ABSTRACT

In the current communication world, smart phones are leading the mobile industry. A smart phone is the fully featured mobile phone with computer capabilities. Most of the smart phones have video, audio and email features. Smart phones are generally called as “high end” phones with QWERTY keyboards or touch screens and are data enabled for Internet access. Most of the smart phones are using the Symbian Operating System for the applications to run effectively. Symbian OS is the advanced, open operating system licensed by leading mobile phone manufactures.

Current Symbian OS based mobiles have features like email sending and accessing some Web pages. There is no specific application for buying airline tickets from desired airline services. This project will go beyond and design such an application compatible with the Symbian OS using Symbian C++ for 2.5G and 3G mobiles like Nokia S60, Motorola, and Sony Ericsson models.
# TABLE OF CONTENTS

Abstract ................................................................................................................... ii

Table of Contents .................................................................................................... iii

List of Figures .......................................................................................................... vi

1. Background and Rationale .................................................................................. 1
   1.1 Introduction to the Symbian OS ................................................................. 1
   1.2 Why was the Symbian Operating System (OS) designed? ....................... 3
   1.3 Symbian OS structure .................................................................................. 3
       1.3.1 Hardware Resources ......................................................................... 4
       1.3.2 Software Components ...................................................................... 5
       1.3.3 Processes, Threads and Context Switching ........................................ 6
   1.4 Characteristics of the Symbian Operating System ..................................... 6
   1.5 Key features of the Symbian Operating System ......................................... 7

2. Narrative ............................................................................................................. 9
   2.1 Technology Used in Symbian Operating System ...................................... 9
   2.2 Existing Mobile Applications .................................................................. 11
   2.3 Designed Mobile Application .................................................................. 11
   2.4 Need for New System ............................................................................... 12
   2.5 Preliminary Investigation .......................................................................... 12
   2.6 Feasibility Study ....................................................................................... 13
   2.7 Application Interface ............................................................................... 14
3. Project Requirements, System Design, and Implementation..........................16
   3.1 System Requirements.................................................................16
   3.2 Use Case Diagrams.................................................................17
   3.3 Class Diagrams........................................................................18
   3.4 State Transition Diagrams..........................................................19
   3.5 Data Flow Diagrams.................................................................22
      3.6.1 DFD Symbols......................................................................22
   3.6 Interface Design.........................................................................27
   3.7 Project Development Tools..........................................................35
      3.7.1 Carbide.C++ IDE.................................................................35
      3.7.2 Symbian SDK.......................................................................36
      3.7.3 JRE 1.4................................................................................36
      3.7.4 ActiveState ActivePerl 5.6......................................................36
      3.7.5 Java Development Kit 1.4......................................................36
      3.7.6 SQL Server 2005...................................................................36
   3.8 Symbian.C++ and Java Code Implementation......................................37
   3.9 Security Measures ......................................................................37
   3.10 Real time implementation of the application ......................................41
      3.10.1 Creation of “.sis” file for real time implementation....................41
4. Testing and Evaluation.........................................................................42
   4.1 Unit Testing..................................................................................42
      4.1.1 Black Box Testing.................................................................42
      4.1.2 White Box Testing.................................................................43
4.2 Test Cases .............................................................................................................43

5. Future Work .............................................................................................................45

6. Conclusion ...............................................................................................................46

Bibliography and References .......................................................................................47
LIST OF FIGURES

Figure 1.1 Symbian Ltd partners ................................................................. 1
Figure 1.2 Symbian Licensees ................................................................. 2
Figure 1.3 Devices based on Symbian OS .................................................. 2
Figure 1.4 Nokia 9210 ............................................................................. 4
Figure 1.5 Software components architecture ......................................... 5
Figure 2.1 Symbian OS v7.0s Architecture .............................................. 9
Figure 2.2 Application Model .................................................................. 15
Figure 3.1 Use Case Diagram for Mobile Ticketing ................................. 18
Figure 3.2 Class Diagram for Mobile Ticketing ........................................ 19
Figure 3.3 State Transition Diagram for Mobile Ticketing ...................... 21
Figure 3.4 Context Level Diagram for Mobile Ticketing ......................... 24
Figure 3.5 Level-1 Diagrams for Mobile Ticketing .................................... 24
Figure 3.6 Level-1 Diagrams for Mobile Ticketing .................................... 25
Figure 3.7 Level-1 Diagrams for Mobile Ticketing .................................... 25
Figure 3.8 Level-1 Diagrams for Mobile Ticketing .................................... 26
Figure 3.9 Level-2 Diagram for Mobile Ticketing .................................... 26
Figure 3.10 Different options in the application ....................................... 28
Figure 3.11 Enquiry screen for the user .................................................... 29
Figure 3.12 Seats availability screen with time and flight number ............ 30
Figure 3.13 Screen to enter number of seats .......................................... 31
Figure 3.14 Structure of seats arrangement ............................................. 32

v
Figure 3.15 Screen for purchasing air ticket .............................................33

Figure 3.16 Confirmation screen after purchase .......................................34
1. BACKGROUND AND RATIONALE

1.1 Introduction to the Symbian Operating System (OS)

The Symbian Operating System is the standard operating system for smartphones, and was specially designed for mobile devices by Symbian Ltd. Symbian Ltd was founded in 1998, initially from Psion, Nokia and Ericsson. Since then, Siemens, Samsung, Sony Ericsson, Panasonic and Motorola have all become partners. Symbian is a software licensing company - its core business is to supply the advanced, open operating system, Symbian OS, for data-enabled mobile phones.

Many major mobile manufacturing companies like Ericsson (in 1998), Nokia (in 1998), Panasonic (in 1999), Samsung (in 2002), and Siemens (in 2002) became partners with Symbian Ltd. to improve the mobile applications. [Symbian 2006]

![Symbian Ltd partners](image)

**Figure 1.1 Symbian Ltd partners**

In addition to the partners of the company, there are a number of other companies that license the Symbian OS for use in their own products. Symbian Licensees are illustrated in Figure 1.2:
Figure 1.2 Symbian Licensees

Figure 1.3 depicts devices using the Symbian Operating System, starting from the year 1999 to 2005. Most of them are Smartphones with many applications; these phones are becoming popular day by day because of they are user friendly by nature.

Figure 1.3 Devices based on Symbian OS
1.2 Why was the Symbian Operating System (OS) Designed?

The Symbian OS was designed specifically for mobile devices and as such has small memory footprint and low power consumption. This is very important, as users do not want to recharge their phone every day! It is an open OS, enabling third party developers to write and install applications independently from the device manufacturers. An extensive C++ API is provided which allows access to services such as telephony and messaging, in addition to basic OS functionality. Some devices that run the Symbian OS may not be switched off for years; therefore the OS was designed so applications could run for years without losing user data. Finally, the OS can run on more than one hardware platform, so it can be used on a variety of device types including those with touch screens and those with pens or keyboards. [Symbian 2006]

Symbian OS is the current name of the operating system, but when it was initially released as "EPOC". EPOC was a family of operating systems developed by Psion for portable devices; EPOC was rumored to be an acronym for “Electronic Piece of Cheese”. The name EPOC was used for some time and will still be found in class/file names and in older documentation. The name EPOC is still used to refer to the kernel.

1.3 Symbian OS structure

Symbian Operating System has a structure the same as an operating system designed for PC’s (personal computers), with similar components like hardware resources, software components, processes, threads, context switching, kernel and library.
1.3.1 Hardware Resources

The following are some of the hardware resources built for mobile devices supporting the Symbian operating system:

- CPU Advanced RISC Machine (ARM), 32 bit, 36-220MHz
- RAM: working memory, disk space, C: drive.
- I/O: keypad, pen input, Memory card slot.
- Communication: GSM, GPRS, UMTS, infrared.

The Nokia 9210 device shown in Figure 1.4 has 14MB ROM, 8MB (SD-RAM), 16MB MMC Card, 4096 color screen, and a full keyboard. This model is even used as personal organizer having computer like capabilities, can connect to Internet and browse through sites and download the stuff with the support of hardware resources. [Symbian 2004]
Memory organization and Disk drives:

Mobiles have C, D, E and Z drives meant for different purposes. C drive has flash RAM which holds application files and user files. D and E drivers are used to browse through flash cards and memory sticks. Z drive holds Operating system files.

1.3.2 Software Components

Kernel, application, server and, engine are software components in the Symbian operating system architecture depicted in Figure 1.5. Kernel manages and controls access to hardware resources with hardware supported privileges in kernel mode. [Symbian 2004]

![Figure 1.5 Software components architecture](image)

Application is the program with user interface which runs in a user mode in its own process, the structure shown in Figure 1.5 is specially designed for single (mobile) application. Server component is also the program which manages resources and provides interface to clients without any user interface. Engine is the application part that manipulates data with a separate DLL (Dynamic Link Layer); engine basically lies in application component to manipulate input data using specified method depending on the input type. Type of engine keeps changing depending on the application type.
1.3.3 Processes, Threads and Context Switching

A process is a fundamental unit of program execution with its own address space, whereas a thread is a fundamental unit of execution with one or more threads per process and is preemptively scheduled by the kernel. Threads improve efficiency and speeds up the calculations and responses.

Context switching is the switching of threads in the processes to improve the response time of the application; it is expensive if switching of threads is done between different processes because thread communication between different processes takes much effort whereas switching of threads within a process is very easy to program.

Objects are active if there is one thread for each process executing part of code.

1.4 Characteristics of the Symbian Operating System

The following are the important characteristics of the Symbian operating system:

- **Integrated multimode mobile telephony** – Symbian OS integrates the power of computing with mobile telephony, bringing advanced data services to the mass market

- **Open application environment** – Symbian OS enables mobile phones to be a platform for deployment of applications and eservices (programs and content) developed in a wide range of languages and content formats

- **Open standards and interoperability** – With a flexible and modular implementation, Symbian OS provides a core set of application programming interfaces (APIs) and technologies that is shared by all Symbian OS phones. Key industry standards are supported
• **Multi-tasking** – Fully object-oriented and component-based, Symbian OS includes a multi-tasking kernel, middleware for communications, data management and graphics, the lower levels of the graphical user interface framework, and application engines

• **Robustness** – Symbian OS maintains instant access to user data. It ensures the integrity of data, even in the presence of unreliable communication, and shortage of resources such as memory, storage and power [Symbian 2006]

1.5 **Key features of the Symbian Operating System**

Symbian operating system has many features; following are the key features among many features:

The Symbian Operating System has a rich suite of application engines – including contacts, schedule, messaging, browsing, office, utility and system control; OBEX to exchange objects such as appointments and business cards integrated APIs for data management, text, clipboard and graphics.

It also supports multimedia messaging using MMS, picture messaging with EMS and text messaging using SMS. Internet email using POP3, IMAP4, SMTP, MHTML, standard attachments and fax. Also supports browsing for full web browser support and WAP stack for mobile browsing.

Another key feature is International locale support – with native Unicode characters, flexible text input framework, and additional font and text formatting (supporting the Unicode Consortium standard). Mobile telephony feature is the abstract API for cellular standards. GSM circuit-switched voice and data (CSD and EDGE ECSD) and packet-based data (GPRS and EDGE EGPRS); CDMA: circuit-switched voice and
data and packet-based data (IS-95 and cdma2000 1x); SIM Application Toolkit and SMS.

Other standards can be implemented by licensees due to the extensibility of the APIs.

[Symbian 2006]
2. NARRATIVE

2.1 Technology used in the Symbian Operating System

Symbian strategic partnership with ARM (a digital architecture company) made to meet requirements like good memory performance, good multimedia and system-on-chip design, Symbian uses Thumb Instruction Set Architecture (ISA’s) to provide security. This operating system uses 32 bit microprocessor to make applications work faster when compared to other 16 bit microprocessor systems. Symbian architecture proves to have everything in it.

Figure 2.1 Symbian OS v7.0s Architecture
The Symbian operating system architecture consists of telephony, communication infrastructure, messaging, multimedia, graphics, application engines, application framework, PAN (personal area networking). [Steve 2007]

**Telephony:** The Telephony subsystem provides a multimode API to its clients. The abstract cellular networks include GSM, GPRS, EDGE, CDMA (IS-95) and 3GPP2 cdma2000 1x (Release A) and is ready for 3GPP W-CDMA making it easier for handset manufacturers to port Symbian OS from one mobile phone standard to another.

**Communication infrastructure:** The Communication Infrastructure subsystem provides the key frameworks and system services for communication and networking. A communications database manager controls the system-wide communications configuration. It also consists of a network interface manager which provides a framework for connection to other computers or networks. The manager provides a mechanism for the client to monitor progress over e.g., a PPP connection.

**Messaging:** The messaging framework provides support for messaging protocols for sending and receiving text messages (SMS), enhanced messages (EMS), multimedia messages (MMS), e-mail and fax messages. The framework uses polymorphic Message Type Modules (MTMs) to handle specific types of message. MTMs can be added at runtime to expand the messaging capabilities of Symbian OS phones after market.

**Multimedia:** The Multimedia Framework (MMF) provides a lightweight, multi-threaded framework for handling multimedia data. The framework provides audio recording and playback, audio streaming and image related functionality. Support is provided for video recording, playback and streaming.
**Graphics:** The graphics subsystem provides Symbian OS applications with shared access to the screen, keyboard and pointing devices input, bitmap fonts and scalable fonts (provided through the Open Font System), and bitmaps by using a shared heap. It also implements the Graphics Device Interface (GDI), providing a generic framework for drawing to any graphics device, and supplies concrete implementations for drawing to windows, bitmaps or to a printer.

The Symbian operating system supports to embed as many applications as possible; and there are many applications and suitable technologies to integrate more applications on Symbian compatible mobiles.

### 2.2 Existing Mobile Applications

Current generation mobiles (smart phones) based on Java technology, Windows and Symbian systems have many useful applications. Some of the applications are mobile banking, mobile maps (Map quest, Google maps), mobile e-mailing, mobile chatting, mobile videos, mobile myspace, mobile weather, mobile games, mobile radio and many more. All of these applications are making life easier and many more applications are coming up to catch current requirements.

### 2.3 Designed Mobile Application

At present a user/person can reserve his air tickets in two different ways, one is over the reservation counter and other is online reservation. In the manual reservation system user has to go to particular booking counter to reserve his tickets. Though this process does not involve much of the technology it grabs user time. Online reservation allows the user to reserve his ticket through the Internet. This process is very comfortable when compared to the manual reservation, but sometimes it may happen that the
computer is not available with Internet. At this point of time reservation can be possible using a mobile device, which is done in minutes without a computer or standing in lines. Yes, it’s very convenient if mobile provides tickets to our destination with just some clicks on the mobile. This project is involved in designing an application to reserve airline tickets using smart phones by taking advantages of the Symbian operating system with support of Java and other technologies.

2.4 Need for New System

Sometimes a user may be in an emergency situation and wants to buy a ticket; this application would be useful to buy a ticket instantly. If the user is already traveling and wants to reserve another ticket then it’s easy and fast with this application to get another ticket. At present the user is able to do this by using online reservation, which may not be possible if he does not carry a laptop. So there is a need for powerful and flexible solution that allows the user to reserve his tickets through a hand held device [Mobile /PDA] from the place where he is. This application will prove the technology advancements as well as it will increase the convenience for the people who fly frequently.

2.5 Preliminary Investigation

As this is a mobile application there is a need of communication between mobile and a system (say server) which serves the request. To make this communication possible services of the carriers to connect the mobile to the database server are needed. In order to deal with the carriers a server which continuously interacts with the carriers is needed. And in order to deal with the service provider one more server is needed. The communication between these servers is point to point connection less network service.
2.6 Feasibility Study

“A feasibility study is a preliminary study undertaken before the real work of a project starts to ascertain the likelihood of the project's success”. It is an analysis of possible alternative solutions to a problem and a recommendation on the best alternative. All the projects are feasible when given unlimited resources and infinite time. This is not possible in practical world and hence there is a need for feasibility studies.

A feasibility study is a sort of simulations of the future development process that helps decide whether the development process is worth while, and if so which development process should be followed. Actually feasibility of any project is checked for the following four levels.

**Economic Feasibility:**

Economic feasibility is the most frequently used method for evaluating the effectiveness of a candidate system. More commonly cost- benefit analysis, the procedure to determine the benefits and savings that are expected from a candidate system are compared with the cost.

Mobile Ticketing is economically feasible because once everything is setup like servers, simulators and all communication channels (Bluetooth adapter) initially, only initial investment is required and later on just maintenance expenses are incurred which would benefit a lot.

**Technical feasibility:**

Technical feasibility centers around existing computer system to what extent it can support the proposed system. This also considers costs incurred to accommodate technical enhancements.
Since Mobile Ticketing can be accessed by any GPRS enabled mobile. It is technically feasible, because most of the mobiles are GPRS enabled. Other reasons are availability of simulators, servers and Bluetooth adapters would prove that this project is technically feasible.

**Behavioral Feasibility:**

Behavioral feasibility considers the advantages of using proposed system over existing system.

- Mobile Ticketing reduces the time spent on reserving tickets,
- This service is available to the user round the clock.

**Resource Feasibility:**

This project has no need of bulk resources. It just requires the development center that has personal computers equipped with hardware and software required to build the system.

### 2.7 Application Interface

Mobile - tickets reservation application will be placed in the mobile device and it can be accessed using an icon named m-tickets. To purchase air tickets, this icon should be selected, after connection has been established this ticketing application will welcome the user to main page to reserve tickets, now user will have options to enter “FROM” and “TO” along with date. Now the system will retrieve the requested information and displays the list of available flights on that day along with different airline services, user can choose the flight and click ok.

Now the system will ask for names and number of seats to be reserved, if available it confirms and asks to enter credit card details along with security code and
gives the confirmation. Authentication is provided in data by encrypting data using public key cryptography using some cryptographic algorithms while coding. Security can be purchased from 3rd party providers like verisign. Figure 2.2 is the application interface (Enquiry phase).

Figure 2.2 Application Model
3. PROJECT REQUIREMENTS, SYSTEM DESIGN, AND IMPLEMENTATION

This project has been designed using Integrated Development Environment (IDE) called Carbide C++ Express version 1.1 with the support of Symbian Software Development Kit (SDK). Java Runtime Environment (JRE) and Perl 5.6 are the supporting tools for Symbian SDK; also Java Development Kit (JDK) is used for writing code for the communication between application and database server (SQL Server 2005). Project is now implemented using Symbian SDK emulator, and it can be implemented on the mobile by the availability of Static IP address from the service provider.

3.1 System Requirements

This project has been implemented on a PC running Windows XP/2000 Operating System with service pack 2. Carbide C++ Integrated Development Environment (IDE) (v 1.1), Nokia Series 60 SDK, Active Perl (5.6), Java Runtime Environment (1.4.2), Java Development Kit (1.4) are required to be installed on the PC. On availability of Static IP, Bluetooth enabled Nokia Series 60 mobile with Bluetooth adapter, Cygwin, Bluesoleil (Bluesoleil is a Windows 2000/XP program that allows communication with Bluetooth enabled devices) are required to implement this application on a mobile phone. [Leigh 2006]

Minimum requirements for running this application:

- **Processor:** 450-Megahertz (MHz) Pentium II-class processor, 600-MHz Pentium III-class processor and higher are recommended.
**Operating System:** Microsoft Windows® Server 2003, Windows XP Professional. Applications can be deployed onto the following systems: Windows Server 2003, Windows XP Professional.

**Memory:** Windows Server 2003: 160 Megabytes (MB) of RAM, Windows XP Professional: 160 MB of RAM.

**Hard Disk:** 500 MB of available space required on system drive, 1.5 gigabytes (GB) of available space required on installation drive.

### 3.2 Use Case Diagrams

Use case diagrams are drawn from the end user point of view. Objectives of use case diagrams are as follows:

- To define the functional and operational requirements of the system by defining a scenario of usage that is agreed upon by the end user and software engineering team.

- To provide a clear and unambiguous description of how the end user and the system interact with each other.

- To provide a basis for validation testing. [Scott 2004] Figure 3.1 shows the use case diagram for this application; firstly mobile user is authenticated for the service and then allowed to enquire and then start reservation. Reservation undergoes two different validations and finally reserves tickets.
3.3 Class Diagrams

Once classes and objects have been identified using the CRC (Class Responsibility Collaborator) model, the analyst begins to focus on the structure of the class model and the resultant hierarchies that arise as classes and subclasses. Using UML notation a variety of class diagrams can be created.

Figure 3.2 shows some classes implemented by some ticketing application. It also shows the class hierarchy of the classes that a standard application uses. All the classes are derived from CBase. CBase has a number of useful features: it initializes all member data to zero, it has a virtual destructor, and it implements support for the Symbian OS cleanup stack. Figure 3.2 shows the class diagram; we can clearly see that all the classes are derived from CBase class. Each level is related to different objects in the programming code. [Series 60 2007]
3.4 State Transition Diagrams

In the context of Object Oriented Systems, two different characterizations of states must be considered.

- The state of each object as a system performs its function.
• The state of the system as observed from the outside as the system performs its functions.

The state of an object takes on both passive and active characteristics. A passive state is simply the current status of all of an object’s attributes. The active state of an object indicates the current status of the object as it undergoes a continuous transformation or processing. [Scott 2004]

A State Machine is a behavior that specifies the sequences of states and objects that goes during its lifetime in response to events, together with its responses to those events.

Actions involved in state transition diagrams are state, event, transition, activity, action. A **State** is a condition or situation during the life of an object during which it satisfies some condition, perform some activity, or wait for some event. An **Event** is the specification of a significant occurrence that has a location in time and space. A **Transition** is a relationship between two states indicating that an object in the first state will perform some actions and enter the second state when specified event occurs and specified conditions are satisfied. An **Activity** is ongoing non atomic execution with in a state machine. An **Action** is an executable atomic computation that results in a change in state of the model or the return of a value.

Graphically, a state is rendered as a rectangle with rounded corners. A transition is rendered as a solid directed line.
Figure 3.3 shows the transition diagram for our application; we can see many transitions states like connect, disconnect, enquire, reserve and confirm are in the diagram.
3.5 Data Flow Diagrams

A data flow diagram is a structure analysis tool that is used for graphical representation of data processes through any organization. The data flow approach emphasis on the logic underlying the system, by using combination of only 4 symbols. It follows a top down approach. A full description of a system actually consists of set of data flow diagrams, which comprises of various levels. And initial over view model is exploded lower level diagrams that show additional feature of the system. Further each process can be broken down into a more detailed DFD. This occurs repeatedly until sufficient details are described.

3.6.1 DFD symbols

**Square**

![Square symbol](image)

It defines a source (originator) or destination of system data.

**Arrow**

![Arrow symbol](image)

It indicates data flow-data in motion. It is a pipeline through which information flows.
**Circle or Bubble**

It represents a process that transforms incoming data flow(s) into outgoing data flow(s).

**Open Rectangle**

It is a data store-data at rest, or a temporary repository of data.

Following are the different levels:

A **level-0** DFD is called a fundamental system model or Context Model that represents the entire software element as single bubble with input and output data directed by incoming and outgoing arrows respectively.

A **level-1** DFD contains five or six bubbles with interconnecting arrows. Each of the processes represented at level one is a sub function of the overall system depicted in the context model.

A **level-2** DFD describes the sub functions of the bubbles indicated in the level-1.

Figures 3.4, 3.5, and 3.6 represents context level level-1 and level-2 data flow diagrams respectively. In figure 3.4 context level can be seen which is also called as level – 0 diagram, it represents entire process using single bubble. Figure 3.5 has many Level 1 diagrams; each represents a sub function of the entire project. Figure 3.6 is the level-2 DFD which describes the sub functions of the bubbles indicated in the level-1.
Figure 3.4 Context Level Diagram for Mobile Ticketing

Figure 3.5 Context Level 1 Diagram for Mobile Ticketing
Figure 3.6 Context Level 1 Diagram for Mobile Ticketing

Figure 3.7 Context Level 1 Diagram for Mobile Ticketing
Figure 3.8 Level-1 Diagrams for Mobile Ticketing

Figure 3.9 Level-2 Diagram for Mobile Ticketing
3.6 Interface Design

The mobile module mainly deals with the interface. How efficiently a user can access details, and book the tickets. Authentication is provided in data by encrypting data using public key cryptography.

Different sub modules in mobile interface are shown in the Figure 3.1

- Connect
- Enquiry
- Reservation
  - Request to reservation
  - Confirmation message
- Exit

Description of each module is as follows:

Connect:

Each SIS (Symbian OS Installer) files consists unique user id. If he wants to access this application all he need’s to do is to send this id to the server. This is done by selecting the connect option. The user cannot perform other operations if he does not select connect. If user selects the connect option, user id is sent to server then server accepts the connection and it sends a connected replay to user. Then user can enter in to enquiry or reservation page.
Figure 3.10 Different options in the application
Enquiry:

In this module the user gives details about his journey i.e. from, to, date and service type as shown in the Figure 3.2. He must fill each and every field otherwise his data is not sent to server. A user will get different services along with time, no of seats.
available and fare for that service after some waiting as shown in the Figure 3.3. If user gives wrong details then he will get invalid data.

Figure 3.12 Seats availability screen with time and flight number
Reservation:

This module contains two parts

1. Request to Reservation
2. Confirm message

![Figure 3.13 Screen to enter no of seats](image-url)
**Request to Reservation:**

In this user gives reservation details like date, from, to, time, service type, and service no, and no of seats as shown in the figure 3.4. According to that user gets another form showing the structure of the seats arrangement in the flight as shown in the figure.

![Figure 3.14 Structure of seats arrangement](image-url)
3.5. These details are sent to the server if entered correctly.

![Screen for purchasing air ticket]

**Figure 3.15 Screen for purchasing air ticket**

The server sends three types of responses

1. If data is valid and seats are available then it sends the service number and available seat numbers.
2. If data is valid and seats are not sufficient then it sends “seats unavailable” and gives how many seats are available.

3. If data is not valid then it sends “invalid data entered”.

Figure 3.16 Confirmation screen after purchase
Confirmation message:

If requested time and seats are available, user will enter into this page. In this user selects seat numbers from the available seats list, and enters credit card type, credit card number, and CVV number as shown in the Figure 3.6. If user provides correct information he will get the confirmation number for his reservation as shown in the figure 3.7. For any invalid data, the user will be asked to enter the appropriate information.

Exit:

After getting confirmation number, user will have an option to exit from the application. By selecting exit server will suspend the connection and sends a connection terminated message.

3.7 Project Development Tools:

The application has been constructed using many tools and software’s. Following are the tools and software’s used in the development of this application.

3.7.1 Carbide.C++ IDE

Carbide.C++ is a software development tool for C++ development on Symbian OS. This product is developed by Nokia family as part of its Carbide development tools family, this software is offered in four flavors called express, developer, professional and OEM (Original Equipment manufacturer). Carbide.C++ Express version 1.1 is the powerful integrated development environment (IDE) for developing such mobile application. It is very productive tool for rapid building of mobile applications and implementation.
3.7.2 Symbian SDK

To do Symbian OS C++ application, Carbide.C++ and Symbian OS SDK (Software Development Kit) are required which contains the windows based emulator, documentation, libraries and header files for development. SDK can be obtained from Symbian website.

3.7.3 JRE 1.4

Java Runtime Environment (JRE) is a software package from Sun Microsystems which allows computer to run Java applications. JRE works as supporting tool for Symbian SDK emulator.

3.7.4 ActiveState ActivePerl 5.6

This tool works as supporting tool for Symbian SDK, because Symbian OS requires some of the perl scripting for running of emulator using Symbian SDK.

3.7.5 Java Development Kit 1.4

Java development kit is the Sun Microsystems product containing java compiler, archiver, document generator and debugger. It is used to compile and debug socket and other programs written in java. This Java carries data from interface and stores or checks with the master database. Java machine should be always running while the application is running, because java works as mediator between application and the database for validations and retrieving required data.

3.7.6 SQL Server 2005

SQL Server is a relational database management system produced my Microsoft. This DBMS package is used to store and load information from user and airline service, and also for executing various queries while application is running.
3.8 Symbian C++ and Java code Implementation:

Symbian project mainly consists of six important folders:

1) **Source**: CPP files are placed in source folder.

2) **Include**: Header files are placed in include folder.

3) **Group**: Mak Make Project (MMP) file and bld files are placed in group.

4) **Data**: Resource Source Script (RSS) file is placed in data folder, which is used for mentioning menus and components.

5) **Gfx**: All image files are placed in gfx folder.

6) **Sis**: System Installation System (SIS) file and package (.pkg) files are placed in sis folder.

In Symbian, program starts its execution at E32Main() as cpp and c starts with main() function as its entry point.

**Example Code:**

```c
GLDEF_C TInt E32Main ()
{
    return EikStart::RunApplication (NewApplication);
}
```

For all kind of Symbian applications a unique ID (UID) is required for its identification. Following is as example function for creating UID:

```c
TUid CTicketingApplication::AppDllUid () const
{
    // Return the UID for the Ticketing application
    return KUidTicketingApp;
}
```
Resource Source Script (RSS) is another important file in which menu items are defined for soft key handling.

Example Code for .RSS:

```
ESOURCE MENU_BAR r_mticket_view1_menubar1
{
    titles = {
        MENU_TITLE
        {
            menu-pane = r_mticket_view1_menu;
        }
    }
};

RESOURCE MENU_PANE r_mticket_view1_menu
{
    items ={
        MENU_ITEM
        {
            command=EEnq;
            txt="Enquire";
        },
        MENU_ITEM
        {
            
        }
    }

```
Java programs in the backend are coded for socket communication and database communication. Example code for socket programming is shown below:

```java
import java.io.*;
import java.net.*;
import java.util.*;
import java.awt.event.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.*;
```
public class main1{
    DatagramSocket ds;
    DatagramPacket dp;
    String resdet;
    String servdet;
    byte bb[];
    String msg;
    InetAddress ins;
    public main1()
    {
        try{
            ds=new DatagramSocket(5553);
            ins=InetAddress.getByName("192.168.1.3");
        }
        catch(Exception e)
        {
            //System.out.println("in main1 constructor "+ e);
        }
    }

    Example database connectivity string:
    
    Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
    con = DriverManager.getConnection("jdbc:odbc:kiran","sa","kiran");
    st=con.createStatement();
3.9 Security Measures

Security can be provided using a protocol called SSL (Secure Sockets Layer), developed by Netscape for transmitting secret data via internet. SSL uses cryptographic system which uses two keys called public and private to encrypt and decrypt data. Security services can be obtained from third parties like GuardianEdge Smartphone Security, Verizon Security Service, and McAfee Mobile Client Security. These services will provide top rated security between client and master databases for sensitive data like credit card info.

3.10 Real time implementation of the application

This application can be executed in the mobile with the availability of static IP address from the mobile service company. To communicate with the servers via Bluetooth, mobile should have an IP address which will be recognized by the computer for the communication. Communication is done via GPRS (General Packet Radio Service), WiFi (Wireless Fidelity), and WLAN (Wireless Local Area Network), in various situations.

3.10.1 Creation of “.sis file” for real time implementation

For the application to be implemented on the mobile phone, Carbide.C++ IDE produces “.sis” file which can be placed as executable file in the mobile. Sis file can be obtained by just compiling the code for phone instead of emulator, and then by building the project will produce “.sis” file. This “.sis” file is created in the sis folder, which can be used for the mobile phone implementation.
4. TESTING AND EVALUATION

Testing is a process, which reveals errors in the program. It is the major quality measure employed during software development. During testing, the program is executed with a set of test cases and the output of the program for the test cases is evaluated to determine if the program is performing as it is expected to perform. [Ron 2005]

In order to make sure that the system does not have errors, the different levels of testing strategies applied at differing phases of software development, those strategies are as follows:

4.1 Unit Testing

Unit Testing is done on individual modules as soon as they are completed and ready to execute. It is confined only to the designer's requirements.

Each module can be tested using the following two strategies:

4.1.1 Black Box Testing

In this strategy some test cases are generated as input conditions that fully execute all functional requirements for the program. This testing has been used to find errors in the following categories:

a) Incorrect or missing functions
b) Interface errors
c) Errors in data structure or external database access
d) Performance errors
e) Initialization and termination errors.

In this testing only the output is checked for correctness. The logical flow of the data is not checked.
4.1.2 White Box Testing

In this the test cases are generated on the logic of each module by drawing flow graphs of that module and logical decisions are tested on all the cases.

It has been used to generate the test cases in the following cases:

a) Guarantee that all independent paths have been executed.

b) Execute all logical decisions on their true and false sides.

c) Execute all loops at their boundaries and within their operational bounds.

d) Execute internal data structures to ensure their validity.

Tests are designed to answer the following questions

- How is the function validity tested?
- What classes of input will make good test cases?
- Is the system particularly sensitive to certain input values?
- How are the boundaries of the data class isolated?
- What data rules and data volume can the system tolerate?
- What effects will specify combinations of the data have on the system.

4.2 Test Cases

Test cases are written for all the phases in the project to test different module functionalities. Each test case will have different outputs based on inputs; errors in the module can be identified to make it perfect. Following are the test cases for different modules of the project:
1. **Connect:** In connect module, input is selecting the connection option. Output can be connected, connection failed (if server busy, any network busy), and license expired. Changes can be made based on the output.

2. **Enquiry:** In enquiry, inputs are user gives correct input; user gives wrong input, and user missing some fields. Output can be response for enquiry, invalid data entered, service not available, and fill all the fields’ message.

3. **Reservation:** In reservation, inputs are user gives correct input, user gives wrong input, and user missing some fields. Output can be response for reservation, invalid data entered, and fill all the fields.

4. **Confirmation:** In confirmation, inputs are user gives correct input, user gives wrong input, and user missing some fields. Output can be confirmation number, invalid data entered, invalid bank details, and fill all the fields.

5. **Exit:** In exit, input is selecting the exit option. Output can be error, thanks for using this service.

Once the project is designed and implemented, all the modules are tested using different test cases and the results will be included in the final project report.
5. FUTURE WORK

This project has some scope for enhancement in the future. The following are some ideas.

1) With the change of interface and idea, many applications can be designed.

2) This application can be extended to book railway and buses tickets from mobiles, and also can enhance the same architecture with upcoming architectures like UMTS (Universal Mobile Telecommunications System) and EDGE (Enhanced Data rates for GSM Evolution).
6. CONCLUSION

The design of this project is aimed at providing airline ticket purchase facility to the mobile users securely, and to design a user friendly application which would be easy to access. This mobile ticketing application would save time and add convenience to the users. This project is designed for Symbian Operating System based mobiles which share major part of the mobile industry; therefore this project can be implemented and extended widely.
BIBLIOGRAPHY AND REFERENCES


