server, the Web server search the database according to the request and then send back the result to user. After the user input his credit card information and buy a ticket. The server will connect (by CORBA or Web Services) to the remote Airline company to confirm the ticket, and get a confirm
number from the company. Then the server will connect (by CORBA or Web Services) to a remote credit card server, such as VISA or Master, to identify the user and confirm the payment. If there is a firewall between them they can use Web Services instead of CORBA. The airline companies provide all the airline information. They can input the information directly from a remote Client through their local server by remote method invocation. Different company uses different port to make sure its security.

Database Design

The database of this system is Oracle. The information in tables AIRLINE, FLIGHTCOMPANY, EMPLOYEE and FLIGHT is input by system operators. The information in Customer table is filled in by the system automatically when customers register into the system. And the Itinerary table is filled in by the system automatically also, after customers book tickets. Before they book tickets, the information about their selection, such as flight number and price, is stored in the session not in database.

The ITINERARY table will store all the information of every ticket. The FLIGHTCOMPANY and EMPLOYEE tables store the information of every company and its employees, then the system management part uses this information to identify the operators when they login in. The CUSTOMER table is filled out by the system when users register. The operators of the airline companies fill out the AIRLINE and FLIGHT tables.
Figure 2. The tables of database
3. THE LOGIC OF THE WEB SITE

Through the Web server, users can book airline tickets and pay bills online. To use this system, the user should register first through the registration page by inputting the user ID, password, user name, address and so on. Then the user can login to the system and trace their purchase by the user ID and password. The user information will be stored in the entire session until he or her logs out. After the user logs in the system, he can search for flights by the search bar or the main search page. The search is based on the destination, departure airport, date and Airline companies. When the user presses the search button at the bottom, the system will search for him from the database and return the result to the user sorted by price. The result includes the price, airline company, airline number and so on. To select the flight, the user just needs to press the “select” button corresponding to that flight result. If it is a round trip, the user will be asked to choose the return flight continually. Finally, the system will show the user what he has chosen. Then the user can click “Buy it” or just search again. If the user wants to buy it, he will go to the next step to pay for it. In the payment page, he is required to input his credit card information by the instruction of the page. Then the system will check the credit card information with the credit card company by using common object request broker architecture (CORBA) or Web Services. After the information is proved, the user will be
asked to confirm their purchase. Then the system connects the airline company to confirm the purchase and returns the user a Confirm number. Then the system will send the purchase information to the Credit Card company to finish the transaction. All the server parts of the credit card companies and the airline companies are simulated by this system. They just waiting for the invocation from the client and serve them by just returning a simple result without dealing with the data really. They are also written in Java. After the purchase is finished, the user can view the Shopping Cart to see what he has bought. And the system will send a confirm email to the user according to the information they input when they registered. There is a Personal Assistant Bar at the left side of all pages, through which the user can change their personal information, such as, password and address.

The above is the main purchase logic. Next I will talk about the system management part at the airline company. It is written in Java using remote method invocation (RMI). The server registers to a RMI registration and waits for clients to connect to it. The user part can be run either on the airline company or the website company. After they login to the system, the system will check the user name and password in the database. Then the verified user can use the system to add/delete/modify/search airline information, user information and the ticket information.

Now let me introduce the data stream in this system. For the booking system, all
the user information, such as the user name and the depart airport, are stored in the whole session until the user logs out or the session is expired. After the user buy the ticket, the ticket information will be stored in the local Oracle database and sent to the airline company also to book seats.
Figure 3. Main pages of Airline Ticket Booking System.
Figure 4. Structure of Client part of Management of Airline Ticket Booking System.
4. THE PLATFORM OF THIS SYSTEM AND CONFIGURATION

WEB-Server : BEA Weblogic™ Platform 8.1, Tomcat™ 4.0

Operating System: Windows XP

Programming languages: Java, JavaScript, HTML.

Database : Oracle 9i.

JDBC: oracle.jdbc.driver.OracleDriver.

Development tools: Jbuilder™ 9 Enterprise Trial
5. INTERFACE OF THE SYSTEM

System Management Part

After starting the Remote Method Invocation (RMI) registry, we can start the server. And the server will listen on the port to accept the connection from clients.

![Figure 5. The Server panel of RMI](image1)

![Figure 6. Login form of the Client part of RMI](image2)
Users can search Employees information according to name, position, ID or Department.

Figure 7. The Client part of RMI for Employee Information

Figure 8. The Client part of RMI for Flight Information
Figure 9. The Client part of RMI for Customer Information

Ticket Booking System on Web End

Figure 10. Login form of the Java Server Pages (JSP) Part
Users can modify their own information by the Personal Assistant
Figure 13. Search form of JSP Part

After the user login the system, they can access this form to search flights or use the Search Bar on the left of the page. After they buy a ticket, they can see the message “There is a new ticket in your shopping cart”.

After the user fills out the form and hit “Search” button, he will go to the result page (figure 12). If the user chooses to buy round trip tickets, he will be asked to choose the return flights also. The price will be added automatically. And the information of the flights is stored in this session, so the next page can use them.
Figure 14. the search result of JSP Part

Figure 15. Search summary of the JSP Part
If the user will be asked to input the adults and children names, which will appear in the tickets, according to the adults and children number the user choose when searching.
The user can see all the itineraries that he has bought.
Figure 19. Detail of itinerary of the JSP part

Figure 20. Register form of the JSP part

The user can register to the system through this page.
6. PROGRAMS

Common Object Request Broker Architecture (CORBA)

IDL(Interface Define Language):
module RemoteServer {
    interface CreditCard {
        string Transaction(in string cardnum, in string expiredate, in string name, in string extend);
    };
    interface AirlineCompany {
        string confirmTicket(in string itinerationNum, in string CompanyName, in string AirlineNum, in string date, in string num, in string extend);
    };
};

Remote Method Invocation (RMI)

The policy file is:
grant {
    // allows anyone to listen on un-privileged ports
    permission java.net.SocketPermission "*:1024-65535", "listen,accept,connect";
};

//Client part, show the GUI to the user and get dataset from remote server.

import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
import com.borland.dbswing.*;
import com.borland.dbswing.*;
import com.borland.dx.sql.dataset. *;
import com.borland.dx.dataset.*;

public class ClientFrame1 extends JFrame {
    String serverobj = null;
    String SqlText = null;
    JPanel contentPane;
    BorderLayout borderLayout1 = new BorderLayout();
    JPanel jPanel1 = new JPanel();
    com.borland.dx.dataset.TableDataSet tableDataSet1 =
        new com.borland.dx.dataset.TableDataSet();
    ClientProvider clientProvider1;
    ClientResolver clientResolver1;

    JdbNavToolBar jdbNavToolBar1 = new JdbNavToolBar();
    TableScrollPane tableScrollPane1 = new TableScrollPane();
    JdbStatusLabel jdbStatusLabel1 = new JdbStatusLabel();
    JdbTable jdbTable1 = new JdbTable();
    ResourceBundle res = Res.getBundle("com.borland.samples.dx.datasetdata.Res");

    //Construct the frame
    public ClientFrame1(String obje, String Sqltext) {
        serverobj = obje;
        SqlText = Sqltext;
        System.out.println(obje);
        clientProvider1 = new ClientProvider(serverobj, SqlText);
        clientResolver1 = new ClientResolver(serverobj);
        enableEvents(AWTEvent.WINDOW_EVENT_MASK);
        try {
            jbInit();
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }

    //Component initialization
    private void jbInit() throws Exception {
        contentPane = (JPanel) this.getContentPane();
        this.setSize(new Dimension(500, 300));
        this.setTitle(res.getString("RS_ClientTitle"));
    }
}
jPanel1.setLayout(borderLayout1);
contentPane.add(jPanel1, BorderLayout.CENTER);
contentPane.add(jdbStatusLabel1, BorderLayout.SOUTH);
jPanel1.add(jdbNavToolBar1, BorderLayout.NORTH);
jdbTable1 dataSet(tableDataSet1);
jPanel1.add(tableScrollPane1, BorderLayout.CENTER);
tableScrollPane1.getViewport().add(jdbTable1);
tableDataSet1.setProvider(clientProvider1);
tableDataSet1.setResolver(clientResolver1);
}

//Overridden so we can exit when window is closed
protected void processWindowEvent(WindowEvent e) {
    super.processWindowEvent(e);
    if (e.getID() == WindowEvent.WINDOW_CLOSING) {
        //System.exit(0);
        this.dispose();
    }
}

Database

There are many functions. This sample program gets flight information from database based on the flight number.

public ResultSet getFlightinfo(String flightnum){
    Connection conn = null;
    try {
        //conn = ConnectionFactory.getConnection();
        Class.forName("oracle.jdbc.driver.OracleDriver");
        String url="jdbc:oracle:thin:@localhost:1521:oracle9i";
        conn=DriverManager.getConnection(url,"thesis","thesis");
        Statement stmt= conn.createStatement(
            java.sql.ResultSet.TYPE_SCROLL_INSENSITIVE,
            java.sql.ResultSet.CONCUR_READ_ONLY);
        String query1 = "SELECT * FROM Airline WHERE AirlineNumber ="
+Integer.valueOf(flightnum) ;
ResultSet rs1 = stmt.executeQuery(query1);
if (rs1==null){
    rs1.close();
    return null;
} else{
    return rs1;
}
}
}
catch (Exception e) {
    System.out.println("Exception found::"+e.getMessage());
    e.printStackTrace();
}
System.out.println("DBmain::getFlightinfo: end");
return null;
}

Java Server Pages (JSP)

This is the example of a JSP page, which is used to process data uploaded by user and invoke a function to get information from database and transfer to the next page.

<%@ page session="true" %>
<%@ page contentType="text/html;charset=gb2312" %>
<%@ page import="java.util.*" %>
<jsp:useBean id="DBMainID" scope="session" class="jsps.DBMain" />
<%  session = request.getSession(true); %>
<%
String username=null;

username = (String)session.getAttribute("username");

if(username==null){
    
    response.sendRedirect("login.jsp");

    System.out.println("session userid is null!");

}

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

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

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

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

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

if(DBMainID.changeUserinfo(username,firstname,lastname,phone,address,email)!=1)

){

    System.out.println("change traveler information failed");

    response.sendRedirect("travelerinfo.jsp");

    return;

}

String url = response.encodeURL("personalassis.jsp");

response.sendRedirect(url);
REFERENCES


