Chinese University of Hong Kong

Department of Computer Science and Engineering

Final Year Project

LYU9901 : TravelNet

Final Report 1999-2000

Prepared by: Ho Chi Ho Malcolm

Supervised by: Michael R. Lyu

Group Member: Ho Chi Ho Malcolm
97618593
Lau Chi Ho Arthur
97590853

Submission Date: 18th April 2000

Table of Content

Abstraction

Internet is growing in a fast manner. It covers a wide range of human activities nowadays. This growth leads to great opportun ities for doing business online. For this reason, We built this Online traveling agency to practice real life E-commerce.
In this report, We will summarize the work done in this semester and state the improvement made on the project since the last report. First of all, there will be an overview of the whole project, and then the facilities provided by TravelNet such as online shopping and flight reservation will be described. After that, actual implementation and system architecture will be introduced. CORB A integration with TravelNet for containing distributed components is also an important new part to be mentioned. TravelNet is cooperated with a secure payment system for credit cards and a micro payment system for store -value cards. Their communication me chanism will be described and a performance measurement on payment will also be conducted also. Finally, a conclusion will be given.

Chapter 1. Introduction

Objective and Overview of TravelNet

There are many companies that expand their business onto the I nternet for selling product, promoting their products and providing services. This kind of business not only opens a new way for business but also provide great convenience for customers. The main objective of this project is to develop such an E -commerce system that is similar to real life and can fulfill the actual needs of the society.

TravelNet provides most of the common type of services, which is provided in reality, they are providing information (Travel Guides, Hotel Information), services (Flight Reservation) and products (Online Shop). As what customers need is an interactive service-providing site, Web server should be capable of handling dynamic content with respect to client requests, so server side programming, back-end information retrieval and dynamic Web pages generation is necessary. In TravelNet, We make use of Java Programming Language with Java Servlet and Java Server Page to fulfill these needs.

Besides making a dynamic system, distributing some of the components such as flight database can improve the performance and fault tolerance. That's why we enhanced some parts by distributing them with CORBA.

Payment is an essential part for any business. Although this should not be a part that developed by merchants, merchants, like TravelNet , must be included payment services. Different from last semester, we are not just simulating a simple bank but using more complex payment servers. Payment by credit cards and MONDEX card (micro payment) can be handle by TravelNet.

Chapter 2. TravelNet Facilities

2.1. Introduction

TravelNet, similar to other traveling agencies, provides a number of convenient functions to users so they can enjoy their travel without spending lots of effort. Services included flight search and reservation, online accessory shop, hotel information and travel guides. Before using these services, a user must register to be our member, so register and login services for membership is also provided.

Below is the main Page of TravelNet, users can access the services through a clic k on the links.

Figure 2.1. TravelNet Main Page

2.2 TravelNet Membership

Registration

In order to use our services, user has to apply for an account. It makes a more convenient use of TravelNet after registration. Personal information will be stored by TravelNet and be protected by pa ssword for illegal use. TravelNet can retrieve information automatically for the users to fill in the necessary form fields after logged in to our system. Those forms field may be name and address for product delivery.

Registration is easy and the process will take a very short period of time. User Just will in some necessary fields then submit the information. Once a user registered, his/her account was activated.

Figure 2.21. Registration page

Login

Afterwards, they can use the username and password to login to our sy stem and enjoy the facilities provided by TravelNet.
Login page will be appeared on the first access of the services that need a user Identification.

Figure 2.22. Login Console

Update Profile

Information can be updated after a registered user had logged in to our system. Beside s the username, all of the fields can be changed since the username must be unique among all users of TravelNet.
For security reason, if a user hope to change his/her password, he/she has to type in the password again even he/she had already logged in. Of course, the new password has to be inputted twice to ensure he/she had typed in the desired password.

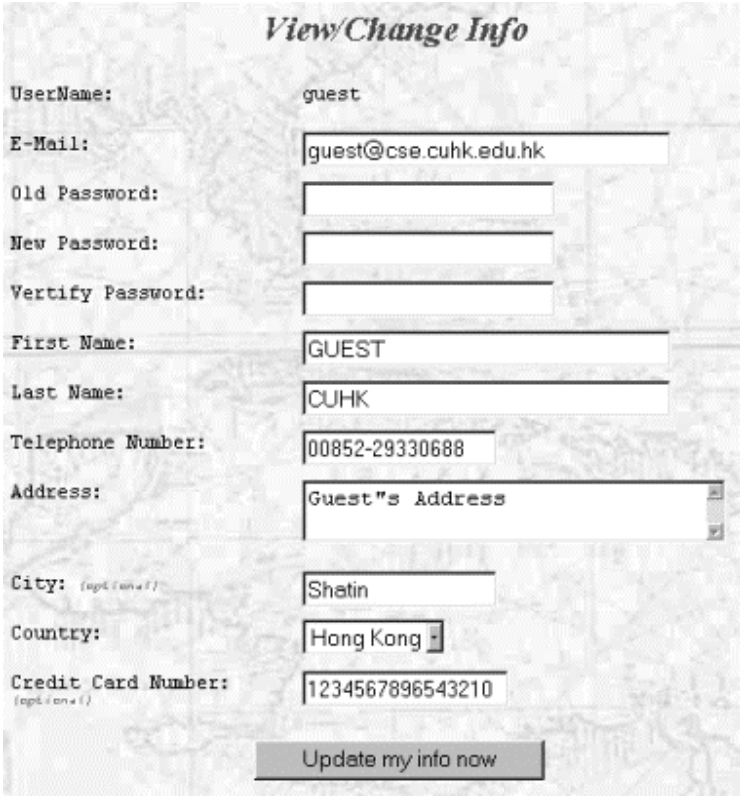


Figure 2.23. Update personal information

2.3 Flight Search and Reservation

Flight search is one of the most significant features of TravelNet. It had been enhanced a lot since the last semester. Now not only the single flight search, but also round trip flight search. In addition, itinerary management and flight ticket reservation system is now available.

Flight Search

In single flight search , users are allowed to consult the airlines ' databa ses with users requirement and make reservation on the search result. The system requires users to input some basic elements on the search. The basic elements of queries includes the departure and arrival cities, the departure date, the types of flight (one way/round trip), the class of service (first class/ business class/ economy class), the age category of the ticket (below 12/adult/above 65). Possible additional requirement includes the exact range for departure time, the choice on fare (e.g. is there any penalties for refund of tickets), the airline company, etc. Usually, the optional requirement helps to lower the size of the search result while the basic method is also provided to enhance the flexibility of the search. The search options are flight company and change penalty. In round trip flight search, the available options are similar except one more return date and time. Round trip is always cheaper than 2 single flights so this flight search is useful for those people who had already planned their trip specifically.

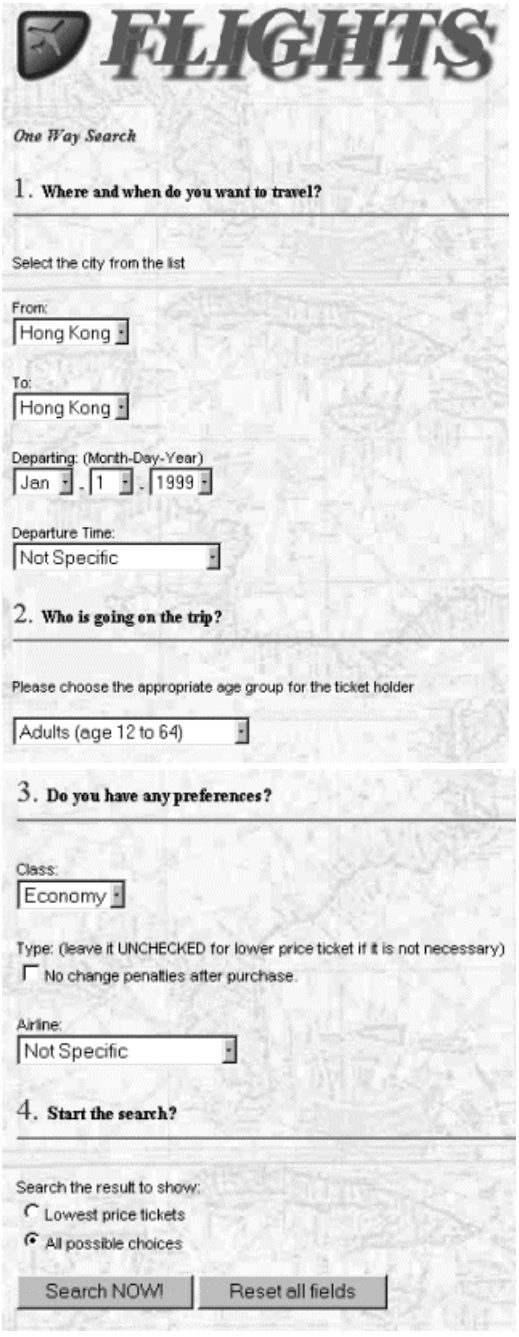


Figure 2.31. Flight Search

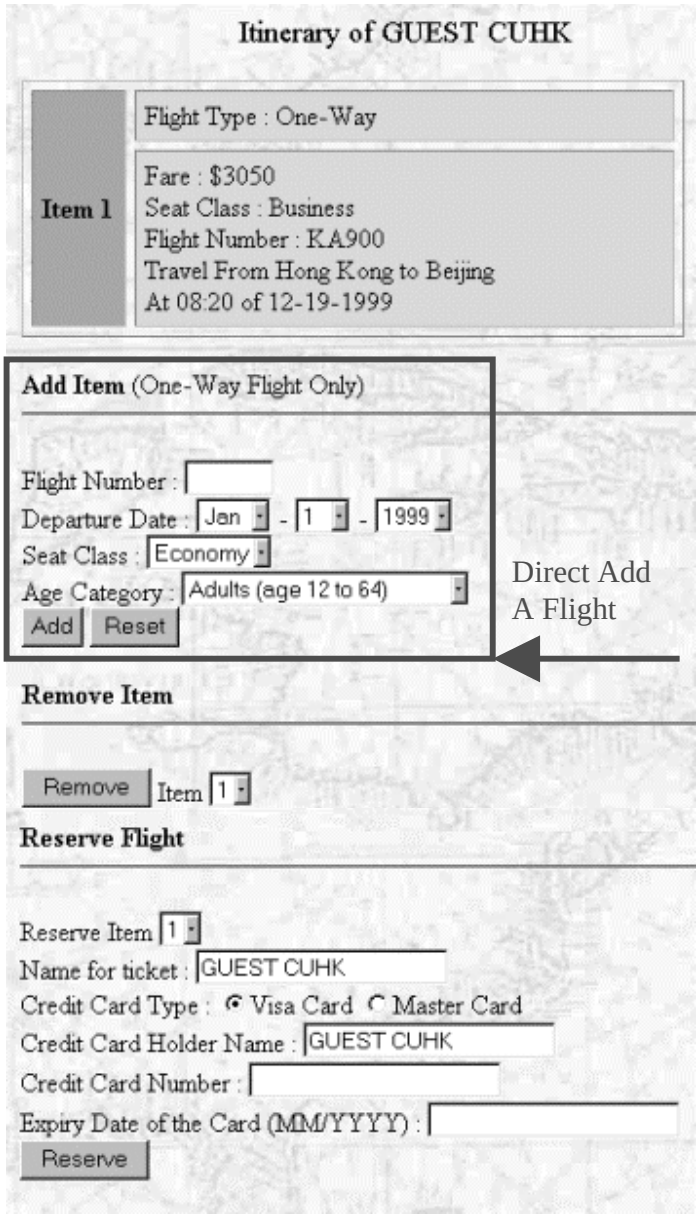
All the selectable fields are combo boxes. These boxes reduce the typing error and provide easier selection. Even those people who are not familiar with flight information can still get what he wants from the selection in these b oxes. After results have been displayed, the user can select any of them and add to there own itinerary.

A sample result as shown on the right. A user can select one of the flight which is more suitable to him/her and add it to itinerary buy click the button on the bottom left corner.

Figure 2.32. Search result

Itinerary management

Each of the users has his/her own itinerary. It stores the flight he interested and related to their trip. They can add a flight from search result and can also remove it from itinerary afterwards. If they had confirmed the current itinerary suits their needs, they can reserve it for their trip. Itinerary of a user will be stored in database, so even a user log out from our system, they can retrieve the itinerary information after another login session.



In the itinerary management, if a user know the exact information of a flight (like flight number, departure time and date), they can query it directly and add it to itinerary immediately. This query is more direct and take less time than general search.

Flight reservation can only be processed by credit card. As the amounts of flight tickets are expensive, it is not possible for then to use micro-payment such as MONDEX to do the payment. User type in the information, then TravelNet will help to connect to a secure payment server to finish this payment transaction.

Figure 2.33. Itinerary management

2.4. Travel Accessories Shop

This shop is another major component of TravelNet. Shopping procedure consist of three main steps: Selecting products, putting then into user's shop baske t and pay for it. There are three main types of category for users to choose. They are luggage, guides and maps, and miscellaneous stuffs. They can view and update their basket anytime during a shopping session, but if a user logged off from our system wit hout checkout the basket item, the items in the basket will not be stored and be cleared.

Here is a snapshot of Travel Shop main page. User can access travel Shop after he/she had logged in to our system. By clicking one of the categories they can s ee more detail on what is currently selling.

Figure 2.41. Travel Shop main page

Shop Basket

After user has find a product he/she is interested in, he/she may add it to the basket by clicking the “Add to Basket” button. Users may also change the color they desired and the quantity they want.

Figure 2.42. Adding an item to basket

In the basket interface, you can drop any item from your basket by selecting the item to drop then click the Update Basket Button. Afterwards, it will be refreshed and updated.

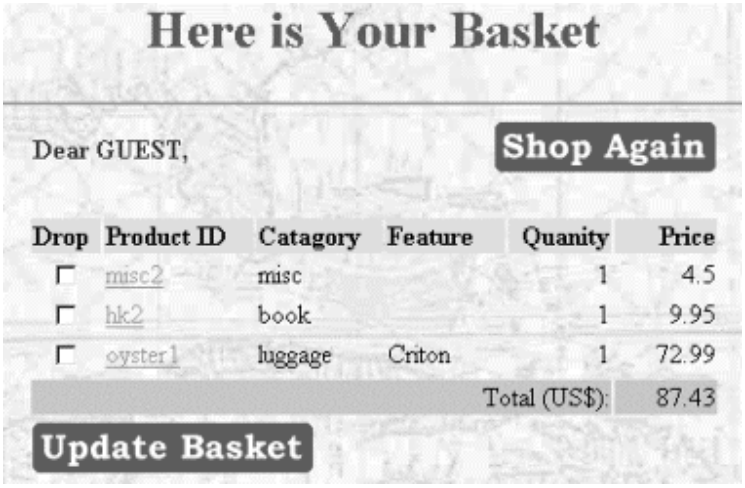


Figure 2.43. Displaying Shop Basket

Shop Payment

Different from flight reservation, users can pay for their selected stuffs in the shop basket not only by credit card but also MONDEX card. As the items sold in Travel Shop is not too expensive, we can make use of Internet micro-payment method to provide a more secure and convenient way for our customers to pay their bills.

For MONDEX payment, user should ready their card into the reader when they hit the link. After that the client will connect to MONDEX payment server and start the payment. For credit card payment, they have to type in the correct information before they checkout. Information like type of card, name of card holder, card number and expire date is necessary .
For both payment methods, upon a successful payment, an acknowledgement will be given, then TravelNet will carry on to the post payment processes.

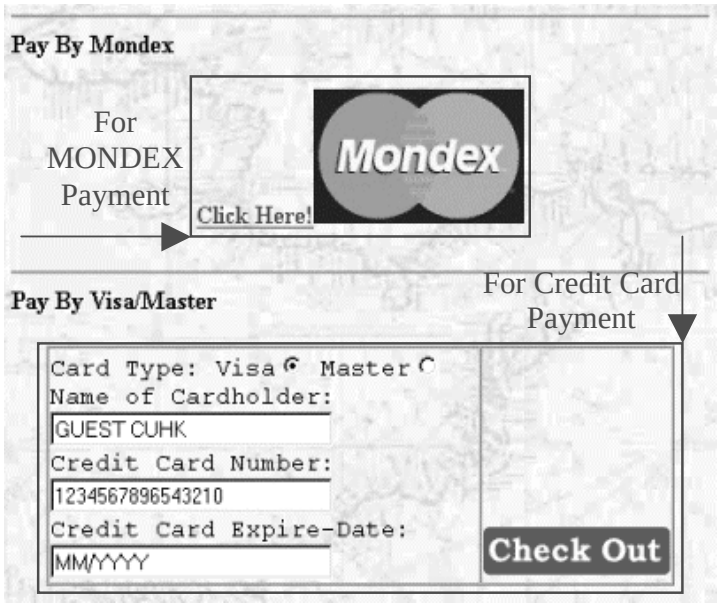


Figure 2.44. Payment Screen

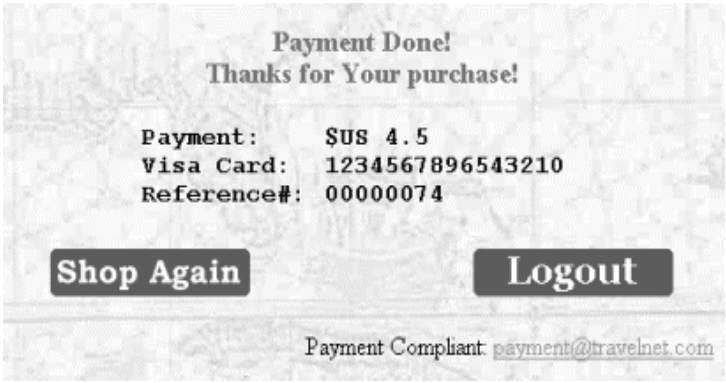


Figure 2.45. Payment Done

2.5. Hotel Information

Hotel is an essential part for any travel, so TravelNet include some useful information for the hotels of the Asian cities that TravelNet targeted on. The cities included Beijing, Hong Kong, Tokyo, Seoul, Shangh ai, Taipei and Singapore. User can easily browse the information of the hotels by some click on the hotel information main page. Hotel reservation is not available since our project time is limited and the mechanism is more or less similar to that of fligh t reservation, so we miss that out.

Figure 2.51. Hotel Information Main Page

The snapshot on the right shows a sample hotel information page of TravelNet. Information includes description, location, room, rates and facilities of a hotel. Also some images of the hotel will be posted in the page. User can still make a reservation request to the hotel, but TravelNet will just direct the request to the specified hotel.

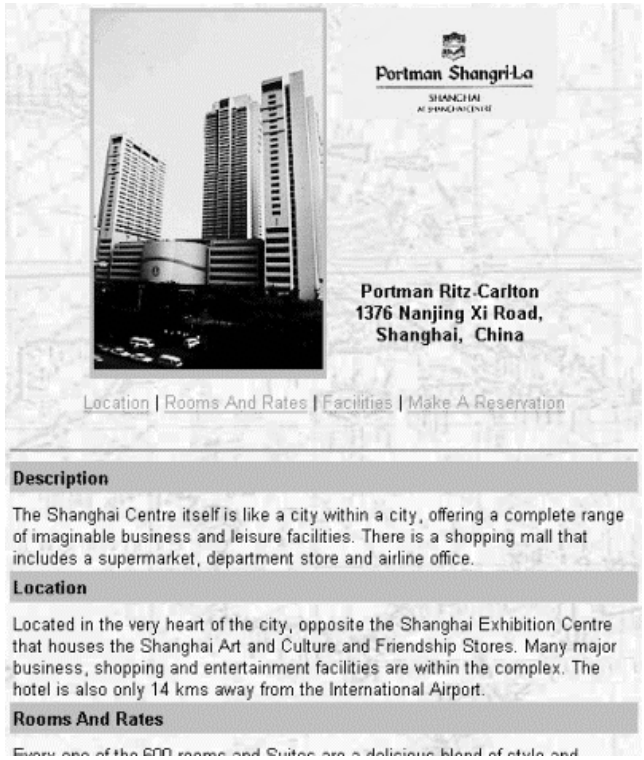


Figure 2.52. Information Page of a Hotel

2.6. Travel Guides

It will be very confusing if a traveler doesn't know much about the information of the places. TravelNet tra vel guides will provide users a complete overview of the targeted countries i.e. China, Japan, Korea, Singapore, Taiwan. Information like basic description of the cities, map of the cities, introduction of some f amous spot, transportation and the currency.

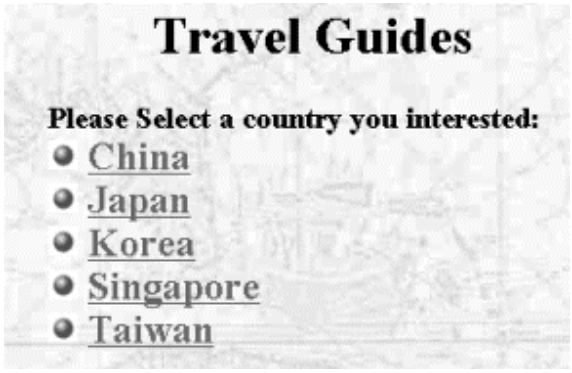


Figure 2.61. Travel Guides Main Page

2.7. Online Payment

As mentioned before online payment is a essential part of TravelNet. Flight reservation and shop checkout needs payment. We had already incorporated with two types of Internet payment, one is the traditional credit card payment and th e other is a new micro payment method by MONDEX card. We will describe the mechanism of these payment methods. Also the communication between these payment servers and TravelNet will be stated in chapter 5.

Chapter 3. System Architecture

3.1. Introduction

In this chapter, the overall architecture of TravelNet will be shown. Component connections will be described abstractly in order to give a conceptual idea of how TravelNet work. Detail design will be provided in the next chapter. Besides, the old model will also be illustrated as a comparison of major changes.

3.2. System Architecture Overview

3.2.1 Previous Centralized System

Below describes the old system architecture Of TravelNet. Most of the processes are direct database access, data manipulation and generating response. The system includes security concern on Internet connection (SSL) even though it lacks nearly real life payment system.

3.2.1 Previous Centralized System (cont')

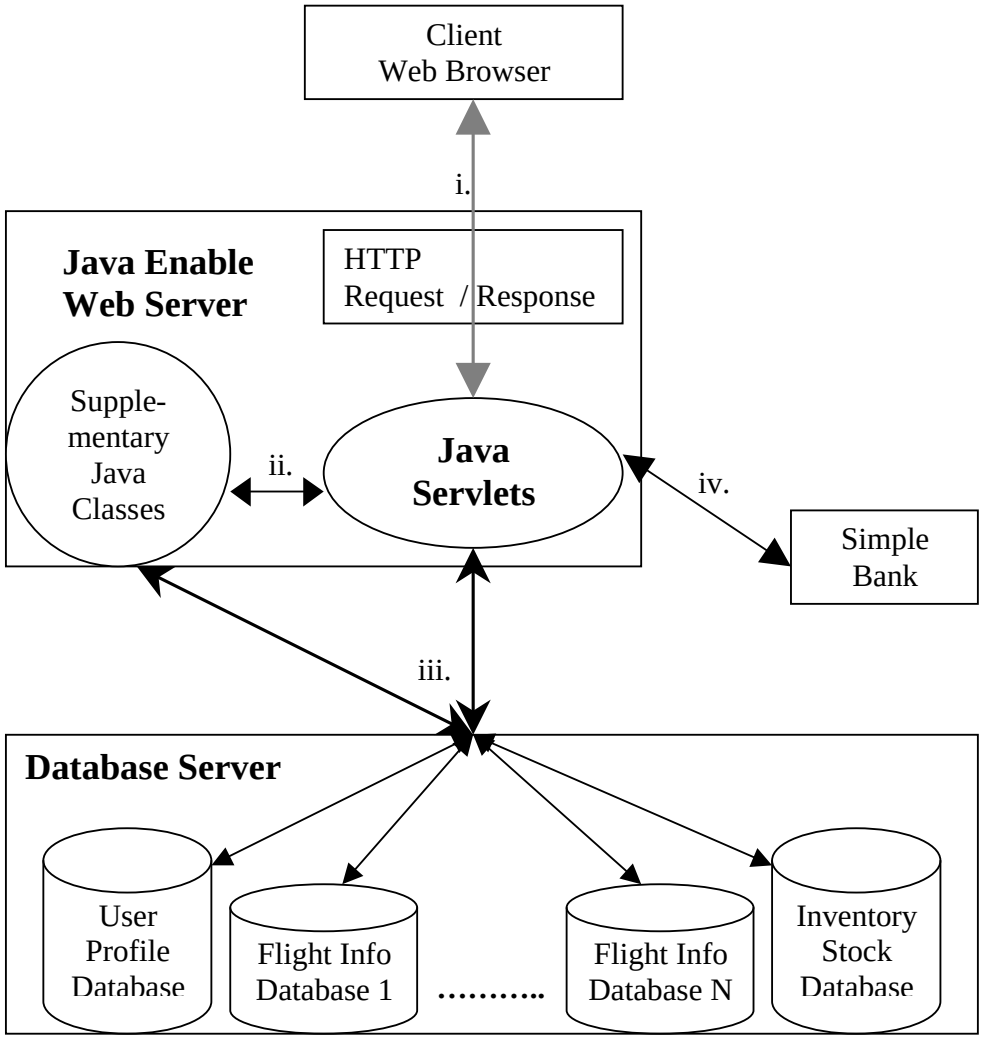


Figure 3.21. System architecture of previous system

Components description:

- *Client Web browser*: Client of TravelNet with an HTTP Web browser that support SSL connection.
- *Java enable Web server*: TravelNet Web server which is capable of running Java programs, Java Servlets and Java Server Pages for handling client requests
- *Java Servlets*: In TravelNet most of the application logic will be programmed inside it such as database connection and generating dynamic Web pages.
- *Simple Bank*: A very simple prototype of a bank to simulate credit card transaction.

- *Database Server*: It consists of a collection of database tables. Tables included TravelNet user profile, flight database of a number of airlines, inventory stock database for Travel Shop.
  - User Profile database- It stores all TravelNet user's profile (username, password, name, address, etc.)
  - Flight Information database- each database represents an airline company database (so it should not be centralized). It stores the information like price, flight detail, classes and plane information.
  - Inventory Stock database- The data of the product sold in Travel Shop will be stored in it.

Dataflow descriptions:

i. Client will use a web browser to access the web services provided by TravelNet Web server. Data/Requests send from client to server will be encrypted on a Secure Socket Layer (SSL). Web server will generate response page according to client request and send back to client browser.
ii. Although Java Servlets can carry out all the application logic by itself, for reusability of components, we can build some classes for some common purposes.
iii. Database access is essential for client requests, such as user management, flight searches and stock management. Java programs can access most of the databases system by JDBC. For this Java Servlets and Java classes can query the database for client requests and provide then necessary information.
iv. At the end of 1st semester, we can not obtain any secure payment servers to incorporate with TravelNet, so we simulate a very simple one to handle credit card payment. When I user checkout from Travel Shop, it will invoke a checkout Servlet and update the bank database for the credit value of the credit card for checkout.

3.2.2 Current system with distributed components

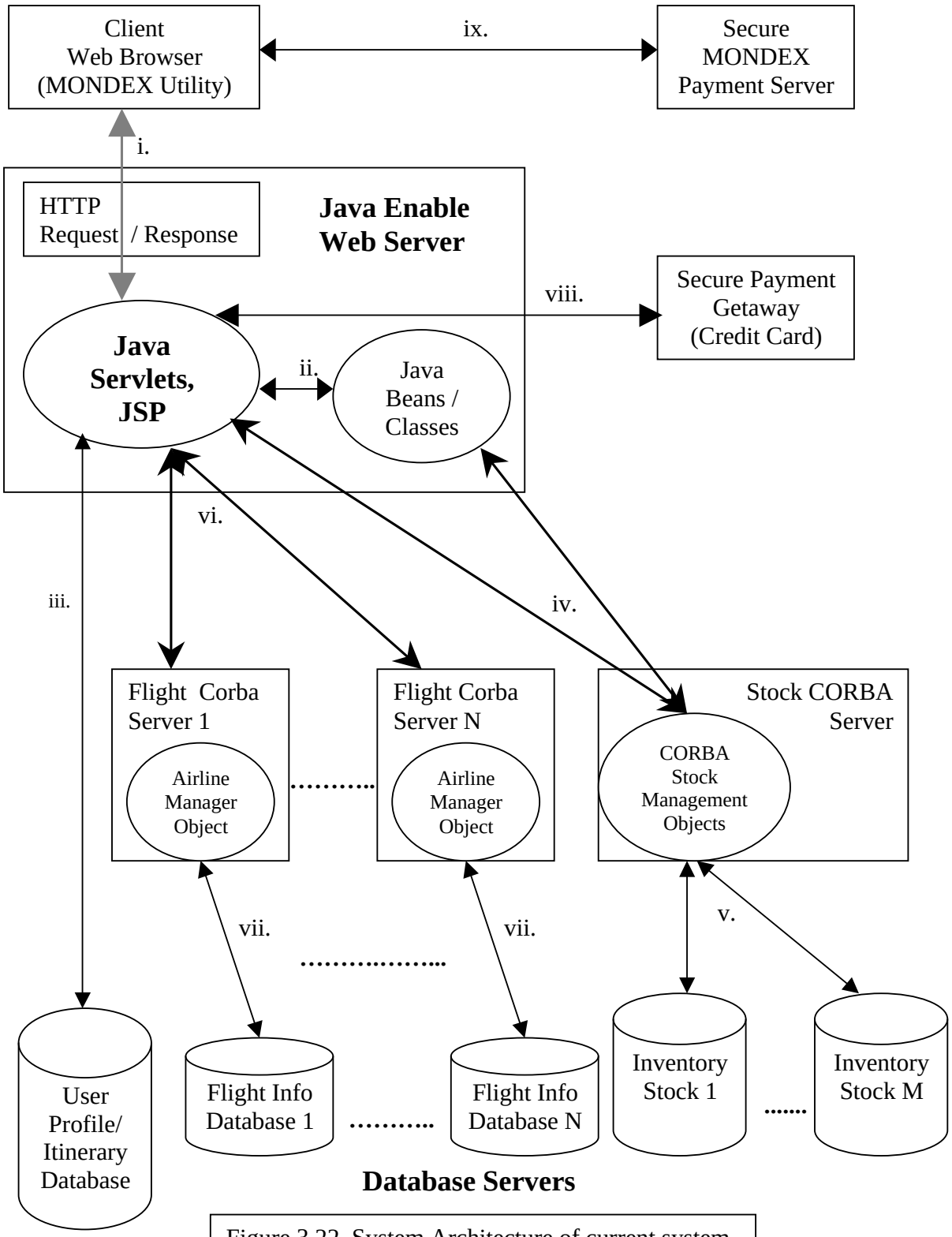


Figure 3.22. System Architecture of current system

3.2.2 Current system with distributed components (cont')

The current system is much more complex than before. With secure payment systems and distributed components added to the previous system, it becomes more realistic and modular. Now we are going to show the detailed flow of data of the system.

Components description:

- *Client Web browser*- Client of TravelNet with an HTTP Web browser that support SSL connection. For the use of MONDEX, client must be equipped with MONDEX hardware and software.
- *Java enable Web server*- TravelNet Web server which is capable of running Java programs, Java Servlets and Java Server Pages (JSP) for handling client requests
- *Java Servlets*: In TravelNet most of the application logic will be programmed inside it such as database connection and generating dynamic Web pages.
- *Java beans and classes*- These Java modules provide reusable functions to fulfill the requests. Java Server pages make use of beans for processing, so that the application logic can be separated from web design
- *User Profile database*- It stores all TravelNet user's profile (username, password, name, address, etc.). Besides, itinerary of user will be stored in profile database also.
- *Flight CORBA servers*- Each server will manage a single (or a number of) airline manager object. Query of any airline information must through the airline managers. These similar objects are distributed by a number of CORBA servers.
- *Flight Information database*- Each database represents an airline company database (so it should not be centralized). It stores the information like price, flight detail, classes and plane information.
- *Stock CORBA server*- A CORBA server than manage the stock CORBA objects. Each object will manage a stock database, for updating and accessing information. Any request to the database must make use of the manager object of the stock server.

- *Inventory Stock database*- Each database stores a category of product, like luggage. This stock database is different from the previous design, since the database is distributed by categories.
- *Secure Payment Gateway*- It is a secure payment system for credit card which is developed by a post-grad student. Credit card payment request will be handled by this system.
- *MONDEX Payment Server*- MONDEX is a common smart card system for international use. The server is maintained by the Center of Innovation and Technology. The server allows MONDEX card for a payment over the Internet.

Dataflow description

i. Client will use a web browser to access the web services provided by TravelNet Web server. Data/Requests send from client to server will be encrypted on a Secure Socket Layer (SSL). Web server will generate response page according to client request and send back to client browser.
ii. Java Servlets will probably call some classes for processing some functions and Java Server Pages can make use of Java beans to handle some application logic.
iii. All process related to user profile access will consult the TravelNet users' database. These access include login, update profile, etc.
iv. When a server application need the stock data from databases, it will request the stock manager in the stock CORBA server using the interface provided by this object.
v. Stock manager manages all the stock managing module of an inventory database of a specific category. According to the requests from web server, Stock manager will consult stock modules to retrieve data from database.
vi. When a flight related query/request is mad e from client, Web server will request the airline managers that reside on different machines for the desired information. Actual database is abstracted from the interface provided.
vii. An airline manager can connect to their responsible database, data will be retrieved from the database according to the request from the interface.
viii. If a checkout or reservation process with a credit card payment is going to carry out, a message of encrypted (by payment server public key) request will be sent out. A

payment result acknowledgment message will be encrypted (by TravelNet's public key) and send back to merchant. Servlet will then process the result and generate output.

ix. This connection is between Mondex client machine and Mondex payment Server. Payment plugin collect enough information of a payment from TravelNet Web server, then it request the payment server directly for a payment process. After payment is done, the result will be return to the client. The Mondex user will pass the payment result back to TravelNet and let it to verify the payment result.

Chapter 4. System Design

4.1. Development tools

4.1.1 Java

Java is an object-oriented language similar to C++, but simplified to eliminate language features that cause common programming errors. Java source codes are compiled into a format called bytecode, which can then be executed by a Java interpreter. Compiled Java code can run on most computers because a Java interpreters and runtime environment, known as Java Virtual Machines which exists on most of the operating system.
It has a comprehensive set of classes for easier programming. Portability and structural object oriented approach make it suitable for large scale cross platform applications. The main reasons for selecting Java as a development tools are the easily integration with CORBA and the advantages for web applications.

4.1.2. Java Servlet and JSP

Java Servlet is a well-defined Java package that bring great convenience to web applications. It is a server-side component that is platform and protocol independent. Servlets can be used to extend the functionality of a Java-enabled Web server. Servlet can be imagined as a faceless applet. Servlets are loaded and invoked by the Web server in much the same way that applets are loaded and invoked by Web browsers.
The HTTP Servlets replaces the traditional CGI programming in a more convenient and efficient way. We can write pure Java language to handle web browser requests. As it is pure Java programming, all the advantages of Java will still be retained.

4.1.2 CORBA

CORBA allows applications to communicate with one another no matter where they are located or who has designed them. CORBA was introduced by Object Management Group (OMG) and defined the Interface Definition Language (IDL) and the Application Programming Interfaces (API) that enable client/server object interaction within a specific implementation of an Object Request Broker (ORB).

The (ORB) is the middle ware that establishes the client -server relationships between objects. Using an ORB, a client can transparently invoke a method on a server object, which can be on the sa me machine or across a network. The ORB intercepts the call and is responsible for finding an object that can implement the request, pass it the parameters, invoke its method, and return the results. The client does not have to be aware of where the object is located, its programming language, its operating system, or any other system aspects that are not part of an object's interface. In so doing, the ORB provides interoperability between applications on different machines in heterogeneous distributed environments and seamlessly interconnects multiple object systems.

The nature of CORBA meets our needs in distributing the system components. Airline companies agreed with a common CORBA interface, different airline companies can have different approaches on their own platform, database design and programming language used. Airline objects should reside on different machines so performance should be better than run all airline searches on a single machine.

4.1.3.1. URL Naming services

URL Naming Services is provided by Borland Visibroker 4.0. It is a simple mechanism that lets a server object associate its IOR with a URL in the form of a string in a file. Client programs can then locate the object using the URL pointing to the file containing the stringified URL on the web server. The URL Naming Service supports any URL scheme that Java runtime supports, such as http.

IOR stands Interoperable Object Reference. It represents a reference to a CORBA object in the form of a string. By obtaining this string, the UR L naming service can help a client to resolve the reference to the object. This service will be used for the CORBA object reference for TravelNet.

4.2. Database Design

4.2.1. User Profile Database

- **USER_PROFILE:**
  This database stores all necessary information of TravelNet users. Credit card number is not a compulsory field because it is not secure to store the credit card number in the database.

Name	Type	Nullity	Integrity
USERNAME	VARCHAR2(12)	NOT NULL	PRIMARY KEY
EMAIL	VARCHAR2(30)	NOT NULL	
PASSWORD	VARCHAR2(20)	NOT NULL	
FIRSTNAME	VARCHAR2(20)	NOT NULL	
LASTNAME	VARCHAR2(20)	NOT NULL	
TELENUM	VARCHAR2(15)	NOT NULL	
ADDRESS	VARCHAR2(90)	NOT NULL	
CITY	VARCHAR2(15)		
COUNTRY	VARCHAR2(5)		
CREDITNO	VARCHAR2(16)		

TRANSCATION_RECORD:

Payment transactions will be recorded in here. For later reference or complain from users. The second field will be stored as credit card number for credit cad payment and MONDEX payment ID for MONDEX payment.

Name	Type	Nullity	Integrity
TRANS_NO	NUMBER(38)	NOT NULL	PRIMARY KEY
CARD_NO/ MONDEX PID	VARCHAR2(16)	NOT NULL	
AMOUNT	FLOAT(126)	NOT NULL	>0
TRANS_TIME	DATE	NOT NULL	

4.2.2. Inventory Stock Database

- **LUGGAGE_STOCK:**
  Inventory stock of luggage will be stored in this database. It reveals the actual stock of Travel Shop.

Name	Type	Nullity	Integrity
PRODUCT_ID	VARCHAR2(10)	NOT NULL	PRIMARY KEY
PRICE	FLOAT(126)	NOT NULL	>0
STOCK	NUMBER(38)	NOT NULL	>0

- **BOOK_STOCK:**
  Inventory stock of books will be stored in this database. It reveals the actual stock of Travel Shop.

Name	Type	Nullity	Integrity
PRODUCT_ID	VARCHAR2(10)	NOT NULL	PRIMARY KEY
PRICE	FLOAT(126)	NOT NULL	>0
STOCK	NUMBER(38)	NOT NULL	>0

- **MISC_STOCK:**
  Inventory stock of miscellaneous products will be stored in this database. It reveals the actual stock of Travel Shop.

Name	Type	Nullity	Integrity
PRODUCT_ID	VARCHAR2(10)	NOT NULL	PRIMARY KEY
PRICE	FLOAT(126)	NOT NULL	>0
STOCK	NUMBER(38)	NOT NULL	>0

4.2.3. Airline Companies Databases

- **FLIGHT_INFO**
  A database stores all the flights operated by the Airline Company.

Name	Type	Nullity	Integrity
FLIGHT_NUM	VARCHAR2(6)	NOT NULL	PRIMARY KEY
SRC_PLACE	VARCHAR2(3)	NOT NULL	
DEST_PLACE	VARCHAR2(3)	NOT NULL	
D_TIME	TIME	NOT NULL	
A_TIME	TIME	NOT NULL	
AIRCRAFT	VARCHAR2(4)	NOT NULL	

- **FLIGHT_SCHEDULE**
  A database for weekly schedule of specific flights

Name	Type	Nullity	Integrity
FLIGHT_NUM	VARCHAR2(6)	NOT NULL	PRIMARY KEY
SUN	VARCHAR2(1)	NOT NULL	
MON	VARCHAR2(1)	NOT NULL	
TUE	VARCHAR2(1)	NOT NULL	
WED	VARCHAR2(1)	NOT NULL	
THU	VARCHAR2(1)	NOT NULL	
FRI	VARCHAR2(1)	NOT NULL	
SAT	VARCHAR2(1)	NOT NULL	

- **FARE_INFO**
  A database stores the fare list of each class of tickets in terms of one-way flights and round-trip flights.

Name	Type	Nullity	Integrity
FLIGHT_NUM	VARCHAR2(6)	NOT NULL	PRIMARY KEY
OW_FCLASS	FLOAT(10)	NOT NULL	>0
OW_BCLASS	FLOAT(10)	NOT NULL	>0
OW_ECLASS	FLOAT(10)	NOT NULL	>0
RT_FCLASS	FLOAT(10)	NOT NULL	>0
RT_BCLASS	FLOAT(10)	NOT NULL	>0
RT_ECLASS	FLOAT(10)	NOT NULL	>0

- **PLANE_SIZE**
  A database stores the capacity of each plane of 3 classes of service(first class/business class/economy class).

Name	Type	Nullity	Integrity
AIRCRAFT	VARCHAR2(4)	NOT NULL	PRIMARY KEY
FCLASS	NUMBER(3)	NOT NULL	
BCLASS	NUMBER(3)	NOT NULL	
ECLASS	NUMBER(3)	NOT NULL	

- **TICKET**
  A database stores the capacity of each plane of 3 classes of service (first class/business class/economy class).

Name	Type	Nullity	Integrity
FLIGHT_ID	VARCHAR2(6)	NOT NULL	PRIMARY KEY
DDATE	DATE	NOT NULL	PRIMARY KEY
FCLASS	NUMBER(3)	NOT NULL	
BCLASS	NUMBER(3)	NOT NULL	
ECLASS	NUMBER(3)	NOT NULL	

- **USER_ITINERARY**
  A database which stores the sold ticket for internal usage.

Name	Type	Nullity	Integrity
TICKET_NUM	VARCHAR2(12)	NOT NULL	PRIMARY KEY
FLIGHT_NUM	VARCHAR2(6)	NOT NULL	
NAME	VARCHAR2(40)	NOT NULL	

***Note:** The above is the database schema for each airline company. Since it is not available to have multiple database for us to use, we simply simulate the situation by append a code as a prefix to the database table to represent the ownership of the table. For example, the code for Cathay Pacific Airways is CX, so all the tables that belongs to the company are started with CX_, like CX_TICKET and so on.

4.3. Online shop Design

In this session, we are going to describe the working mechanism of Travel Shop. It uses Java Servlet, JSP and CORBA stock objects.

4.3.1. Product descriptions Pages

Product description page is necessary for customer to choose a product to buy. Product supply differ from time to time, so even the page can be statically stored, it will be better if can be generated dynamically.

Product page is generated dynamically by a description file with specific format. This file contain the product ID, product descriptions, image location and price of a collection of products that are going to be displayed in one page. By making use of JSP, the format of the description page is defined in HTML. Inside JSP, some Java scriptlet read in the product information from a description file and output them in the HTML format that had already specified in JSP.

Product item information may be removed, added or updated. Making use of a static HTML file may be difficult to achieve this. Maintain a description file is easier than managing an HTML file with output format and data inside.

It is possible for the data of the description file to be stored in a database system, but it will have a dependency on a DBMS. For this simple page generation, we choose to use a file instead.

4.3.2. Shopping basket mechanism

Every user of Travel Shop will have a virtual shop basket; it keeps track of the items a logged in user had selected. Java Servlets can save any Java objects into a session of the http client. An HTTP Session contains a collection of key-value pair. An object can be mapped to a key and put it as a session value. Using a session, the item selected can be stored.

When an item is going to add to the basket, it will first check for whether there is enough stock for the user to buy it. A Java bean is responsible for checking stock level. This stock bean will use the CORBA interface of the stock object, which is responsible for the inventory stock of the category of the item to be selected. After the request of stock bean had finished, it will be acknowledged. Upon the status of stock, the shop basket will be updated. For removing an item, it simply updates the session value.

If a user wants to checkout the items in the shop basket they have to choose the way for check out (Credit card & MONDEX card). For MONDEX payment, user will connect to Payment Server directly. For Credit Card payment, TravelNet will be act as a middleman between the Payment Server and user. For both payment methods, upon a successful Payment, post payment process will be carried out and an acknowledgement page will be generated.

Shop Basket will not be stored in database. It will only be stored for a login session.

4.3.3. Post Payment Process

After payment is done, the responding change of stock will be updated through the stock bean again (quantity of sold items is deduced from database). Some other things have to be recorded for reference, they are the delivery log and transaction record. Delivery log keeps the items to be deliver and transaction record is used for future reference of payment problems.

4.4. Stock Management

4.4.1. Mechanism

Stock is managed by a CORBA object resides on a machine other than the Web server machine. As mentioned before a Java bean is responsible for resolving reference of this object and invoke the method for access stock database indirectly. The IOR of this CORBA object is placed on another web server. Stock bean resolve the reference by obtain this IOR.

The interface of stock CORBA object provides 2 main methods. They are checking the price of an item and ordering an item. In the Following subsection, we will describe the interface in more detail.

4.4.2. Communication Interface

- **Stock Manager Interface**

```
interface StockMgr{
    Stock open(in string name);
  };
```

This interface will return a stock object of a specified category. For example, a luggage stock management object. When the requested object is not instantiated, an instant of this interface will be created and be resided on the CORBA server then the reference of it will be returned. When the requested object had already existed, a reference of this object will be returned.

- **Category Stock Interface**

```
interface Stock {
    float check_price(in string pid, in long quanity)
raises (out_of_stock, internal_error);
    boolean order(in string pid, in long quanity) raises
(internal_error);
    boolean reset() raises (internal_error);
  };
```

This interface represents a stock manager of a specific category. It provides 3 methods for maintaining the stock.

Check price method will return the price of the item of this category when the item stock is enough otherwise an out_of_stock exception will be raised.

Order method will first check the stock of the requested item, if the quantity is enough a true value will be returned and the stock value will be updated. If there is not enough stock, a false will be returned.

A reset function is provided for the stock to reset to default values.

All of the methods above will raise an internal exception when there is some non - recoverable error happens.

4.5. Itinerary Management Design

Itinerary or every TravelNet users will be stored in a database. The facilities provided are similar to shop basket. User can add a flight to itinerary from search results or remove a flight from the itinerary. Different from shop basket, itinerary update is directly done on database but not on user session. This is to ensure the consistency of the itinerary because a user can log off or quit the Web browser any time, then we have to handle this inconsistency problem if update is made on session first. A Java class is developed to handle the update of the itinerary database.

Inside the itinerary database, a unique ticket number is used to identify a ticket. Meanwhile the flight number and the name of the ticket holder will also be stored.

4.6. Flight Search and Reservation

4.6.1. Mechanism

TravelNet provides a number of ways to search for a flight. Single flight search, round trip search and direct query. Airline manager is developed for each airline for them to serve their client independent of each other. TravelNet will collect all information from these airline managers of different company and display them to the user in response to their request.

Airline manager class is responsible for resolving the reference of the CORBA airline service objects. Search parameters from user will be passed to the methods provided by CORBA airline manager objects. Result will be returned to the Servlets that handle the response page to the user. Result will be formatted and printed on the Client browser.

Based on the search results, user can select one of the flights and add it to itinerary. He/She can keep updating the itinerary by searching new flight or adding specific flight.

After a user had confirmed on a flight, he may then reserve it. The process handle client request will be connect to a CORBA airline service again and to reserve that flight ticket. For reserving a flight, payment process will be executed first then a book request will be issued to the airline service object. The availability of that flight of the corresponding class will be reduced in the database. As the airline service object informs TravelNet that the flight is reserved, this flight item will be removed from itinerary.

4.6.2. Communication Interface

♦ **Airline Service Interface**

```
string query_all(in string serv_type, in string  src_place,
                 in string dst_place, in string seat_class,
                 in string dweekday, in long mindt,
                 in long maxdt, in string rweekday,
                 in long minrt, in long maxrt,
                 in string dept_date, in string retr_date)
     raises (internal_error);
string query_one(in string flight_num,
                 in string serv_type, in string seat_class)
     raises (internal_error);
boolean is_flight_exist(in string flight_num,
                 in string weekday, in string seat_class)
     raises (internal_error);
boolean is_seat_avail(in string flight_num,
                 in string dept_date, in string seat_class)
     raises (internal_error);
string book(in string serv_type, in string holder_name,
            in string dept_fnum, in string dept_date,
            in string dept_seat_class,in string retr_fnum,
            in string retr_date, in string retr_seat_class)
     raises (internal_error);
```

Query_all will carry out a full search of the airline database according to the search options (parameters). It is used for a single filght search. For round trip, we just do single search twice and give a discount on total of two flights. In fact, the query result will be quite complex. The easiest way to return it back to client is concat the result in a string with specific format.

Is_flight_exist is called to check whether this flight exi st, a Boolean true will return if flight exists otherwise a false will be returned. Similar to this, is_seat_avail is provided to check whether the seat is available for this flight and seat class.

Book function of this interface allows client to book a flight. A unique ticket ID in a form of string will be return for successful booking, otherwise a null string will be returned.

4.6.3. Performance Comparison

We had conducted a performance comparison on CORBA and non -CORBA version for One-way flight search and round- trip reservation time (in ms). The results are as below:

Experiment 1: One-way flight search between Hong Kong and Taipei

Run	Distributed version	non-distributed version
1	19010	13139
2	15883	11146
3	16364	11878
Average	17086	12054

Experiment 2: Round-trip flight reservation between Hong Kong and Beijing

Run	Distributed version	non-distributed version
1	5668	5819
2	5828	4877
3	5734	5051
Average	5743	5249

It shows that performance of distributed version will not be better than that o f non - distributed version. Distributed components have other advantages that are essential for such as application.

Location of airline manager object is transparent to the online traveling agent, even the machine host is down, and it can be migrated to o ther host without the notice of agency. Meanwhile, it is not practical that a machine to be uncharged in such a large database. In

case of this server is down, the whole system will be down also (single point failure), so distribute it can provide a better fault tolerance.

The performance measurement above just shows the scenario of single user, if the number of user increased, a performance bottleneck will appear in a centralized system. The performance would be very poor and server may be down easily bec ause of overloading.

4.7. Web Site Map

There is not much different in the appearance of TravelNet, so the site map is more or less similar to the previous one. The updates are Hotel information and Booking of Flights.

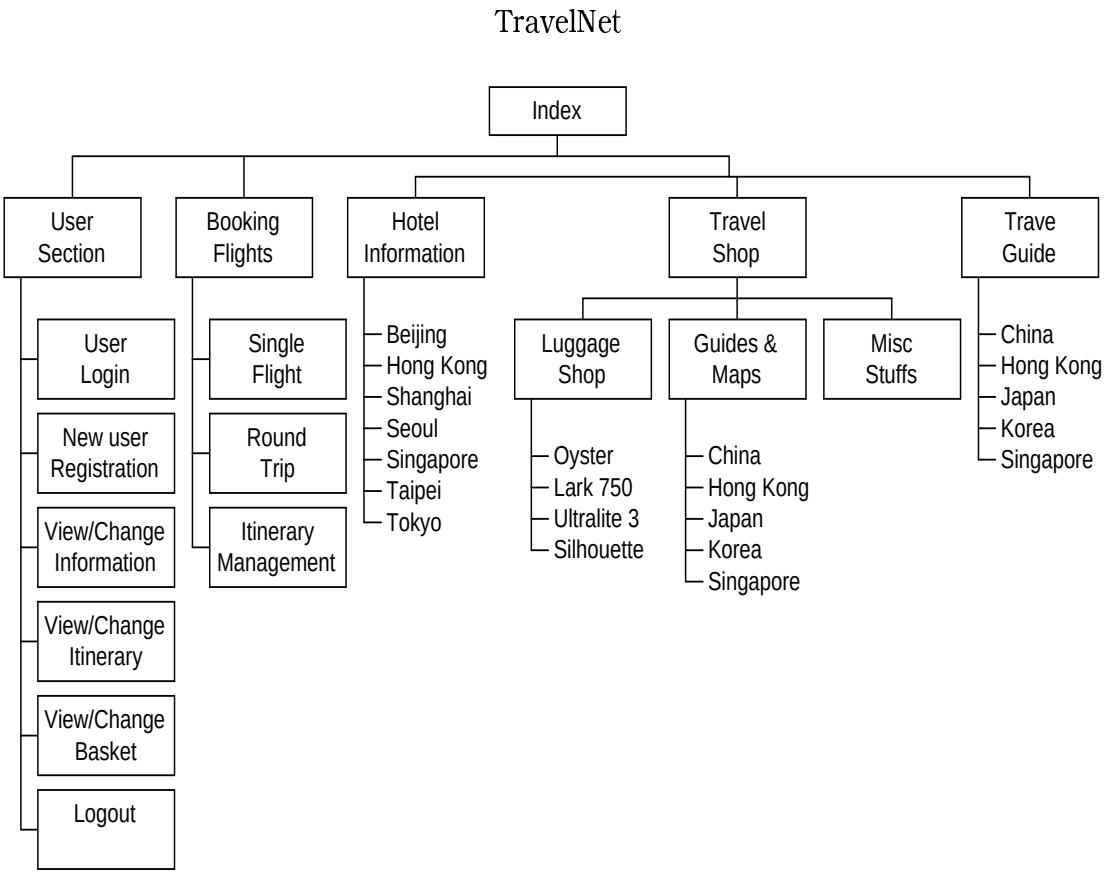


Figure 4.71. Web site map of TravelNet

4.8. Security Concerns

TravelNet as an online E -commerce application, it needs customers to provide some confidential personal information to make transaction done. One of the most important information is credit card number. Besides, name, telephone, address etc . are necessary for an Web application to provide services to them. That means these essential information must be able to send securely in order make E -commerce exists in real life. One of the most common protocol used nowadays is SSL (Secure Socket Layer).

4.8.1. SSL Introduction

SSL is the Transmission Control Protocol/Internet Protocol (TCP/IP) that governs the transport and routing of data over the Internet. Other protocols, such as the HyperText Transport Protocol (HTTP), Lightweight Directory Access Protocol (LDAP), or Internet Messaging Access Protocol (IMAP), run "on top of" TCP/IP in the sense that they all use TCP/IP to support typical application tasks such as displaying web pages or running email servers.

The basic idea of Netscape on security is that the programming for keeping your messages confidential ought to be contained in a program layer between an application (such as web browser or HTTP) and the Internet's TCP/IP layers. The SSL protocol runs above TCP/IP and below higher -level protocols such as HTTP or IMAP. It uses TCP/IP on behalf of the higher -level protocols, and in the process allows an SSL -enabled server to authenticate itself to an SSL -enabled client, allows the client to authenticate itself to the server, and allows both machines to establish an encrypted connection.

Netscape's SSL uses the public-and-private key encryption system from RSA, which also includes the use of a digital certificate. These capabilities address fundamental concerns about communication over the Internet and other TCP/IP networks:

⇒ *SSL server authentication* allows a user to confirm a server's identity. SSL -enabled client software can use standard techniques of public -key cryptography to check that a server's certificate and public ID are valid and have been issued by a certificate

authority (CA) listed in the client's list of trusted CAs. This confirmation might be important if the user, for example, is sending a credit card number over the network and wants to check the receiving server's identity.

⇒ *SSL client authentication* allows a server to confirm a user's identity. Using the same techniques as those used for server authentication, SSL -enabled server software can check that a client's certificate and public ID are valid and have been issued by a certificate authority (CA) listed in the server's list of trusted CAs. This confirmation might be important if the server, for example, is a bank sending confidential financial information to a customer and wants to check the recipient's identity. However, this function is not used as it is not a common practice for every user to apply for a client certificate before using our service. We just use our user account system for this purpose.

⇒ *Encrypted SSL connection* requires all information sent between a client and a server to be encrypted by the sending software and decrypted by the receiving software, thus providing a high degree of confidentiality. Confidentiality is important for both parties to any private transaction. In addition, all data sent over an encrypted SSL connection is protected with a mechanism for detecting tampering --that is, for automatically determining whether the data has been altered in transit.

SSL comes in two strengths, 40-bit and 128-bit, which refer to the length of the "session key" generated by every encrypted transaction. The longer the key, the more difficult it is to break the encryption code.

4.8.2 SSL in TravelNet

Due to it is reliable and popular, we decided to use this as our protocol for server authentication. There is no other way for use to provide encryption for Web communication on browsers.

Since a number of web servers and the major web browsers (e.g. Netscape and Internet Explorer) have already supported SSL, the major thing for us to use SSL is to get a server certificate and a fixed IP machine for the web server such that we can use it to apply for a digital certificate for the web server. On the other hand, client side authentication is not yet popular on the Internet, so we just limited the authentication on the TravelNet server. Once the machine is settled, we have applied a trial certificate from Entrust Technologies, which is an international CA. Trial version of the certificate works just the same as the commercial one except its valid period is shorter. The key length of this certificate is 40-bit. Although it is doesn't give the maximum security, that's enough for our purpose as the use of 128-bit is the same as 40-bit key.

After installation the certificate into the web server, the SSL connection is rea dy to use. In our system, we just need to refer our code (html) for form submission by https, which is a syntax of calling SSL through URL. An indication of the SSL enabled connection is by a small lock icon in the browser.

Figure 4.8.2. TravelNet on SSL Connection

Chapter 5. Payment Methods

5.1. Introduction

E-Commerce can't be developed if there is no secure payment methods. People who pay for the services provided by online company must be sure that they pay for their service without the risk of losing more money than the service worth. Moreover, it's unacceptable that their payment information is stole by illegal user so that those people can buy things on the Net without paying bills. When there is no trusted service for customers, they will not join into these E-activities. Providing a secure payment method is a must.

Even though it seems risky for Internet payment, there is nothing that's any risk at all. Credit cards and smart cards can also be used illegally when someone stole if from your purse. The probability of your cards are being stolen is greater than that of your cards information are being leaked to hackers.

In this session, we are going to describe two payment methods that used by TravelNet. They are credit card payment and smart card payment. By integrating these payment methods into TravelNet, it looks more realistic and complete.

5.2. Secure Payment Method on Credit Card

Credit card is the most common way for Internet Payment. Almost every one has one and money can be credited to service provider once the card information is approved. Due to this convenience, anyone who gets the card information can shop online without getting authorized by the card owner. This situation is unacceptable. The most effective way to secure the card information, that is being sent on the Net, is encrypting it.

TravelNet had made use of a payment system proposed by a post-grad student of CUHK. It's secured by using a public-private key encryption for encoding information. In the following sub-section, we will describe in detail on its system architecture, performance and security level.

5.2.1.System Architecture

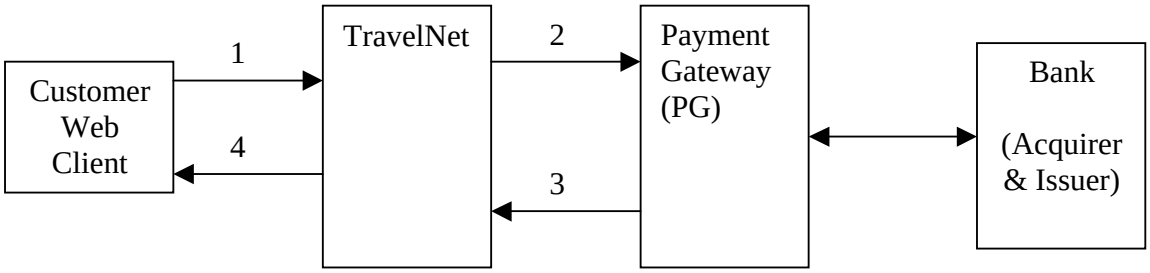


Figure 5.2.11: Credit Card Payment Architecure

1. The customer first goes to the TravelNet homepage and browses products, and puts the selected goods into a virtual shop basket. After the customer finishes choosing the products, the payment process is triggered by clicking a checkout button on the show basket user interface. A secure connection between the customer and the TravelNet is established using SSL protocol for communications. The customer then enters personal information and credit card information into the browser. In addition, the product information and the total amount will be included in the message, which is sent to the TravelNet. The message content (M1) in this step is
2. Upon the receipt of message M1, the TravelNet can get the desired item listing and credit card information of the customer. The merchant then requests payment authorization and validation of credit card from cardholder's financial institution by composing a message (M2) which consists of the customer 's personal and credit card information, together with the total amount and the merchant's name (TravelNet). This message will be encrypted by the merchant's private key to serve as an authentication. A header, which contains the merchant identification nu mber and a number, denoting the payment option the customer chose, is attached to the message. The whole message is encrypted with the payment gateway's public key to prevent eavesdropping and message tampering.
3. When the PG receives the message (M2) from the TravelNet, the PG first uses the private key to decrypt the message to get a decrypted message and a header. The PG will notice the message is sent by a specific merchant(TravelNet) but only the

merchant's public key can decrypt the header message. Nex t, PG will communicate with the issuer (the bank issue customer's credit card) and the acquirer (the bank where TravelNet account resides) through an existing banking network, which is assumed secure. After the PG receives the response from the issuer and the acquirer, the PG will compose a message (M3) including the response (whether the credit card is valid and the purchase is within the credit limit) and a receipt to the merchant for record purposes. It is then encrypted by the PG's private key for authe ntication. In addition to the message, the PG's certificate is adhered to the message. The whole message is encrypted by the merchant's public key for privacy and security purpose

4. Upon the receipt of the PG's message, the merchant will decrypt the message using the private key and then using PG's public key to obtain the original message. After checking the result, the merchant will compose an non -encrypted message (M4) to inform the customer if the purchase is successful or not. The message will be displayed as an html document for the customer. Since TravelNet only support server authentication, M4 will not be encrypted. Meanwhile the content inside M4 need not be contained any private information, as customer already knows it.

5.2.2 Security Concerns

For a system to be secure from potential attacks, it should handle the attacks on eavesdropping, message tampering and masquerading . TravelNet and the payment system are secure from those attacks.

⇒ Eavesdropping: Attackers cannot see the contents of the messa ge (M1) transferred from client browser to TravelNet server on the personal information throughout the payment process. The customer's information is encrypted by the SSL protocol. The TravelNet payment request message (M2) sent to PG is encrypted by the P G's public key. Besides, the acknowledgement message (M3) sent back to the TravelNet from PG is encrypted by the TrravelNet’s public key. Hence, no one can understand the message except the one who owns the corresponding private key for message decryption.

⇒ Message tampering: Any encrypted message cannot be tampered with, since it will not be possible to decrypt it after it has been changed. By using message digests, a digitally signed message cannot be tampered with. In M2 and M3, for example, digitally signed messages are used to prevent message tampering attack.

⇒ Masquerading: TravelNet system gets a server certificate from a trust third party for authentication purpose. Masquerading is consequently prevented on the system. Moreover, messages are authenticated with a digital signature to prevent masquerading. As a digital signature uses an owner's private key, no other people owns the private key except the owner

5.2.3. Performance Measurement

We had conducted an experiment on the performance of the payment gateway (PG) with TravelNet. In our experiments, the server always allows concurrent users to request a payment and all the requests can be executed concurrently. In TravelNet, however, can specify the type of execution scenario, either sequential or concurrent. For a single request, the total checkout time in TravelNet is between 1.7 seconds and 2 seconds. The time could be as long as 10 seconds in the worse scenario. To filter out noises, we perform 5 executions to obtain the average time measure for each data point in every experiment. The performance measurement is based on two different models: Multiple-threaded model and single-threaded model.

In the multiple-threaded model, requests are processed in parallel. Each request will obtain only a portion of the server resources, which is inversely proportional to the number of requests. For example, when there are 10 concurrent users requests, each client process will be on the average 10 times slower than each executing alone, as each of them only grasp 10% of the server resources. The time of overlapping processes will consequently be longer. There is also an extra task-switching overhead that is very significant when the number of tasks becomes large. As displayed in Figure 5.2.31, the payment process time increases as the number of concurrent user increases. We can also see in Figure 5.2.31 that the total payment process time is divided into two parts: time spent on the Merchant client and time spent on the Payment system server. In terms of

the portion of time spent for the total checkout process, payment server contributes over 80%.

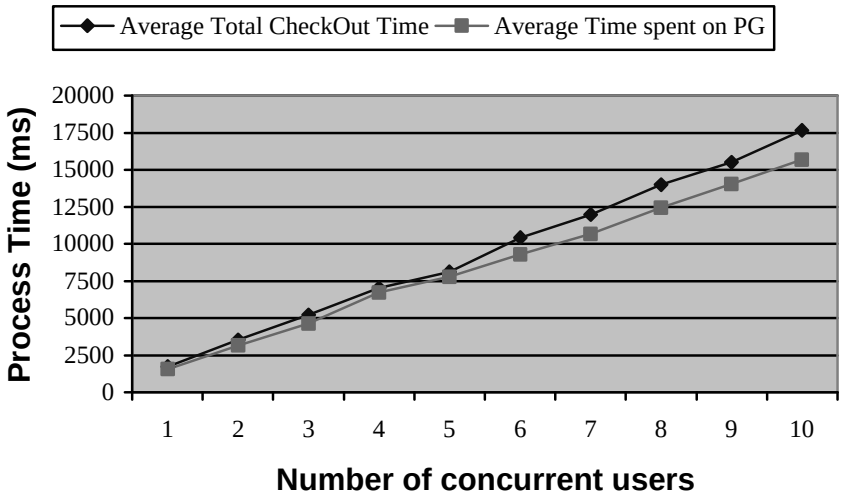


Figure. 5.2.31: Payment Transaction Time in Multiple-Threaded Model

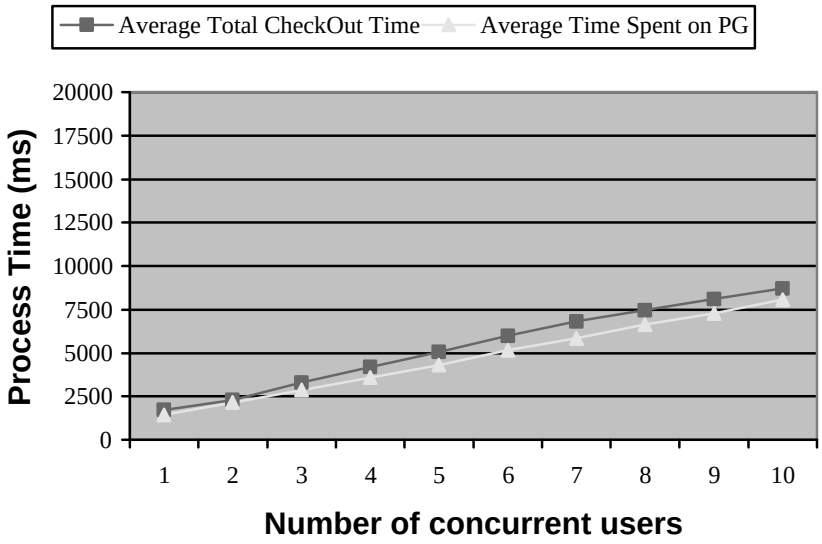


Figure. 5.2.32: Payment Transaction Time in Single-Threaded Model

In the single-threaded model, TravelNet clients request in a first-come-first-serve manner. Every request waits for all the previous requests to be finished before it can gain access to the server resources. Figure 5.2.32 shows th e average total process time and the time spent on PG for the single -threaded model. As a comparison, we can see from Figure 5.2.33 that its average process time is much shorter than that of the multiple -threaded

model. The main reason is due to database r esource conflict for the multiple -threaded model when the multiple concurrent processes access the PG, which currently has only one merchant, namely, TravelNet. As the PG server resources have to be shared among the multiple requests, the requests will hol d resource (e.g., lock a data item) and compete with each other, thus delaying the complete time. In the single -threaded model, server resources are not shared among the requests and only a task -switching time is necessary between each request. As the resp onse time is quite important in such an interactive application, the single-threaded model behaves better than the multiple-threaded model. It is noted, however, that if we have multiple merchants in the PG, which handles different requests with independent merchants, the multiple-threaded model would be significantly improved.

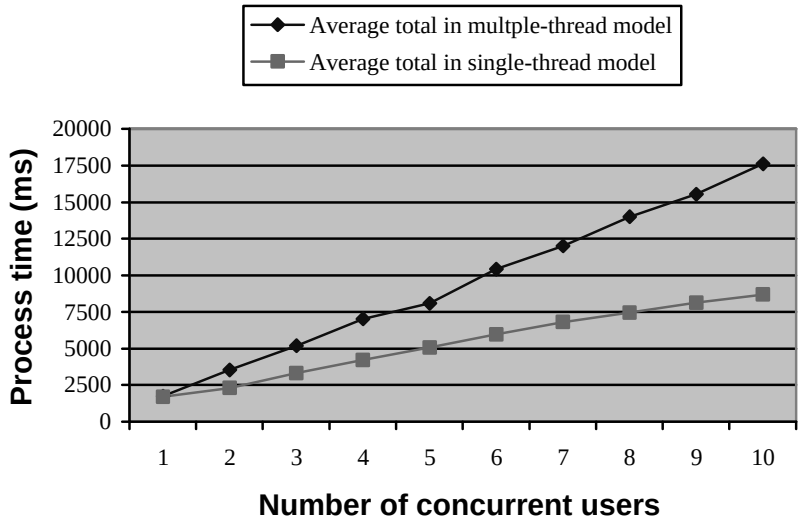


Figure. 5.2.33: A Comparison for Single-Threaded and Multi-Threaded Model

For a better performance for TravelNet, we decided to use a Single -Threaded model for TravelNet's checkout policy for credit card.

The payment processing time can be divided into two parts as well: the time required to perform cryptography algorithms (including message encryption and decryption), and the time required to transmit messages and handle payments. Figure 5.2.34 shows the comparison on the payment process time on the PG regarding the overhead due to cryptography. We found that when the number of concurrent users increases, the gap

showing the difference on the process t ime between using cryptographic algorithms and without using them becomes larger. This overhead indicates that for a more secure payment system, there is a tradeoff on the time to handle payment transactions. This tradeoff is quantitatively provided in TravelNet for a detailed analysis.

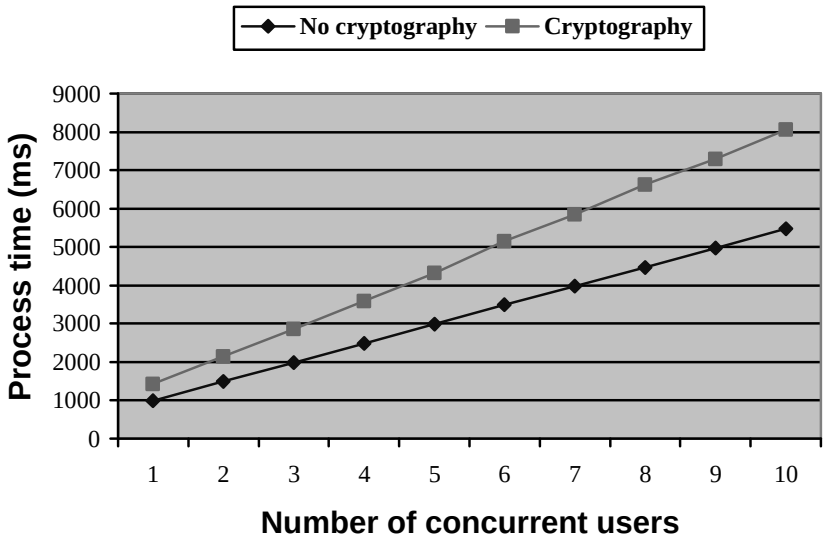


Figure. 5.2.34: Single-Threaded Model on the Payment Transaction Time on PG

5.3. Micro Payment Method

Smart cards becoming more and more popular over Internet payment since it is convenient and more secure. Mondex is some of the most famous store -value smart card in the market. On the Mondex card, there is a microprocessor embedded on it. This microcomputer has been programmed to function as an "electronic purse". The electronic purse can be loaded with value, where it is stored until it is used as payment for goods or services at retailers or service outlets or transferred to another Mondex Card, by inserting the Card into a card reader. The electronic purse can also be locked using a personal code so that only the card's owner can access the value on it.

We met the golden chance that there is a joint project between the Center of Innovation and Technology of CUHK and the Mondex developing company for the testing of Mondex in a medium size comm unity, CUHK campus. A Mondex payment server is ready to use in our computer science department, and we had obtained the equipment that is necessary for accessing this server and can have a try on actual Mondex payment on the Internet. We integrate it into TravelNet so as to provide micro-payment services.

5.3.1. System Architecture

A Brief Description

The concept is as follows.

1. The consumer checks out at the TravelNet, the merchant prepares and signs the payment request, and gives it to the consumer.
2. The consumer goes to the payment server and submits the payment request.
3. The payment server verifies the signature of the payment request. If it is correct, it proceeds with the payment.
4. The payment is completed, Payment Server prepares and signs the payment result, and gives it to the consumer.
5. The consumer goes to the merchant again and submits the payment result.
6. The merchant verifies the signature of the payment result. If it is correct, it proceeds with the post-payment processing.

The Payment Flow

The figure below shows the flow of a Mondex payment using digital signature.

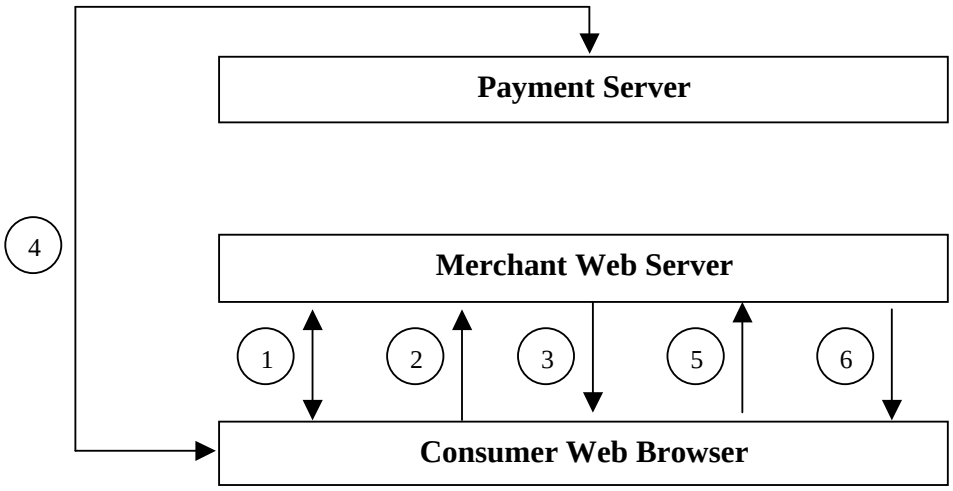


Figure 5.3.11. The Mondex Payment Flow Using Digital Signature

1. Shopping. A consumer reaches a TravelNet and logged in, he or she either interact s with the TravelNet shopping system or selects the desired products. After they selected the desired products, he/she wants to pay for the service charge, for example to pay for the electric bill.

   All the item selected will be displayed in the shop baske t of TravelNet. After the customer had confirmed to buy the items in the shop basket then they can issue a check out operation to start the payment.

2. Confirm the payment. From the payment confirm web page, the consumer selects one of the available paymen t methods, which can be Visa, Master and Mondex. Finally the consumer presses the Confirm Payment button to confirm the payment on Mondex.

3. Upon requesting a checkout by Mondex, a server program will be run and it does the followings:
   (i) Check whether the state of payment is valid.
   (ii) Construct the payment request from database.

(iii) Sign the payment request using the Mondex Merchant utility library provided by the developer.

(iv) Construct a web page embedding the Consumer Mondex Payment plugin program reference and the corresponding plugin input arguments, and send it to the consumer. The plugin arguments contain the payment request and the merchant signature on the payment request.

4. The customer plugin connect to Payment Server and start the payment. Upon the consumer receives the web page containing the plugin reference, the plugin is invoked. The plugin connects to the payment server via SSL. It authenticates the Payment Server and then submits the payment request to it. Payment Server first verifies the signature of the request, then queues it up; and eventually the Mondex payment between a merchant Mondex card and the consumer Mondex card begins. Finally, the result of payment will be signed by Payment Server and send to the consumer plugin.

Figure 5.3.11. A snapshot of plugin

5. Submit the payment result. The consumer plugin calls a another processing Servlet from TravelNet, say Result, to submit the signed payment result received from Payment Server.

6. Deliver the post-payment web page. The Result program first verifies the signature of Payment Server using the library provided. If it is correct, it does the post-payment processing and responses a web page to the consumer. It sends a web page containing the payment result and reference number to the consumer.

5.3.2. Mondex Client Equipment

Every Mondex client will have a light weight card reader called iReader (figure5.3.22). It must be plugged into the machine that is used for browsing Internet for shopping. Besides, a driver of this device should also be installed. A customer management con sole program (figure 5.3.21) is available for custom to manage his Mondex card and keep track of payment Records. In this program, user can also can the currency to display, locking the card and checking the card status.

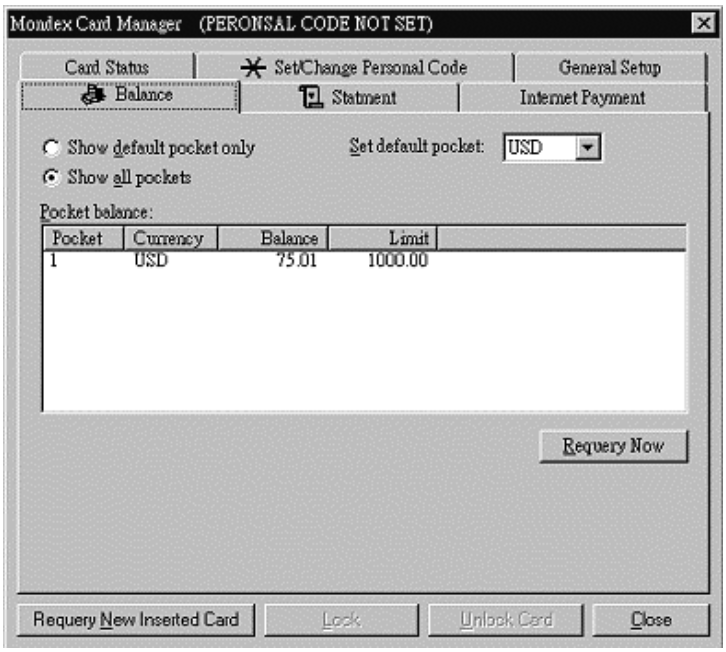


Figure 5.3.21. Mondxe management program

Figure 5.3.22. iReader

5.3.3.The Benefits

Eliminate the communication between the merchant and the Payment Server
It saves the processing power and communication bandwidth on both the Payment Server and the merchant web server on establishing the SSL connection since client will directly call a plugin to connect to payment server and process payment.

More convenient shopping
For the use of credit card, customer has to fill in detailed information and send it over the Internet. This process seems troublesome and customer will naturally worry about the security of the payment. By Mondex, all the client has to do is insert the card properly then issue a start operation on payment. This sounds more easy and convenient.

Favor non credit-card-holder
Not every one own a credit for shopping on Internet since applying for a credit card have some restrictions like age and income. Mondex Card favors those who have no credit card. They can still enjoy the convenience provided by nowadays technolog.

Limited value
Mondex cards have a maximum stored value limit. Even an unlocked card has been stolen by others, the maximum lost in monetary value will be at most the maximum value of a Mondex card.

5.5.4. Security Concerns

Replay of messages

Since the consumer acts as the middle party on delivering the payment request and payment result between the merchant and the Payment Server, it is possible that the consumer captures the payment request or result and re -submit them later in order to get any benefits or interrupt the merchant service. Both the Payment Server and Content Server should be designed to detect and eliminate any replayed payment requests or results.

Detection of Replayed Payment Request in Payment Server

To detect the replayed payment request, the principle is to make each payment request *unique*. Payment ID i s not adequate, because the same payment ID may be used in payment resume. To achieve that, each payment request will have a GMT time stamp appended. The time stamp is generated by the merchant and specified in the plugin -input parameters. The time stam p is signed, so any change to it will be detected. Hence,

Payment ID together with the GMT time stamp will used to identify a payment in Payment Server.

Payment Server will keep a history of the received payment requests. It will check against each inco ming payment with the history to detect if any replayed request. However it is not feasible to keep each received payment request in history or memory due to limited system resources. Therefore, Payment Server will first check if the time stamp is matche d with the current time. Since there is time difference in the merchant web server and the Payment Server, a tolerant range of time difference say 2 hours is introduced. If the received payment request is outside this range of time, it is rejected. If it is within the range of time, it will be checked against the history. So the history needs only to contain 2 hours of payment request. The history of payment request will also be saved to disk and be read back when Payment Server is re-started next time. The tolerant time difference is configurable.

Detection of Replayed Payment Result in Content Server

To detect the replayed payment result, the GMT time stamp from the payment request will be put in the payment result sending to Content Server. Theref ore, provided the Content Server is designed to use both the Payment ID and the GMT time stamp to identify a payment, the replayed payment result can be detected.

Chapter 6. Conclusion

We have successfully finished a complete E -commerce application, with sophisticated services and payment methods. Besides some external qualities, we also devote lots of efforts on the modular and structural deigns.

In the last term, TravelNet contain a number of services, they included member ship management, Travel Shop, s imple single flight search and some travel guides. They are developed in centralized manner and all the access of database is directly from Servlets process, which also handle the output layout. Credit card payment is simulated by simple database access. M ost of the effort last term spent on testing and investigating some suitable software tools, Web security and hardware for further development of TravelNet. That's why TravelNet seems incomplete and premature in that stage but we prepared a good base for us to develop a better system in the coming term.

In this term, we concentrate on distributing system components and payment methods incorporation. For a more modular design on Web layout design and process component, we make use of the concept of Java bean and Java Server Page. CORBA integration for distributed components is another major enhancement of TravelNet. CORBA Flight managers and CORBA stock managers are developed and run with TravelNet in a distributed manner for a better performance and toleranc e. By the way, we add some more services like more flight search option, reservation, itinerary manager and hotel information.

Payment methods are a great advancement of TravelNet. When combined with credit card payment and Mondex payment, it makes TravelN et more realistic. Especially for Mondex payment, we can gain the experience of the mechanism of real life micro - payment system.

As a conclusion, it is a rewarding project for us and the effort we paid on this is a useful experience that we gain for us to devote to the E-society.

Chapter 7. Reference

[1] B.Eckel. *Thinking in Java*, Prentice Hall Inc. 1998.

[2] "JDK™ 1.1.8 Documentation".
http://java.sun.com/products/jdk/1.1/docs/index.html

[3] "The Java Tutorial".
http://java.sun.com/docs/books/tutorial/

[4] Victor Wolters. *Introducing Internet Information Server*, Que. Oct 14, 1996

[5] "Security in Internet Transaction".
http://www.holt.ie/text/security.html

[6] "Web Application Development".
http://www.winwinsoft.com/articles/wad.html

[7] "Expedia.com".
http://expedia.msn.com

[8] "Travelocity"
http://www.travelocity.com

[9] "Introduction to SSL"
http://developer.netscape.com/docs/manuals/security/sslin/index.htm

[10] "How SSL works"
http://developer.netscape.com/tech/security/ssl/howitworks.html

[11] C. Darby, "Developing 3-Tier Database Apps w/ Java Servlets", *Java Developers Journal*, Feb 1998

[12] IBM Corporation. "The Web Application Programming Model". *IBM Application Framework for e-business*. IBM Corporation.

[13] Z.Yang, K.Duddy. "CORBA: A Platform for Distributed Object Computing ". Operating Systems Review, 30(2):4-31. ACM SIGOPS, Apr. 1996.

[14] "The Design, Implementation and Evaluation of an Internet Payment System", K. L. CHONG, C. H. HO, C. H. LAU, MICHAEL R. LYU, Y. S. MOON, Feb 2000.

[15] "Mondex Electronic Cash."
http://www.mondex.com

[16] “Cyber-A Corporation - iReader”
http://www.willas-array.com/sct/products/ec/mouse.htm

[17] “VisiBroker for Java 4.0”
http://www.borland.com/techpubs/books/vbj/vbj40/framesetindex.html

[18] “New Atlanta – ServletExec 2.2”
http://www.newatlanta.com/

[19] “JavaServer Pages(TM)”
http://java.sun.com/products/jsp/

Appendix

A. Software

⇒ **Java API 1.1.8** : Java is an object -oriented language, which is poplar all around the world today. Because of its portability, it grows along with the Internet related technologies. Its complete and robust API brings pro grammer and software developer a convenient developing environment. Since it is slower than native programming language, Java is not suitable for low level programming or real time processing. On the other hand, it is perfect for net working application programming.

⇒ **Java Servlet API** : Servlets are the Java platform technology of choice for extending and enhancing Web servers. Servlets provide a component -based, platform-independent method for building web -based applications, without the performance limitations of CGI programs.

⇒ **Java Server Page**: It uses a format similar to HTML tags. It includes special tags for including Java scriptlets. Direct usage of component beans help to separate web design and application logic. JSP will be compiled to Java object code and can be start service on Java enable Web Server.

⇒ **Windows NT Server 4.0 with IIS 4.0:** Windows NT Server is a quite common commercial product Microsoft Windows NT Server 4.0 is a multipurpose operating system specialized on Server operations . IIS is a co mmon Web server for NT servers. It supports SSL server authentication and capability of add new module for Web services.

⇒ **Oracle 8I:** A popular database server. TravelNet make use of this DBMS as data storage. Oracle8i, the database for Internet computing, c hanges the way information is managed and accessed to meet the demands of the Internet age, while providing significant new features for traditional online transaction processing (OLTP) and data warehouse applications. It provides advanced tools to manage all types of data in Web sites, but it also delivers the performance, scalability, and availability needed to support very large database (VLDB) and mission-critical applications.

⇒ **Servlet Exec 2.2** :ServletExec is a Servlet engine. It is a high -performance, reliable, inexpensive web application server and Servlet engine that implements the Java Servlet API and JavaServer Pages (JSP) standards, components of the Java 2 Platform, Enterprise Edition (J2EE) suite of standards defined by Sun Microsystems. ServletExec runs on all major web servers and operating systems.

⇒ **Borland Visibroker 4.0:** VisiBroker is a complete CORBA 2.3 Object Request Broker (ORB) that supports the development, deployment, and management of distributed object applications across a variety of hardware platforms and operating systems. In addition to VisiBroker (the ORB), three other components are available with this product. They include

- Naming Service
- Event Service
- Gatekeeper

⇒ **Mondex Merchant Utility:** A utility provide a function for me rchant to sign a payment request and verify a payment result. This is the essential function for Mondex payment.

B. Hardware

⇒ **Web Content Server:** Pentium II 300MHz, 96-MB memory. A mid-end machine is needed for a web server to handle requests concurrently especially our system request handler is Java Servlet. A Pentium 2 300MHz is just meet our demand. It is a server with a static Internet address. The Internet name is ntsvr4.cse.cuhk.edu.hk.

⇒ **CORBA Server:** A number of Sun Ultra workstations, 128 -MB, 100Mbps - network speed with Unix Operating System. A nice environment for distributed network.

⇒ **Mondex iReader, Test Card:** Test card has a certain value inside for testing purpose no actual value. IReader is a device for reading data from Mondex Cards. This device will be connected to a COM port of a machine.

C. Client Requirement

⇒ **Netscape 3.0+ or Internet Explorer 4.0+:** Travel Net client only needs a simple web browser. It is recommended that client browser is SSL enable because the client will submit critical informati on through the Internet. This unprotected transmission is very insecure. If information is being hacked, hacker may use this information for illegal shopping.

⇒ **Mondex customer plugin:** A plugin program must have to be install in the client machine. It will be invoked when a Mondex payment is issued from merchant.

⇒ **Mondex iReader:** Used to read and process with Modex card.

D. Program Listing

Module	Sub module	Number of Lines	Number of characters
User Profile Management	Login.jsp	90	3518
	LoginBean.java	110	2428
	UserSessionBean.java	53	1003
	Register.java	238	8981
	ViewUserInfo.java	178	7036
	UpdateInfo.java	153	5582
	Logout.java	20	464
Sub Total		842	29012

Travel Shop	Shop.jsp	120	3427
	ShopBasketBean.java	44	1383
	ViewBasket.jsp	152	5730
	CheckOut.java	250	9327
	mondex.jsp	90	3580
	Mondex.java	78	1930
	Result.java	265	9248
	mondex.bas	72	2299
Sub Total		1071	36924

Stock Management	Stock.idl	17	391
	StockMgrImpl.java	20	547
	StockServer.java	25	747
	StockImpl.java	107	2931
	StockBean.java	62	1841
Sub Total		231	6457

Airline Service	AM.idl	27	1144
	AirlineManager.java	498	13716
	SearchFlight.java	510	21009
	RserveFlight.java	353	13596
	AirlineServer.java	53	1843
	AirlineServiceImpl.java	484	14124
Sub Total		1925	65432

Itinerary Management	ItineraryManager.java	482	13211
	AddItinerary.java	84	2539
	ViewItinerary.java	293	14074
	RemoveItinerary.java	43	1236
Sub Total		902	31060

Hotel Information	hotelresv.jsp	175	7881
	Hotel.jsp	60	1748
Sub Total		235	9629

Supplemantary Classes	Mail.java	39	1471
	Html.java	20	523
	Database.java	45	1373
Sub Total		104	3367

Total Number of lines = 5310

Total Number of Characters = 181881