DESIGN AND IMPLEMENTATION OF AN SMS BASED ROAD TRAFFIC OFFENCE TRACKER AND PROFILING SYSTEM

BY

MOHAMMED, Auwalu Magaji
M.Sc/SCIEN/02125/2008-2009

A THESIS SUBMITTED
TO
POSTGRADUATE SCHOOL
AHMADU BELLO UNIVERSITY ZARIA
NIGERIA

IN PARTIAL FULFILMENT FOR THE AWARD OF
MASTER OF SCIENCE DEGREE IN
COMPUTER SCIENCE

DEPARTMENT OF MATHEMATICS
AHMADU BELLO UNIVERSITY, ZARIA
NIGERIA

MAY 2015

DECLARATION

I declared that the work in this dessertation ‘Design and Implementation of an SMS Based Road Traffic Offence Tracker and Profiler’ has been carried out by me, in the Department of Mathematics, under the Supervision of Professor S B Junaidu and Professor S O Adewale.

The information derived from the literature has been duly acknowledged in the text and list of references provided. No part of this thesis was previously presented for another degree or diploma to the best of my knowledge at any University.

Auwalu Magaji Mohammed	_____	_____
Name of student	Signature	Date

CERTIFICATION

This dessertation entitled “Design and Implementation of an SMS Based Road Traffic Offence Tracker and Profiler” by Auwalu Magaji Mohammed meets the regulations governing the award of the degree of Master of Science in computer of Ahmadu Bello University, Zaria and is approved for its contribution to knowledge and literary presentation.

_____ Signature:_____

Chairman, Supervisory Committee

Prof. S.B. Junaid

Date: _____

_____ Signature: _____

Member, Supervisory Committee

Prof. O.S Adewale

Date: _____

_____ Signature: _____

Head of Department

Dr. S.B. Sani

Date: _____

_____ Signature_____

Dean, School of Postgraduate Studies

Prof. A.Z Hassan

Date: _____

ACKNOWLEDGEMENT

Glory be to Allah, the Lord of the world, the Beneficent and the Merciful Who made it possible for me to reach this far in this Programme.

My sincere appreciations and gratitude to my Research Supervisors; Professor S. B. Junaidu for the wonderful guidance and his patience with my inattentiveness during the research work and Professor S. O. Adewale for his professional advice and guidance.

I wish to express my sincere gratitude to the FRSC Management for giving the opportunity to pursue this programme and to Corps Marshal and Chief Executive for his magnanimity. My appreciation also goes to my Head of Department, Officer AA Abu (DCM), Deputy Corps Marshal, Adminstration and Human Resource Department, FRSC.

Equally, my appreciation to Malam Abdullahi Mohammed, Malam Barroon Ismail, Malam Dishin Salihu, Malam Mustapha Bagiwa, Malam Abdullahi Umar for their contribution towards the success of the work and all the staff in Mathematics Department, ABU, Zaria.

Finally, I would like to thank and salute my lovely wife, Suwaiba Mohammed Auwal for encouraging me to further my education and her support to me during the programme. My children, Aisha, Salma, I thank you all and my late son Mohammed Mohammed Auwal may he rest in Jannatul Firdaus, Ameen. To my friends and colleagues, Sabitu Abdu and Nasiru Mohammed (Harder), I say thank you for your support, encouragement, and contribution toward the success of this work. May Allah bless and reward you abundantly, ameen.

ABSTRACT

Federal Road Safety Commission Act, Cap 141, Law of the Federation of Nigeria 2007 (FRSC Act, Cap 141, LFN 2007.) has mandated FRSC Operatives to seize, suspend or disqualify from driving on Nigeria's roads, any recalcitrant Road Traffic Offenders (RTO), who is adjudged to have accumulated 25 or more penalty points from the road traffic offenses. However, due to lack of access to road traffic information, the mandate has never been implemented to date. In this dissertation, we set out to re-engineer FRSC Patrol Operations. Our objective is to provide the FRSC Field Operatives with ubiquitous access to road traffic offenders' information from the patrol locations. We analyzed the old FRSC record management system; we re-structured the data on road traffic and designed 'An SMS Based Road Traffic Offense Tracker and Profiler, called (Tracker), which solved the problem of access road traffic offense information for the FRSC Operatives. Global System for Mobility (GSM), Short Message Service (SMS), Java, Android, Extensible Mark-up Language (XML), PHP and other relevant technologies were used to implement the Tracker. The Tracker had been successfully implemented and tested with knocked-off RTO data from Federal Capital Territory, Abuja (FCT, Abuja).

It is found worthy for use by FRSC, as it is working correctly and capable of receiving, processing and replying users' requests, not only from Nigeria, but from all over the world. The Tracker offered the FRSC Operatives real time ability for tracking and profiling the road traffic offenses, vehicle and vehicle owners' and implementation of the RTO Penalty Points for the disqualification or suspension of recalcitrant motorist.

TABLE OF CONTENTS

LIST OF TABLES

TABLE OF FIGURES

ACCRONYM

API	Application Program Interface
CDC	Connected Device Configuration
CDMA	Code Division Multiple Access
CLDC	Connected Limited Device Configuration
CUG	Closed User Group
CVM	Compact Virtual Machine
DBMS	Database Management System
DGD	Dangerous Driving
ETSI	European Telecommunication Standard Institute
FRSC	Federal Road Safety Commission
FRSC-MVA01	Vehicle Registration Form
FRSC-MVA04	Vehicle Registration Book
FRSC-MVA04	Fresh Driving License Application Form
FRSC-MVA04	Renewal Driving License Application Form
FRSC-OPS06	Offenders Register
FRSC-OPS06	Payment Register
FRSC-OPS06	Multiple Offenders Register
FRSC-OPS06	Vehicle Impounded Register
FRSC-OPS06	Wanted Offenders Register
GPRS	General Package Radio Service
GSM	Global System of Mobile communication
HSDPA	High Speed Downlink Packet Access
HSUPA	High Speed Uplink Packet Access
IP	Internet Protocol
J2EE	
J2ME	Java 2 Micro Edition
J2SE	Java 2 Standard Edition
JAD	Java Application Descriptor
JAR	Java Archive
JDK	Java Development kits
JNI	Java Native Interface
JVM	Java Virtual Machine
KVM	Kilobytes Virtual Machine
MIDP	Mobile Information Device Profile

OSV	Over Speed Violation
5G	Fifth Generation Telecommunication Network
PDA	Pocket Data Assistant
RAD	Rapid Application Development
RTC	Road Traffic Crash
RTO	Road Traffic Offender, Road Traffic Offence
RTR	Road Traffic Rules and Regulation
SMS	Short Message Service
SMSC	Short Message Service Centers
SMSD	Short Message Service Daemon
SQL	Structured Query Language
TDMA	Time Division Multiple Access
TomdApplication	Traffic Offenders Management Application
TomDatabase	Traffic Offenders Management Database
UML	Universal Modeling Language
WAP	Wireless Application Protocol
WML	Wireless Markup Language
wwww	Wireless World Wide Web
1.5G	First Derivatives of the First Generation
1.75G	Second Derivative of First Generation Telecommunication Network
1G	First Generation Telecommunication Network
2.5G	Second Derivative of Second Generation Telecommunication Network
2G	Second Generation Telecommunication Network
3.5G	First Derivatives of the Third Generation Telecommunication Network
3.75G	Second Derivative of Third Generation Telecommunication Network
3G	Third General of Telecommunication Network
4G	Fourth Generation Telecommunication Network
XML	Extensible Mark-up Language

CHAPTER ONE: INTRODUCTION

1.0 BACKGROUND OF THE STUDY

Road traffic administration is a complex and crucial issue as it involves managing people and vehicles as well as movement of people and properties on the road. Vehicular movement most often causes road traffic crashes that usually bring about loss of human and material resources. Road traffic management requires information on road users' vehicles and their owners. There is also need for a good road traffic information management system that is capable of processing and disseminating of the information as fast as possible to the road traffic law enforcement agents in order to equip and empower them to discharge their duties with increased diligence.

In Nigeria, law enforcement agents, particularly road traffic enforcement agents lack access to information on road users especially information pertaining to road traffic offenders, vehicle and drivers when carrying out their statutory functions from the duty posts. Nigerian road users' are aware of these inadequacies especially if one look at the act of lawlessness by these road users. Disobedience to Road Traffic Rules and Regulations (RTRs) has reached an unprecedented level of unacceptability. The lack of information has been a hindrance which affects the effectiveness of law enforcement agents, and is therefore no more acceptable if we consider the level of advancement in the area of information management and technology especially in the area of mobile computing. FRSC (2010).

Since the advent of mobile computing coupled with the improvement of mobile small devices processing power, memory capacity and development of mobile wireless telecommunication networks, access and dissemination of information becomes ubiquitous.

Mobile devices are now being used to run complex and dynamic applications that were only run on computer (PC) in the past. These mobile devices are now capable and rapidly replacing

computer systems for running business application both dynamic and standalone such as surfing of internet, remote monitoring applications and many more. Several communication technologies or protocols such as WAP, Bluetooth, SMS, and GPRS are now being used in conjunction with mobile devices to develop and run applications.

Simplicity and universality of the Short Message Services (SMS) makes it popular for development of applications for mobile device by programmers. SMS is a communication protocol which allows exchange of short textual message between mobile device users. It is one of the most popular services provided by GSM networks.

Simplicity, portability, ubiquous access to information, mobility, cheapness and network availability are some of the characteristics and advantages offered for running public and private business applications on mobile device than on computers. SMS based mobile device applications offer businesses the advantages of mobility and ubiquitous access to information at cost effective price.

1.1 PROBLEM STATEMENT

FRSC Act Cap 141, Law of the Federation of Nigeria 2007, mandated Federal Road Safety Corps to reduce Road Traffic Crashes (RTC) to the barest minimum in Nigeria. The Commission in its early years of inception actualized this mandate. However, the problems of RTCs resurfaced again, and unlike before, the increase of RTCs and the attendant loss of human and materials resource from it, is so unprecedented. In Nigeria today, Road Traffic Administration is so patchy and ineffective because;

a. Data on Road Traffic Offences (RTOs) are scattered across the country with no standard means of storage, access, protection and backup.

b. Where available, the data are not harmonized and are processed manually.
c. No connectivity between data in various departments, particularly between Operations and Motor Vehicle Administration Departments.
d. FRSC Patrol Operatives cannot access RTOs information from patrol locations.

These problems present a serious challenge to the capacity of FRSC as a leading agency in road traffic management in Nigeria. Among the noticeable effects of the above problems are;

a. Incessant disobedience to RTRs and disregards to constituted authority by the Nigerian Road users which often results to RTCs that causes loss of human and materials resource.
b. Forgery of Vehicle Plate Number, Driving License and other Vehicle Registration documents which cause loss of revenue to both Federal and State Governments.
c. Inability to trace and apprehend run away Road Traffic Laws Violators (RTVs)
d. Lack of effective communication facilities to exchange information among Field Operatives.

1.2 RESEARCH MOTIVATION

We are motivated to undertake this research work because the Researcher participated in a committee work constituted by FRSC Management that studied and found out the causes of recent RTCs increase in Nigeria.

Our aim is to provide solution to some of the major issues and problems hindering RTCs prevention and RTOs identification which is due to lack of data and on-the-spot (ubiquitous) access to information on RTOs and other related Road Traffic issues as identified in the Committee's Report.

1.3 RESEARCH PROBLEM

Can a universal SMS-based Road Traffic Offences Tracking system be built to empower FRSC Operatives in order to improve RTA?

1.4 AIM AND OBJECTIVES OF THE RESEARCH

The aim of this Research Work is to re-engineer the FRSC Patrol Operations by introducing modern concept of Information Management in the area of RTRs enforcement in order to help FRSC actualize its mandate as enshrined in the FRSC Act, Cap 141, LFN 2007.

The objectives of the research are as follows;

a. Come up with a comprehensive organized data on RTOs, Vehicle, Vehicle Owners, Driving Licensing and other related information, carefully formatted for suitable manipulation by the proposed system.

b. design a system capable of manipulating the data through SMS service of GSM network and mobile small devices (Cell Phone) in order to improve the capacity of FRSC Operatives and empower them with the ability to;

    i. Enable computations of offence penalty point from the Operatives Patrol locations when a RTO is adjudged to accumulate sufficient number of point to warrant arrest and prosecution for disqualification from driving.
    ii. Enable the tracking and apprehending road traffic rules violators (RTVs) from the Operatives Patrol locations.
    iii. Identify and ascertains the genuineness of Vehicle Plate Number, Driving License and other Vehicle Registration document from the Operatives Patrol locations.
    iv. Provide lead information for identification of RTCs victims where such victims'

body are mutilated beyond physical recognition.

1.5 SIGNIFICANCE OF THE RESEARCH PROJECT

a. To inculcate the notion of no escaping FRSC arrest when one violates RTRs into the Nigerian motorist mind which would induce discipline on the road user.
b. To discourage forgery and of Vehicle Plate Numbers, Drivers License and other documents.
c. To simplify contact with RTCs victims' families and relations when there is emergency.
d. Provide a platform for the actualization of implementation of penalty points as enshrined in the FRSC Act, Cap 141 LFN, 2007.

1.6 RESEARCH METHODOLOGY

In order to achieve our objectives, we followed the following procedures;

System Analysis Stage: here, we carried out a detailed study and analysis of the existing FRSC system in order to identify how the functional process work, it problems, relevant documents that are of interest to this research work, the sources of data and how the data are inter related with one another.

Conceptualization design stage: we used Universal Modelling Language (UML) to develop the conceptual model of our system. We developed various proto-type block diagram of the system and then choose the one that suited our purpose and continue to refine it in order to perfect it. Detailed of the system components and parts were determined at this stage.

ER-diagram Design Stage: the design schematic diagram of the system and implementation of the database also took place at this stage.

User Interface Design: user interface is how interaction would take place between the user and

the device. This stage is where we designed the interface of the proposed system. We determined the mode of interaction between the user and the system as well as selection of communication protocol and transmission media we would use to exchange information between the system components and users.

Backend Application Design and Configurations: this is multi activity stage where we designed the system's backend application and then configured the other hardware devices with the PC Server. The backend application is the message processing engine which is implemented with PHP, SQL and Bash scripts.

Testing and Evaluation: At this stage, we coupled the various components of the system into one and then conducted tests and evaluate the performance of the overall system.

1.7 RESEARCH TOOLS

The following are the tools needed for the successfully design of the proposed system.

1.7.1 HARDWARE TOOLS

a. Personal Computer

b. Mobile Cell Phone

c. GSM Mode

1.7.2 SOFTWARE TOOL

a. Operating System: Ubuntu 12.04 LTS Desktop for the PC Server

b. Messaging Gateway: Gammu

c. Database Management System: Mysql and Mysql Workbench 6.0

d. J2ME Development Environment: NetBeans IDE 7.1.2 and JCreator

e. Android Application Development Kit (Eclipse and Android SDK Manager).

1.7.3 PROGRAMMING AND SCRIPTING LANGUAGE

a. Java Language and Java 2 Micro Edition (J2ME)

b. XML, PHP, SQL and Bash Scripting languages.

1.8 THESIS OUTLINE

This write up comprises of five chapters. Chapter one gives the introduction to the research work. it looks into the relevance of the research by giving detailed overview of the research area in section 1.1, the research problems in section 1.2, section 1.3 is the motivation behind the research work, aims and objectives in section 1.5 and 1.6 as well as the significance of the research work in section 1.7. Finally, the research methodology and the thesis outline were discussed in section 1.9 and 1.10 respectively.

Chapter two looks into the relevant literature related to the research. It also discusses the technologies used in developing the system.

Chapter three is on design and implementation of the system. Characteristics of the system are discussed in section 3.1, the overall system architecture is given in section 3.2. The detail design and implementation of the SMS messaging and processing applications were discussed in section 3.3 and 3.4, respectively.

Chapter four focused on the functional structures and processes of the whole system. Equally deliberated in section 4.3, are the various tests and evaluations conducted after the system was implemented.

Chapter five presents the summary, recommendations, future work and conclusion of the research.

Phase	Activity	Jan				Feb				Mar				Apr				May				Jun				Jul				Aug			
		W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4
1	Database Design & Implementation																																
2	User Interface Design & Implementation																																
3	Gammu Configuration																																
4	Backend Application Design & implementation																																
5	Trigger Design and Implementation																																
6	Testing and Evaluation																																
7	Project Report																																
8	Research Defence (Internal)																																

1.1.

Table 1.1: Research Plan

CHAPTER TWO: LITERATURE REVIEW

2.0 REVIEW OF RELATED WORK

Significant research works have been done in the area of using Mobile cell phone and SMS as communication protocol for running complex dynamic and stand alone applications. Most of the applications are for controlling and monitoring of systems, equipment and devices, while some are for managing public and private businesses.

Control, monitoring and building automation systems with cell phone, using GSM network and SMS service is an area that is attracting researchers' interest. The compatibility of mobile cell phone and the availability and the wider coverage of the GSM network lead to the rapid expansion. For example;

Dey et al (2000) described a system "Context-aware System for Supporting Reminders" that allows users to defined a complex condition under which a reminder will be generated at a time interval (e.g Time is "8:00" and Location is "Office"). The system uses GPS to determine user's location and send a reminder as an SMS message.

Werff *et al* (2004) described a system for controlling home appliances automatically using GSM/GPRS modem. The system consists of three main components, the SMS messaging Application implemented with J2ME, for sending SMS message from User Cell Phone, the control system comprising of a GSM/GPRS modem, Atmel Micro Controller and Computer in conjunction with AT commands textual processing software. The software is responsible for interfacing the modem with computer as well as processing SMS message.

Ciubotarus, *et al* (2006) presented an approach for implementation of control and monitoring system base on SMS. The system composed of three main modules. The sensing module,

represented by digital input lines. This may be a networked server that communicates its internal status to the monitoring system via serial interface or ethernet controller. The processing module which is a micor-controller, interfacing the sensing unit to detect events in the process that is monitored by the system. And lastly, the communication module that allows the processing unit to send notification in the form of SMS message to the system administrator in case of breach or fire alarm. The communication system can be implemented with either GPRS modem or commercial cell phone interfaced with the server serial RS-232 port.

Alomary, (2007) presented discuss on a system for controlling remote devices through the Internet, SMS and Telephone Networks. The paper described the 3 main types of GSM7BIT standard for availing SMS services on GSM network. The system uses ActiveXperts, SMS and Toolkit protocols. Decoding and switching circuits are used for controlling devices and the application was developed with e-.net Compact Framework platform.

Jawarkar, *et al* (2008) described a method of remote monitoring through mobile phone involving the use of voice as commands. The voice commands are generated and sent in the form of text SMS to the control system and then the micro-controller, on the basis of SMS takes a decision of a particular task. This system allows users ubiquitous control and monitoring of device and event from their location.

Murthy, (2008) explored the use of SMS services of the GSM network for primary health-care management for the rural population. The system involved the use of SMS and cell phone technology for information management, transactional exchange as well as patient record management and personal communication between the health-care providers and the patients

Kana, *et al* (2009) deployed an Information Technology for Vehicle Control and SMS vehicle tracking and vehicle owners' profiling system. The system had three main components; the user

interface made up of SMS Messaging Application for composing sms message and a web based application for sending requests to the system. The application tier is responsible for processing user's request. It translating the request into relevant database queries and executing the queries. Service layer which is responsible for converting the data retrieved from the database into a suitable file format for presentation to the user. The system is concerned with tracking vehicle owners' profile when there is report of vehicle theft. It also allows vehicle owner's to send an alert to police of a given area when a vehicle is stolen or there is vehicle robbery, by sending a simple SMS message via cell a phone or by filling a web based form.

Khiyal, *et al* (2009) presented an approach for controlling home appliances and automation system using an SMS servce of GSM network. The system comprises of two sub system, appliances control sub system which enable users to remotely control appliances like TV, AC by sending an SMS via a GSM modem to change the condition of the appliances e.g. on or off. The automation sub system is a monitoring system that detect intrusion to a house by sending an SMS signal to the house owner if an intrusion is detected by a sensor circuit. The sensor circuit is also connected to the GSM modem.

Ratnesh, *et al* (2009), also describe a system of controlling home appliances through SMS. In this approach, the appliances are connected to a parallel port of a pc computer via GSM modem. A database containing the records of the appliance condition is stored in the pc computer. When an SMS message is send from the user cellular phone, the controlling process of the desired appliance is executed, matching the message contents with that of the database and changes the status of the aplliances. The system generate an acknowledgement for the sucessful execution or otherwise and send back reply to the user.

Roth (2009), described a system that deals with raw location of data specific positioning system

like GPS and performs further processing to get an appropriate semantic position of users. It is a client server application that comprises of e client mapping and location based servers, the positioning system and a mobile node. The mobile node host processing application which was developed in java. The server side consists of the position system that include mapping and location based servers and the location based application. An XML file configures the server process during start up and the spatial engine hosted in the location server is responsible for the processing of spatial information to decide if a location resides in an area or not. Mapping server gets its knowledge about the mapping of the positioning system through drivers plugged to the servers. Positions are communicated via SMS message.

Mohammed *et al* (2010) A GSM Based Electrical Appliances Control System that allowed users to control electrical appliances while away from home. Their system also comprises of SMS messaging, message processing system and control circuit. The SMS messaging application is tailored made software residing in the Users' Cell Phone, implemented with Java, C, C++ or any other programming language. Message processing system is made up of textual AT Commands based files that establishes connection between the GSM/GPRS modem and Server as well as fetching and passing messaging from the Modem's Inbox and Outbox. The fetched message is used to control electrical circuit. The control circuit is made from Micro Controller Chip and Relay. Message fetched from processing system is feed into the Chip where it is converted into electrical pulse or voltage. The voltage triggers the relay which turns ON or OFF the required electrical appliance. Confirmation SMS message is sent back to user by the processing system.

Barroon, (2011) developed a remote system of monitoring and management of servers from Mobile Devices. The system uses SMS service of the GSM Telephony Services to control computer servers remotely. The system is made up of a frontend application which is installed in

a mobile cell phone for composing and sending SMS messages, the backend that comprises of two computers, the server and a monitoring computer. The monitoring computer has a mobile cell phone attached to it, a database for storing received messages from users' and reply messages to the users' as well as a message processing programs. The message processing programs are series of batch scripts programs. Monitoring and control of the computer server are done by executing the batch scripts programs which carries out the automation process of checking the status of the server, the services on the server, as well as handling the receive and sending of messages from and to the users'.

An SMS Based Road Traffic Offense Tracker and Profiler is built around handling problems of road traffic laws violations in Nigeria. It comprises of message composing and message processing applications. Our system is concerned about how an FRSC Operative can determine the number of times a motorist was arrested for traffic offense? How to determine whether he is a Wanted, runner away RTO or not and how compute motorist offense penalty points, profile license owner and owner's profile from their patrol location

2.1 REVIEW OF RELATED TECHNOLOGY

In this section, we give the review of the technologies that would be use in this research work. Some of these technologies are new, therefore they need to be clearly explained.

2.2.1 WIRELESS TELECOMMUNICATION TECHNOLOGY (GSM)

According to Bhalla (2010), Mobile Wireless Telecommunication Networks Technology was stated in the early 1970s. Over the decades that follow, it has experiences great revolution, changing from generation to generation. Each generation is denoted by capital letter 'G' to indicate modification or improvement over its predecessor generation. These modifications are

also denoted by letters or acronyms like CDMA, GSM, HSUPA, HSDPA, GPRS, EDGE, and CDMA 2000 and so many more, over various generations and their corresponding derivatives like 1G, 2G, 3G, and 1.5G, 2.5G, 3.5G, 3.75G etc.

1G: is the first generation of wireless telecommunication network wchic is condidered to be the first analog cellular telecommunication system that provides the facilities for making voice calls and sending text masseges. The main disadvantage of this network was that no cross operators roaming facilities were provided. Contact is allowed within the premises of particular nation and also there was no data transmission services available.

2g: repressents the second generation of telecommunication network system that appeared in the early 1990s. 2G network provided an enhanced voice services in digital and supported limite data transfer services and roaming facilities than 1G network. The data transfer speed in 2G network is between 9.6kbpe to 19.2kbps. this let to the exponential growth in terms of both subscribers and services. The 2G network is also refered to as Global System for Mobility (GSM). It extended the functionalities of 1G network and provided new additional ones in terms of

a. Less power consumption
b. Increase in sound quality
c. Introduces SMS services
d. Enhance privacy and fraud reduction among others.

In between the 2G network and the next generation (3G), a mid generation known as **2.5G** network was introduced. It was introduced to offer higher data transfer rate, sort of busting the capacity of the 2G network, hence 2.5G provided data rate of upto 144kbps and ability to browse website from low resource devices like Cell Phones, PDAs and so on. Among the notable 2.5G

technologies are GPRS, EDGE, CDMA2000 e.t.c

Another enhancement to the 2G network is also known as the 2.75G, which is the second derivative of the 2G network that increase the capacity of the 2.5G in terms of data transfer rate of upto 180kbps and ability to download stream video and MP3 files with the low resource devices.

3G networks repressents the third generation of telecommunication networks that spercede 2G networks and its derivatives by overcoming the limitations of the previous networks in terms of functionalities and data rate. 3G networks services includes Wide-Area Wireless Telephony Video Calls and Conferencing, Broadband Wireless data transfer services (in a wireless environment) among others. The data transfer rate of about 14.4mbps on uplink and 5.8mbps on downlink.

The 3.5G networks was the first derivatives of the 3G networks known as High Speed Downlink Packet Access (HSDPA) which is a mobile telephony protocol that provides higher data transmission rate of upto 8-10mbps on the uplink and 20mbps on downlink over 5MHZbandwith frequency.

3.75G networks is an improvement of the 3.5G networks which is refered to as High Speed Uplink Packet Access (HSUPA). It is the second derivatives of 3G networks that provides an enhanced advance person-to-person data applications capacity with higher and symetric data rates.

The fourth among the generation of wireless telecommunication technologies that provides an all IP packet swithing network, mobile ultra broadband (in Giga Bytes speeds), access and multi-carrier transmission. 4G networks provides the platforms for the realization of implementing Wireless World Wide Web (wwww), an iterconnection of wireless devices (low resource devices)

to share resources just like the current interconnection of computers in the www. The data transmission rates of atleast 100mbps at peak rate in full mobility wide area coverage and 1gbps in low mobility local area coverage is supported in 4G networks

The 5th generation of the Wireless telecommunication Networks (5G), under research, expected to be finalize very soon and it would provide more data rates and functionalities for the full consolidating of the of the wwww from 4G networks and revoltionize the wireless networks. Bhalla (2010).

2.2.2 JAVA 2 MICRO EDITION (J2ME)

Developer's Guide described Java 2 micro edition as a sub set of java platform specifically for developing applications for a category of device that are resource constrains in nature such as mobile phones, personal digital assistant (PDA) e.t.c

J2ME is the most lightweight among the three java platforms because of the categories of devices it is intended for, which have limited resources. The functionalities and specifications in J2SE and J2EE were greatly reduct. To understand the function of these layers, one has understand the categories of the low resource devices since differences exists between them in terms of memory, processing power, screen, persistence storage e.t.c. In order to defined a single technology that suit all of them.for the J2ME to work as better as possible, the three layers are defined as follows:

a. Configuration Layer
b. Profile Layer
c. Optional Package Layer.

According to Developers' Guide, a configuration layer is defined as a minimum java platform for a family of devices. It is a specification aimed at devices with similar requirements or

resources such as memory size, processing power and so on.

In J2ME. Configuration layer is the first layer directly on top of the devices operating system (OS) and java vitual machine (JVM). Therefore, configuration can be said to be a set of libraries for making programs (applications) and they are configured into two according to the grouping of the low resource devices known as

a. Connected Limited Device Configuration (CLDC)
b. Connected Device Configuration (CDC)

2.2.2.1 Connected Limited Device Configuration (CLDC)

This connfiguration is concerned with the development of applications for use on the most low reource devices like cell phones, PDAs, Pagers and so on. However, on order for the CLDC to fits into this devices, the standard APIs in J2SE and J2EE were abundon e.g. the 40

classes in the java.utility.packages and interface were reduced to only 10 classes in the J2ME. The result is that CLDC does not have the same functionalities like the those of J2SEand J2EE such as the UI classes e.t.c. CLDC configurations are of different versions, each new versions provides additional functionalities than the previous one. Example, CLDC version 1.0 is used in devices CLDC version 1.1 provided more functionalities and support than the former by adding new features like floating, double point type and weaker reference supports and so on. Also, the version 2.0 provided more improved functions than the version 1.1 and so on.

It has 128KB to 512KB memory, 16bits to 32bits processor and more classes are provide like Java.lang, Java.io, Java.Util, Java.microedition.io classes and many more.

2.2.2.2 Connected Device Configuration (CDC)

As described by Developers' Guide, Connected Device Configuartion (CDC) is larger than the CLDC, best intended for the development of applications for use in devices with more reources

that those defined in CLDC configuration. Devices like television sets, home appliances (DVD, CD, RADIO), Automatic Navigation Systems (GPRS), Routers, Point-of Sale Terminals work with CDC. Compact Vitual Machine (CVM) and to some extend compatible with J2SE and reduced set of JDK-1.3 libraries and allow the use of java native interface (JNI). CDC configuation is used for devices with atleast 2MB or more memory, CVM (Compact Vitual Machine), Use 32bits processor, limited wireless bandwith support and so on. Among the classes supported are; Java.io, Java.lang, Java.security, Java.util, Java.microedition etc.

2.2.2.3 PROFILE

A profile is an extension of the configuration that addresses the specific demands of a device family. It is defined as a collection of java technology (APIs) that supplements a configuration to provides extra capability for a specific device or group of devices. Profile layer is placed on top of the configuration layer, its determines applications field, its life cycle and addition of functionalities needed to builds user interfaces for applications both for CLDC and CDC. Many profiles exists, but the most notable ones are; Mobile Information Device Profiles (MIDP), Information Device Profiles, Digital Set-top Box profiles e.t.c.

2.2.2.4 Mobile Information Device Profile (MIDP)

MIDP profiles are used for devices described in the CLDC configurations, it reside on top of CLDC, it added features such as user interface, network and persistence data, Bluetooth, GPRS support and so on, for the development MIDlets in the areas mentioned. Many versions of MIDP exists, each new versions supports earlier ones and more functionalities. Example, MIDP version 1.0 gives two APIs in order to create user interfaces, these are high level APIs for creating objects like textboxes, command bittons, check boxes just like in the AWT libraries of J2SE. The low level APIs gives users (MIDlets Developers) the ability control screen, keyboards,

The version 2,0 provides backward compatibility with previous versions except in memory where version 1.0 is used for devices with 128KB non volatile memory, version 2.0 is used for devices with 256KB non volatile memory and provided more enriched classes for development of applications in the area like networking, SMS messaging, record management and database support, Game, bluetooth etc.

2.2.3 ANDROID

Android is an Operating System for low resources device that runs on battery and are full of hardware devices like Global Position System (GPS) receivers, Cameras, WiFi, UMTS (3G telephony) connectivity, Touch Screen etc. Android is a multi-tasking operating system that uses Delvik Virtual Machine which can run on devices with minimum total of 64 MB memory, (Jianye et-tal, 2011).

The architecture of Android OS tack is subdivided into five layers; Application, Framework, Libraries, Android Runtime and Linux kernel Layers.

a. **Application Layer:** is a for developing all Android Applications such as Browser, e-mail client, SMS Applications like TomdApp etc. Programs in this layer are written in Java Programming Language.
b. **Framework Layer**: defines the framework for developing android application which includes Rich and extensible views for building User Interface like Textbox, buttons etc.
    i. Content Provider to enable androids applications access or share data such as contacts information, memory access etc.
    ii. Resource Manager which provides access to non coded localize strings, graphics, layout files etc.

iii. Notification Manager enables android applications displays customs alerts on status bar

iv. Activity Manager is responsible for managing activity lifecycle and provides common navigation back stack.

c. **Libraries Layer:** this layer consists of sets of C/C++ libraries that are used by various components of the Android's System and it also provides supports for the framework layer.

d. **Android Runtime Layer:** comprises of sets of core libraries and Delvik Virtual Machine. This layer is placed directly on top of Linux kernel layer provides support to the application framework.

e. **Linux Kernel Layer:** acts as an abstraction layer between the hardware and the rest of the software stack. It provides core system services such as memory management, process management, network stack and driver model etc. Figure 2.2 is a snap shot of Android OS software stack.

Figure 2.1: Android Architecture.

2.2.3.1 COMPONENTS OF ANDROID APPLICATION

An Android Application is based on four components; Activity, Service, Broadcast Receiver and Content Receiver.

a. **Activity** is described by Jianye Et-tal (2011) as a single screen with user interface that presents data like (image) or allows interaction (like Button, Textbox). An Android Application can contain one or more activities. Activities in an application work together to form a comprehensive user experience, but each activity is independent of another, therefore, different application can start any individual activity. All Activities are subclasses from android.app.activity and their lifecycle are controlled by OnXYZ()

methods. In a multi activity application, one activity is designated as main-activity which is presented to user when the application is launch just like index.html in the web application. Each activity can then start another activity just like hyper link in web application. When new activity starts, the previous running activity stopped but it is preserved by the Android System in a stack (Back Stack)

b. **Service:** is an Android components that runs in the background to perform long-running operations or to perform work for remote processes and does not provides user interface, but an activity can connect to it (service) that is running. Jianye Et-tal (2011)
c. **Broadcast Receiver:** is in charge of the reception of system wide broadcast and take response aiming at the information that a broadcast transmitted. Example, a broadcast can announce that screen has turned off, battery is low etc. A broadcast can originate from system or application and there can be more than one broadcast receiver in an application. Jianye Et-tal (2011)
d. **Contents Provider:** is a component that provides data share mechanism among android's applications. The data shared could be in file system, SQLite database, or any other persistence storage an application can access. Content Provider defines the data format it supports and sets of method that enables other applications to query or modify the data. It is implemented as a subclass of content provider. Jianye Et-tal (2011)

Three of the four components types- Activity, Service and Broadcast Receiver are activated by an asynchronous message called Intent. Intents bind individual component to each other at runtime. Activities and services defines the action to be perform and specifies the URI of the data or action, but for broadcast receivers, the intents only defines the announcement being broadcast, while content provider is not activated by intent, rather it is activated when targeted by

a request from content resolver.

2.2.3.2 ANDROID APPLICATION LIFECYCLE

A state of Android Application is determined by the state of its components, most importantly its activities. Figure 2.2 is a graphical representation of Android's Application Lifecycle.

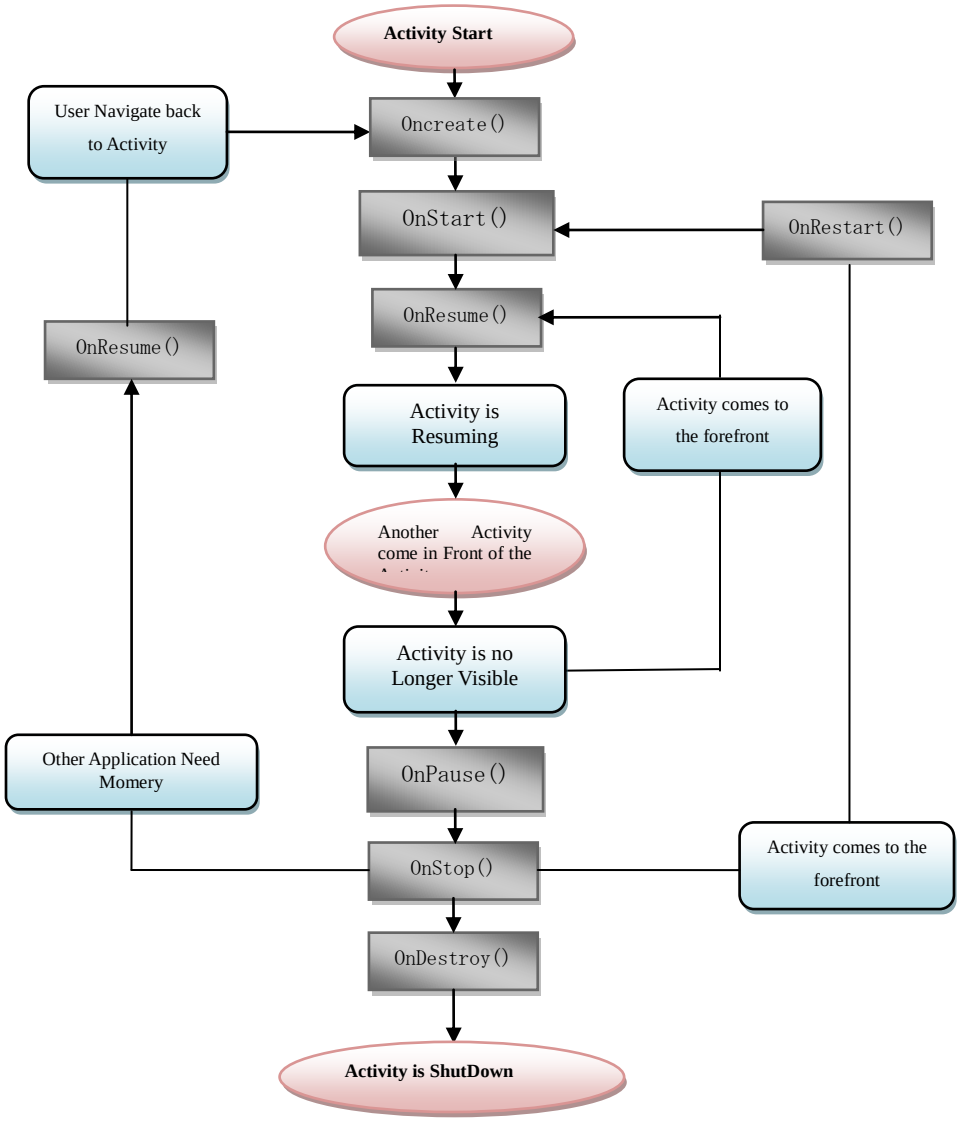

Figure 2.2: Android Application Life Cycle.

OnCreate(): called on the method is first created. Inside this is where static set up, creation of view, binding of data to list and so on is done. It is always followed by OnStart().

OnRestart(): called after an activity has been prior to it being started again. It is followed by

OnStart().

OnStart(): this call is made before the activity becomes visible to the user. If the activity comes to the foreground, it is followed by OnResume() or by OnStop() if it is hidden.

OnResume(): is called before the activity starts interacting with user. At this point it is at the top of the activity stack with the user input going to it and it is followed by OnPause().

OnPause(): is called when the system is about to start resuming another activity. It is usually called to commit unsave changes to persistence data, stop animation or other thing that may be consuming CPU. If it returns back to the stack's top, it is followed by OnResume() otherwise it is followed by OnStop() if it becomes visible.

OnStop(): called when the activity is no longer visible to the user, usually when it is destroyed or another activity (existing or new) has been resumed. It is followed by OnRestart(), if coming to interact with user or by OnStop() if going away (Shutting down).

2.2.3.3 ADVANTAGES OF ANDROID

Android Operating System provides some benefits and advantages over other mobile device operating systems. Some of the advantages are as follows;

a. Resources are separated from the source code
b. All non code resources are define in an XML files (Android Main.xml etc)
c. Every resource that is included in Android Project, the SDK build tools defines a unique integer ID it which can be used to reference it from the application or from other resources define in the XML files.
d. Provision of resources separate from source code makes it possible to update various characteristics of an application to be modify without the modifying the source code.

e. Android OS provide a set of alternative resource that enables developers to optimize the application for a variety of device configurations such as screen size or language etc.

2.2.4 GSM/GPRS MODEM

According to Sommaire (2010), A GSM/GPRS Modem is a wireless device that works with GSM wireless telecommunication network for sending and receiving data through radio waves. A GSM Modem can be external device or a PC Card/PCMCIA Card. A typical external GSM Modem is connected to Computer through a serial or USB port, while a PC Card/PCMCIA Card is usually used in a Laptop Computer connected via the PC Card/PCMCIA Card slot.

A General Packet Radio Service (GPRS) Modem is also a wireless device that allows transmission of data over GSM network. It is an extension of GSM Modem as it has higher data transmission capacity than GSM Modem. GSM Modem allows about 6-10 sms per second transmission, which is the utmost achievable performance while in GPRS 30 sms per second is achievable. GSM/GPRS Modem requires SIM Card from the Wireless GSM Carrier Operator in order to function. GSM/GPRS Modem can be control from computer with AT-commands.

2.2.5 SMS MESSAGE

According to Sommaire (2010), SMS stands for Short Message Service and it is a globally accepted wireless service that allows transmission of alphanumeric message between mobile phone users and external system such as paging, voice mail system etc. SMS messages are handled by Short Message Service Centres (SMSCs) that is hosted within the operator network. SMS is an easy to use service that has become extremely popular with billions of message handle

daily. The “short” refers to the maximum number of text handles in SMS, where 160 characters (texts, numbers and symbols) are allowed in Latin alphabets and other characters (Chinese, Arabic) maximum allowed are 70 characters.

SMS is based on store-and-forward technology. Messages are sent to an SMSC located at Operator’s side, held for the intended recipient, and then sent from the SMSC to the recipient's mobile phone. SMS is capable of confirming whether a message has been delivered, though it is up to the individual carrier to implement this capability. The confirmation function is handled by the SMSC

According to Developer's Home (2010), SMS was started in 1992 in Europe, included with Global system of Mobile Communication (GSM) standards by the European Telecommunication Standard Institute (ETSI) but later it was carried to other wireless telecommunication networks like Code Division Multiple Access (CDMA), Time Division Multiple Access (TDMA). It was first developed by ETSI but today 3GPP has taken over the development and maintenance. SMS are used for;

a. Peer-to-peer communication e.g person-to-person messaging.
b. Value added Service: provisions of information like news, weather reports, alerts and notification etc.

2.2.6 MESSAGE GATEWAY

According to Sommaire (2010), Message Gateway is a program/Application or tool that allows exchange of SMS message between two or more Short Message Service Centers (SMSCs) that support same or different SMS Protocols. Different GSM Networks Operators may be operating with different SMS protocols, SMS gateway translates one protocol to another to make exchange

of SMS possible between them possible, thus it provide interoperability between the different SMS protocols. SMS gateway is necessary for the following reasons:

a. Compatibility - Supports inter-networking amongst various SMSCs implemented by various vendors
b. Conversion of SMS protocols such as SMPP, CIMD, and UDP to HTTP format and vice versa.
c. Handles and control of large SMS Message traffic
d. Keeps track of the records such as user details, time stamp billing etc.
e. Supports APIs for easy plug in of applications.

There are so many SMS gateway, developed by different vendors. Some of the SMS gateways are free while others are commercial such as MOBITSMS, NowSMS, MS SMS GATEWAY, Gammu, Wammu, WEBSMS, OzekiNG etc, each with different capacity and functionalities.

MS Gateway: is a messaging gateway for use with a modem UMTS/3G/GPRS. It is professional, reliable and easy to use. It offers many opportunities for integration with other systems through text files, ODBC, SQL, FTP, XLS, MDB, CSV, and HTTP requests. It runs in user mode or as a Windows service. You will find the characteristics of your business, such as broadcasting, transmission, automatic response; the importation of any format, models of pre-paid subscription security services an many more.[30]

Ozeki NG (2010) described Ozeki NG gateway as having high capacity SMSC connection over SMPP, UCP, CIMD2 and GSM Modem that can be used over HTTP SMS connection. It is a good platform for SMS notification, SMS paging, two way SMS system, or for beaming an SMS service provider. It supports SMS API for SQL, .NET, ASP, ASP2, VB, C#, C++, Delphi, PHP,

JAVA and for OEM vendors. It is capable of sending up to 400 SMS per second and can allows up to 1000 simultaneous connected applications

WebSMS is a program that is installed on the user's PC which allows sending Short Messages to mobile phones around the world using SMS-Servers. WebSMS keeps an easy-to-maintain address book of friends, colleagues or business partners like a normal Email program. There is no need to remember the cellular phone number and to type it when you try to send a short message using a WEB page. - WebSMS allows adding a personal signature to each short message you can send. - WebSMS has a list of predefine standard messages, that you can easily access and put in your short message. - WebSMS contains a history list, which shows you all the short messages you have sent, the date, and the send status. - WebSMS has a bulk SMS function, which allows sending text and flashing SMS to clients, club members, and any kind of announces. - WebSMS supports now several SMS Service providers like clickatell, Mobilant, SMSWhiz. These providers have worldwide and very reliable SMS services. It is a commercial gateaway. [30]

MCTEL (2010) is described as an SMS gateway that allows Value Added Service Providers (VASP) and corporate organizations to connect to the Mobile network Operator's gateway in order to manage SMS and offer SMS Value Added Service and Applications. It support large message forwarding and exchange with optional connection like FAX, MMS, TELEX, EMAIL, and Voice text to speech, C++, PHP, HTTP and HTML.

Chow (2009) described gammu as an SMS gateway program for building SMS Server which can be install on most of the popular Operating System such as Microsoft Windows, Linux Ubuntu, MacOs etc. It is a command line program written in C programming Language. Gammu

consists of several packages in binary files such as gammu sms daemon, gammu library and Python bindings can be used for developing robust applications. For gammu to know each phone and its features, it needs to refer to the gammu phone database which is configured in the gammu configuration files. Gammu uses the settings in the configuration files to detect the phone. Several models of cell phone are supported by gammu ranging from the old to the new models such as Alcatel, Apple, Huawei, LG, Motorola, Nokia, Sagem, Sonny Erickson, and Samsung etc. gammu provides access to many phone features such as

a. Call listening, mailing and handling.
b. SMS retrieval, backup and sending
c. MMS retrieval
d. Phone listing, export and import.
e. Calendar
f. Retrieval of phone and network information.
g. Access to phone's files system.

Gammu has many commands for manupulating mobile phones and most of theinclude several options. The sysntax of the command is; gammu [--] implies' command [option] eg gamm u--identify

for commands related to call, gammu has answerall, cancelall, dialvoice, divert, holdcall etc and some commands related to SMS message are savesms, getsms, addsms, backupsms, sendsms, setsmsc and it also has command related to phone memory, picture, ringtone calendar such as getmemory, searchmemory and so on.

CHAPTER THREE: DESIGN AND IMPLEMENTATION

3.0 INTRODUCTION

The chapter has three Sections, Section 3.0 introduced the chapter. Section 3.1 outlines the design of the overall system architecture, characteristics and the various parts that made up of the entire system. Section 3.2 presents the detail implementations of the various design of the system components and Section 3.3 conclude the chapter.

3.1 OVERALL SYSTEM DESIGN

As described in the beginning of this chapter, this Section presents the detail design of the SMS Based Road Traffic Offence Tracker and Profiling system which we also refer to as System. The Overall System consists of two main parts; the SMS Messaging Application which is the frontend Application and the SMS Message Processing Engine and System Database. The frontend was implemented with Java 2 Micro Edition and Android. Two versions of the application were produced. The MIDlet version for use with Java enabled phone and the Android version for use with Android cell phones. Backend application is the messaging processing system and it was implemented with SQL, PHP and Bash scripts program

3.1.1 CHARASTERISTICS OF THE NEW SYSTEM

The system is meant to receive users' requests in form of SMS messages, process the requests by extracting RTOs information from the TomDatabase and sends the processed information back to users' as reply SMS message.

Based on the above therefore, the mode of interaction is master and slave interaction which is better known as client server system. However, unlike in the usual client server system where the mode of communication or the protocol used is HTTP, in this system the mode of interaction

is Short Message Service (SMS) of GSM network.

Simplicity and universality of Short Message Service (SMS) is what informed our choice of it for use in this system. SMS is a communication protocol which allows exchange of short textual messages between mobile device users. It is one of the most popular services provided by GSM network.

Our choice of SMS would allow FRSC to use existing GSM infrastructures (CUG Phones and Billings systems) already used by the Corps in partnership with Globacom Network to run the application at zero cost.

3.1.2 ARCHITECTURAL DESIGN OF THE SYSTEM

Figure 3.1 illustrate the overall system. The System is made up of Frontend and Backend Applications. The frontend application was implemented with J2ME Technology and installed on the users' mobile cell phone. The backend comprises of GSM Modem connected to PC Server via USB Port, the system database called (TomDatabase and SMSD) and the SMS Message Processing Application.

The database stores the Road Traffic Offences (RTOs) records while the SMSD stores SMS Message sent by user via the GSM Modem. The SMS Message processing application is responsible for processing the SMS Message stored in the SMSD database in order to retrieve the required information of RTOs from the TomDatabase. It is made of up bash script files that contains a series of SQL, PHP and Bash Scripts files. Gammu was used as the sms gateway. It was configured by the two ini type files known as gammurc and smsdrc configuration files. The two ini files contained the settings that were used to interface the GSM Modem with PC Server as well as the SMSD database.

Gammu SMS Gateway is a program that permits sending and receiving SMS Messages from a

computer automatically. Gammu was chosen for this Research work because it is free software and it supports almost all mobile cell phones, GSM/GPRS modems and it can be installed on Windows, Linux, Mac OS systems etc. It is also used for Online or Offline applications development

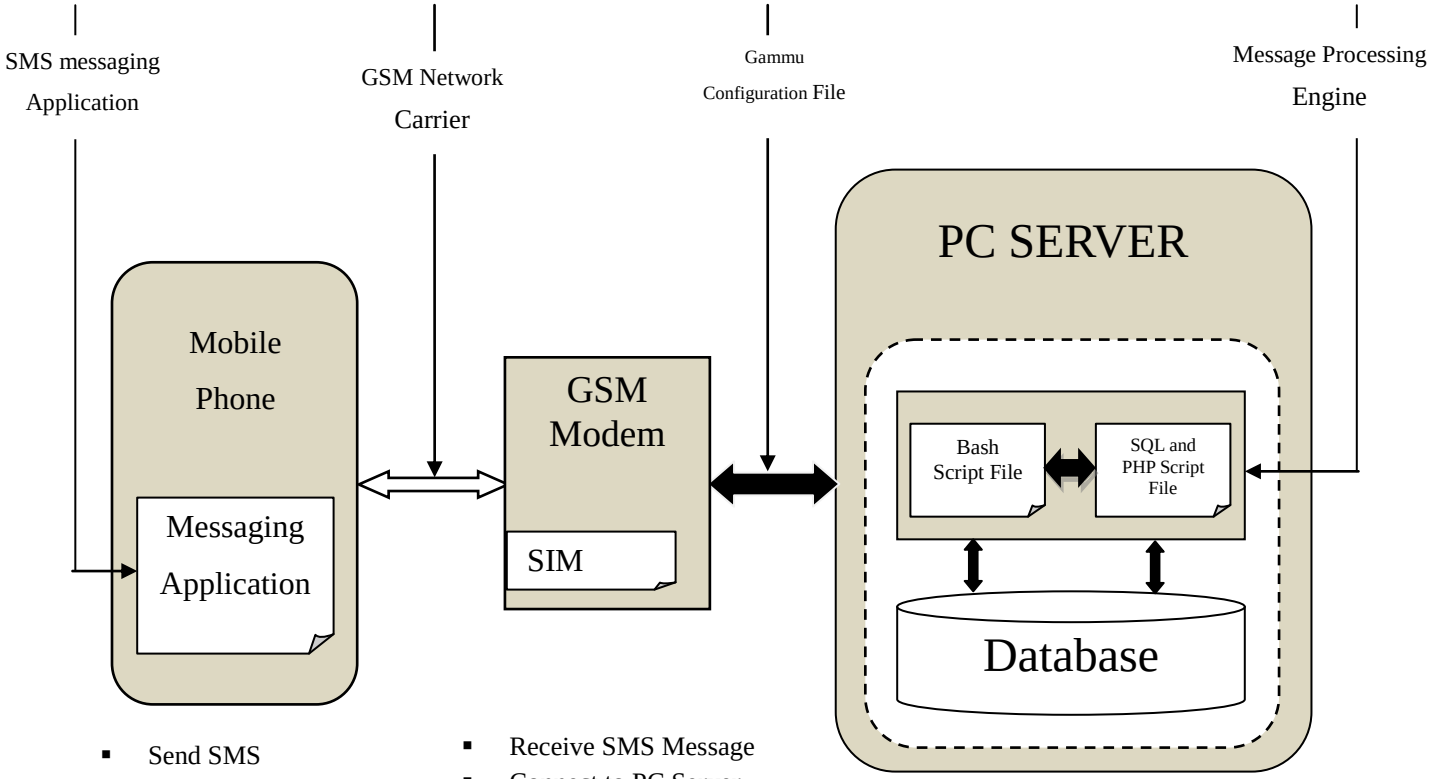

Figure 3.1: Overall System Architecture.

3.1.3 DESIGN OF SMS MESSAGING APPLICATION

Since the system is a client/server system, the SMS Messaging Application is the client side application through which the users interact with server side. SMS Messaging Application is used for composing and sending SMS message to the SMS processing application. Two version of the application were designed, MIDlet and Android versions for used with Java and Android enabled cell phones. It comprises of two windows, login and SMS composing windows for authentication of the system's users and composing SMS message. The login window serves as the main window just like the index.html in web application. It is the first window that is displayed when the application is launched. Users' are expected to supply information and click OK button to login and compose message, if the information they supplied is correct, otherwise the system display an error message informing user to re-supplied the login information and try again. The system shutdown when three unsuccessful login attempts are made. On successful login, the SMS composing window is displayed to allow user compose and send SMS message to the backend.

3.1.4 STRUCTURAL DESIGN OF SMS MESSAGE

The SMS Based Road Traffic Offence Tracker and Profiling System (RTO Tracker) had been designed to accept and process SMS message that conform to A designated format. SMS Message designed for the tracker has two parts;

a. SMS Body: this part contained only alphanumeric characters which must be a valid vehicle registration number or person driving license number.
b. Information Access Security Number (IASN): this is a numeric character or code which

must be attached with SMS Body and send to the backend together. IASN is used to determine the system module to execute and the type of information to access.

Four modules were designed for the system in order to achieve the objectives of the project. The modules are

a. **Findoffender:** for processing multiple offenders information request. It process and returns number of times a traffic offender was arrest by FRSC.
b. **CheckPenaltyPoints:** this compute and returns the total penalty points accumulated by RTO for offences committed and sanction.
c. **FindVehicleOwner:** which process request of vehicle's owner profile.
d. **FindlicenseOwner:** process request ofdriver's licence owner profile.

3.1.5 DESIGN OF SMS PROCESSING APPLICATION

Here, we deal with the design of the SMS Message Processing Application. It is an application that retrieve users sms messages from the inbox table of the smsd database, decomposes the message into segments and uses the decomposed message to retrieve required information from the TomDatabase and inserts the information into the outbox table from where it is sents to users as a reply message.

The application was implemented with Bash, PHP and SQL Scripts, in several files which are executed in batches. The files and the flow of their execution are shown in Figure 4.1.

SearchInfo.sh: is a Bash Scripts file that contained series of command lines statements which are executed in batches. It is invoked by the gammu daemon whenever an sms message enters the inbox table of smsd database. It triggers the execution of the various SQL and PHP Scripts batch by batch. See Appendix B (a) for the scripts.

ExtractUserMsg.sql: Contains the SQL statements that, when executed extracts the message from the inbox table and write the results into **MyMsg.txt** file.

LoadUserMg.sql: this file contains SQL statements that reads the contents of **MyMsg**.txt text file word by word as separated by comma(,) and inserts each of the word into a column in the stopover table of TomDatabase. The **LoadUserMg.sql** file decomposes the message body retrieved from inbox into three parts and inserts each part into the **msgtext, sendernumber** and **selectvar** columns of stopover table. Content of selectvar column is used as decision variable in the later stage of the program.

ExtractUserReply.php: is a PHP Script file, which is executed from the **SearchInfo**.sh file. Its main function is to retrieves the last record entered into the stopover table, place the values into three variables; **$mysendno**, **$mytext**, **$myselectvar.** These variables are passed to an **ExtractInfo** function, the function then use these variables to search and retrieve the required information of the traffic offences from the TomDatabase. The $**myselectvar** variable is used as a decision parameter to select the appropriate module to execute. The module or functions are implemented in the form of SQL-statement and **$mytext** variable is used to holds the core user message sent earlier and it is used in the where clause of the SQL-statement of the module in order to retrieve the required information from the TomDatabase. The **$mysendno** holds the sender number and it is used for sending a reply message to the user. Information retrieved from the TomDatabase is inserted into the smsd database outbox table. Once inside the inbox table, the gammu daemon forwards the information to the user as a reply message via the cell-phone number earlier stored in $mysendno variable**.**

3.1.6 DATABASE DESIGN

In this section, we described in detail the design and implementation of TomDatabase. We start

by identifying source of data and the redundancy that exists in the system, relationship between data before designing the database schema and finally implementation of the schema.

3.1.6.1 DATA SOURCES

As the application was developed to solve real-life problems, we use real-life data to exercise the application. The data are sourced from various units of the FRSC. The data sources are as follows;

a. FRSC OPS 06: Offenders Register

b. FRSC OPS 07: Payment Register

c. FRSC OPS 08: Multiple Offenders Register

d. FRSC OPS 09: Vehicle Impounded Register

e. FRSC OPS 10: Wanted Offenders Register

f. FRSC MVA01: Vehicle Registration Form

g. FRSC MVA04: Vehicle Registration Book

h. FRSC MVA11: Fresh Driving License Register.

i. FRSC MVA12: Renewal Driving License Register.

3.1.6.2 ENTITY REFINEMENT

From the study carried out, we discovered that all the registers mentioned in Section3.5.1 exist as entities in the FRSC system. The problems we identified with existing FRSCVsystem are;

a. Data are stored and processed manually particularly data related to road traffic offences and offenders.
b. Duplication of data in various registers which constitute data redundancy.
c. Data are scattered across the country with no standard means of storage, access, protection and back up.
d. No connectivity between data in the various departments particularly between Operations and MVA Departments.

We refined the above entities by relegating the status of Multiple Offenders register, Impounded register and Wanted Offenders register from entity status and implementing them as reports instead. We also merged Fresh and Renewal Driving License register and created Driver Record and Renewal register entities in their place. Vehicle Registration Book was decomposed into Vehicle and Owners registers entities, while Vehicle Registration is maintained to serve a source of data to the Vehicle and Owners entities.

3.1.6.3 DATA REFINEMENT

After we eliminated the redundant entities, the following entities and attributes were obtained;

offenders_register (ticket_number, vehicle_number, make_type, driver_name, driver_address, booking_date, patrol_route, confiscation, licence_number, command, offences, staff_pin, receipt_number, teller_number, fine_amount, custody_fee, towing_fee, paymt_date, remark)

vehicle_register (owner_name, address, occupation, state, lga, town_residence, GSM/te photo, thumb_print, vehiclw_number, vehicle_make, vehicle_type, l_number, vehicle_model, chassisno, engineno, engine_capacity, colors, purpose)

drivers_licence_reg. (licence_number, name, address, sex, blood_group, facial_mark, height, weight, GSM/tel_number, issuing_date, issuing_state, last_issuing_state, expiry_date, licence_class, thumb_print, picture)

fines_register (ticketno, receiptno, tellerno, finesamt, custody_fee, towing_fee, paymt_date**).**

The above entities were further refined to remove data redundancy using the normalization techniques. *Normalization has been defined as method or technique of decomposing Entity (Table or Relation) into two or more tables in order to remove data redundancy (Morris, 2005).* Therefore, after further normalization of the above entities, we obtained the following Tables;

Offender_register (*ticketno, vehicleno, make-type, colors, purpose, driver_name, booking_date, confiscation, licenceno, route, pin, cmd_code)*

offence_register (*ticketno, licenceno*)

offence_record (*offence_code, title, fines, penalty_points*)

payment_register (*ticketno, receiptno, tellerno, fine_amt, custody_fee, towing_fee, payment_date, payment_bank*)

owner_register (*ownerID, owner_name, address, occupation, state-origin, lga, town_residence, gsm_no, thumb_print, photo_passport, remark*)

vehicle_register (*vehicleno, ownerID, make_type, chassisno, engineno, colors, purpose, register_date,*)

drivers_record (*licenseno, name, address, sex, dob, b_group, height, weight, gsm_no,*

photo_passport, thumb_print)

renewal_register (*licenceno, issuing_date, issuing_state, expiry_date, last_issuing_state, licence_class*)

bookingsheet_register (*pin, startno, stoppingno, total_used*)

issuance_register (*cmd_code, cmd_name, zone_code, zone_name*)

employee_record (*pin, name, sex, rank, cmd_code, dob, entry_qual, add_qual, soo, lga, fappt, pappt*)

3.1.6.4 SCHEMA DESIGN

We defined and established relationships between the tables according to the interaction between data in real FRSC functions.

According to Morris (2005), a relationship is an association among two or more entities in a relational database. There are broadly three types of relationship in relational database; one-to-one, one-to-many and many-to-many relationships. Relationship defends on the data interaction between entities forming the relationship.

In this research work, Entity Relation Diagram (ER-Diagram) was used to design the database schema. Figure 3.2 is the schematic diagram of the TomDatabase.

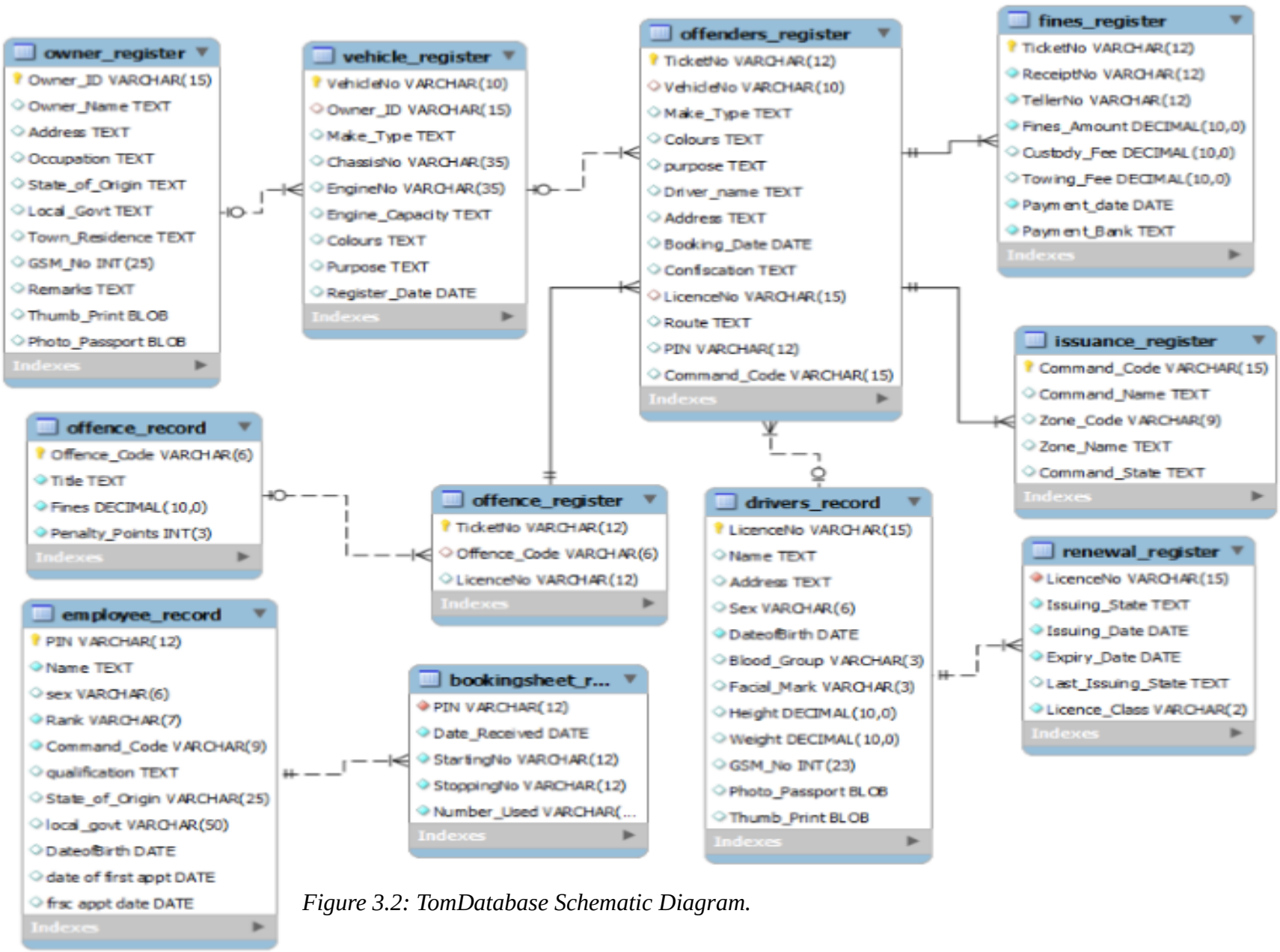

Figure 3.2: TomDatabase Schematic Diagram.

3.1.7 SYSTEM AND INFORMATION SECURITY DESIGN

The developed system is made for FRSC use only and it should be operated by the FRSC authorized personnel. These restrictions are necessary in order to protect the organization and public, because information dealt with by the system is confidential. Unauthorized access or disclosure of the information by unwanted or unauthorised persons may damage the organization integrity and endangers individual and general public security.

3.1.7.1 THREAD ANALYSIS

During the system development, we consider the vulnerability and the threats that could harm the system operations, organizational and individual security. Neglecting these threats could have serious consequences that might damage the corporate image of the FRSC and endangers the individual member of public lives whose information is illegally accessed. These threats are;

a. Unauthorized use and misuse of the system by both insiders (FRSC Staff) and outsiders
b. Unauthorized access to information by outsiders if the system is left accessible to all and sundries

3.1.8 OVERALL SYSTEM FLOWCHART

Figure 3.3 represents the overall system flowchart showing processes flow and direction as well as the various decisions during the process execution.

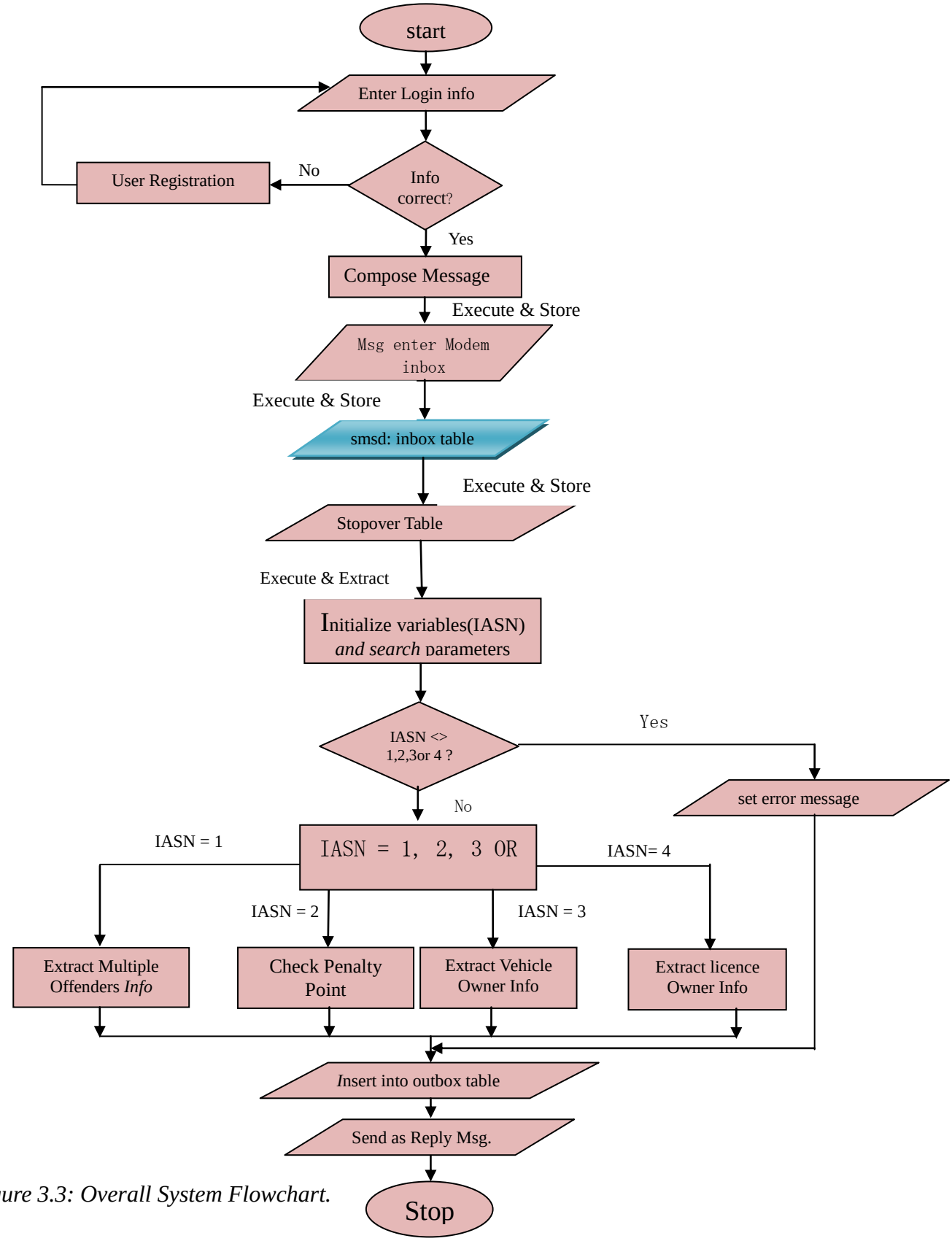

Figure 3.3: Overall System Flowchart.

3.2 IMPLEMENTATIONS

Having described the overall system architecture in the above section, we hereinafter discuss in detail, the implementation of the various system components, which we referred to as the Frontend and Backend Applications.

As stated earlier, we used Connected Limited Device Configuration (CLDC) and Mobile Device Profile (MIDP) of the J2ME Platform for implementation of the application (Java version). XML, Java APIs and Android SDK Manager were used to develop the Android version.

3.2.1 IMPLEMENTATION OF MIDlet SMS MESSAGING APPLICATION

In this phase, we were concerned with building the MIDlet (Java version) of the SMS Messaging Application. We started by identifying the required Packages (APIs) needed for the implementation of the classes of our messaging application as well as versions of CLDC and MIDP. We used the following application programs interface (APIs) to construction the application classes;

a. **Javax.microedition.io:** for managing input and output.
b. **Javax.microedition.midlet.MIDlet:** responsible for controlling the MIDlet application like creating, starting, pausing and destroying the MIDlet thus, it manages the MIDlet lifecycle.
c. **Javax.microedition.lcdui:** this API is responsible for providing the necessary features required for creating user interface object such as forms, DateField, ChoiceGroup, TextField etc for MIDP Application. The .lcdui stands for Liquid Crystal Display User Interface. Construction of the class objects were made possible by javax.microedition.lcdui.

d. **Javax.wireless.messaging:** for SMS message sending capability.

These APIs were used to write the code for the application phase under the NetBeans IDE 7.1.2 and JCreator Integrated Development environment.

COMPILING AND PRE-VERIFICATION:

The assembled code is compiled and pre-verified as illustrated in Figure 3.4.

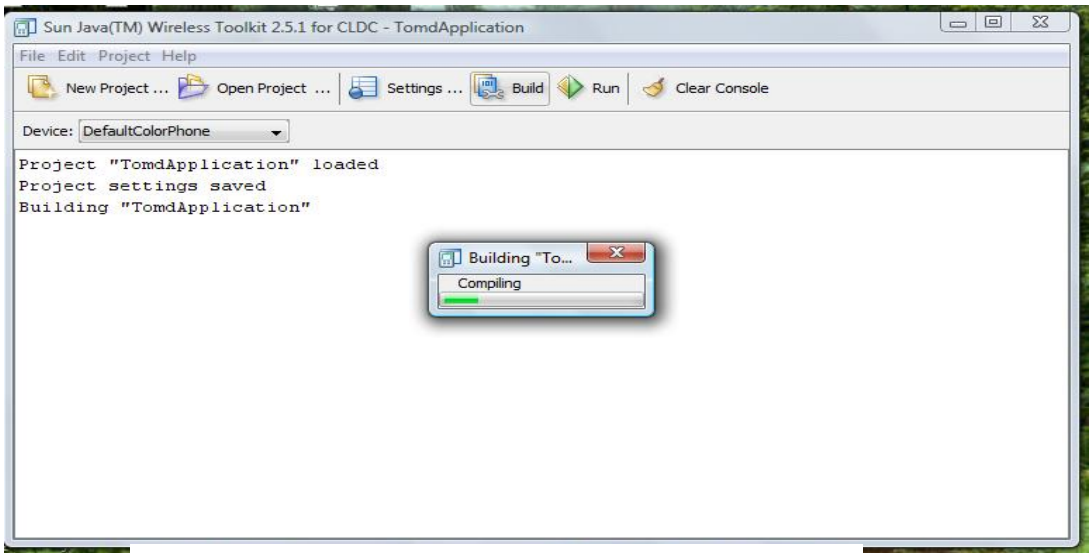

Figure3.4: Compiling TomdApplication MIDlet.

Testing was done after compilation and pre-verification. The NetBeans has a simulator Cell Phone which pops up once the **RUN** button was clicked. The simulator Cell Phone is launched and displayed the application as it would appear in a real Mobile Phone. See figure 3.5 (a), (b) and (c).

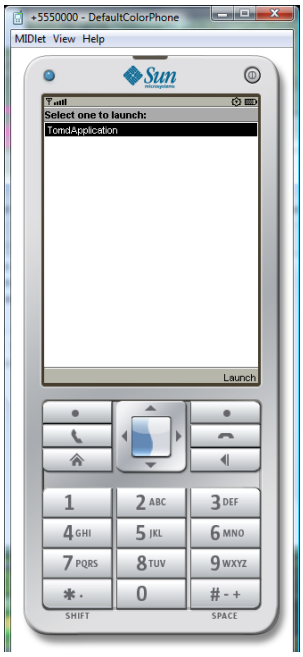

Figure3.5 (a)

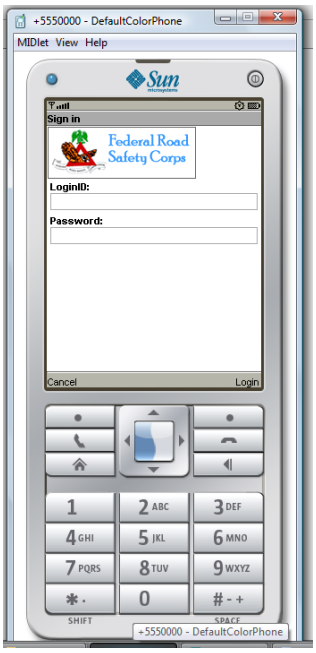

Figure3.5 (b)

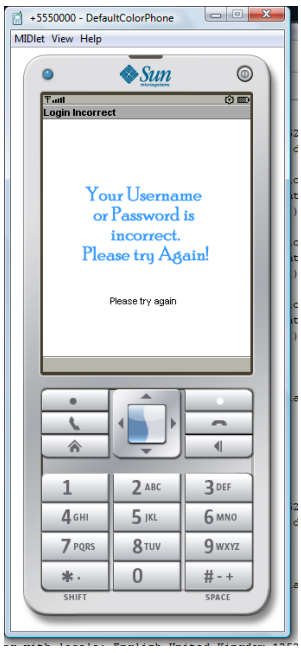

Figure3.5 (c)

3.2.2 IMPLEMENTATION OF ANDROID MESSAGING APPLICATION

Having determined the various requirements of the application (TomdApplication) in the previous stage, we went ahead to create our TomdApp project debug and compile it.

3.2.2.1 CODING

In coding this version of application, XML and Java Programming were used for. The code implementing the application is structure as follows;

a. **Assets directory:** this directory is used to hold the raw assets files for an android application like audio files, animation files etc.

b. **src directory:** is where we kept all our source files. These are R.java, TomdApp.java and the AndroidManifest.xml files.

c. **AndroidManifest.xml:** is an xml file, its structure was auto generated by the Eclipse IDE. It is where global settings of an android application such as permissions, intents filters activity views are define and saved. This file is where we defined the settings of the Tomdapp's UI, permissions and intent filters. As mentioned it is stored in the src directory of the TomdApp Project Folder. The code is shown in Appendix A (5).

d. **R.java:** is also auto-generated and non editable. It contained pointers into the application's drawables, layouts and values directories. R.java file is referenced from the AndroidManifest and TomdApp.java. See Appendix A (3).

e. **Activity_xml:** is also an xml file. It is where we defined our user interface and layout settings. This file is first called when our application load on the execution of OnCreate() method. It draws the layout the user interface components on the Activity (Form) according to the settings described in the file. See Appendix A (4).

f. **TomdApp.java:** its template was also generated by the Eclipse IDE; we develop the codes of the application classes. The codes in this file drives our application as it implements our methods. The file is named after the project by the Eclipse Wizard and it is stored in the src directory. Below are some of the APIs we included for the construction of the classes in this file;

    i. **android.os.Bundle:** to enable adding android operating system components and capability.

ii. **android.telephony.smsManager** to allow our application have sms message sending capability

iii. **android.view.View:** allow us create user interface for the application, like textbox, button etc.

iv. **android.view.view.OnClickListener** which allows inclusion of the onlicklistener class or method.

v. **android.widget.EditText** to enable creating an instance of the textbox on the Activity (form) of the application

vi. **android.widget.Button** for creating an instance of command button for the application.

vii. **android.wicket.Toast** this allows us include a splash screen that flashes when message is successfully sent.

viii. **android.AlertDialog** this allows us add a dialog box which posp up and displays a warning message every time user tried to send empty message. See Appendix A (1).

3.2.2.3 COMPILING AND PRE-VERIFICATION

Compiling and Pre-verifications of the source codes was done simultaneously as for the **.jad** application. See Figure 3.6 (a) and (b).

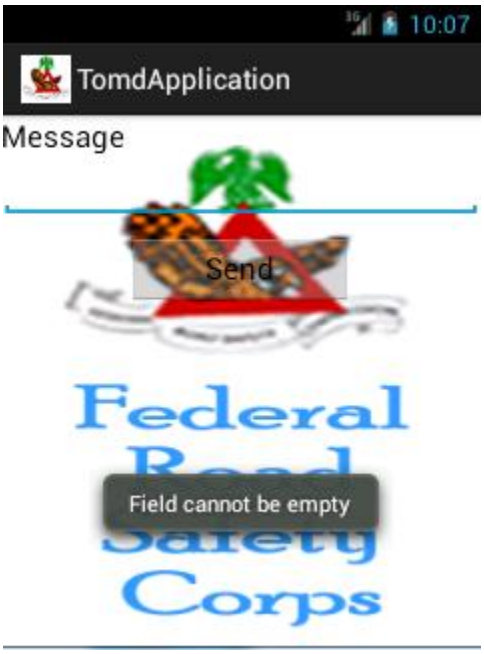

Figure 3.6a: Attempt to send empty SMS with Android Version.

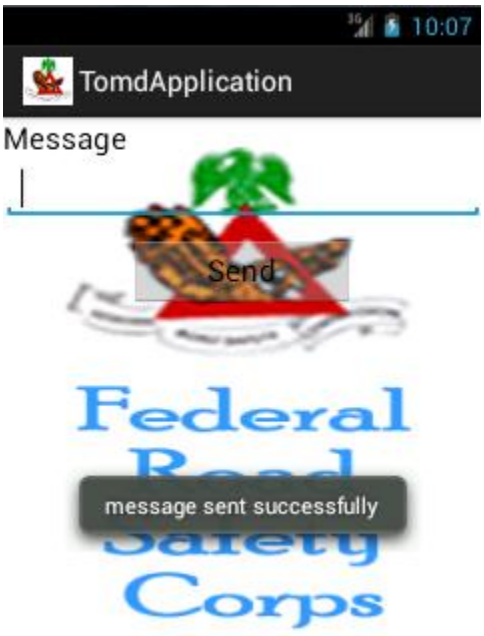

Figure 3.6b: Successful Sending of SMS with Android Version.

3.2.2.4 DEPLOYMENT

After the TomdApp was compiled, pre-verified and tested, we copied the compiled TomdApp.apk to our target Android, from the bin directory where it was auto-stored by the Eclipse IDE. When we copied the TomdApp to our target phone, we installed the application by

a. Clicking the TomdApp.exe file.
b. Then we clicked Package Installer; the package was installed successfully was displayed after the installation.

3.2.3 DATABASE IMPLEMENTATION

This is the stage where we translated our schema designed in Figure 3.2 was created on our chosen Database Management System (DBMS). We used Mysql DBMS to implement our design tables and their entities, attributes and relationship as depicted on the schema. Brief description of each table and its primary, foreign keys and relation with other table is given below;

a. **Offenders_register Table:** Table 3.1 is used to store traffic offenders' records, its structure and attributes are shown in Table 3.1. It has one-to-one relation with payment_register table such that every traffic offender pay for his fine one time and many-to-one relationship with vehicle_register, drivers_record, employee_record and issuance_register table, such that one vehicle can be arrested many times and one driver can be booked for road traffic violation many time while one employee can arrest a particular vehicle many times driven one or more registered driver(s).

Table 3.1: Offenders Register Table.

Attribute (Field)	Type	NULL	Default
TicketNo	VarChar(10)	No	
VehicleNo	VarChar(15)	Yes	NULL
Make_type	Text	Yes	NULL
Colours	Text	Yes	NULL
Purpose	Text	Yes	NULL
Driver_name	Text	Yes	NULL
Booking_date	Date	No	
Confiscation	Text	No	
LicenceNo	VarChar(15)	Yes	NULL
Route	Text	No	
PIN	VarChar(12)	No	
Command_code	VarChar(15)	No	

b. **Offence_register Table:-** Table 3.2 stores the record of offences charged every traffic offender, it has many_to-one relationship with offenders_register table because a traffic offender could be charged for one or more offences in single arrest. It has the following structure and attributes.

Table 3.2: Offenders Register Table.

Attribute (Field)	Type	NULL	Default
TicketNo	VarChar(12)	No	
Offence_code	VarChar(6)	No	
LicenceNo	VarChar(12)No		

c. **Offence_record Table:** Table 3.3 contain the detailed description of all the offences that could charged against road traffic offender. It has one-to-many relationship with offence_register table such that one offence could be recorded (charged) against an offender one or many times. The attributes' structure are shown Table 3.3.

Table 3.3: Offence Record Table.

Attribute (Field)	Type	NULL	Default
Offence_code	Varchar(6)	No	
Title	Text	No	
Fines	Decimal(10,0)	No	
Penalty_points	Int(3)	No	

d. **Payment_register Table:** Table 3.4 is for storing the record of payment made by every apprehended road traffic offender. It has one-to-one relationship with offenders_register table such that every offender can only made one payment for every arrest since no part payment is accepted. It has TicketNo as Primary-key with following attributes and structure.

Table 3.4: Payment Register Table.

Attribute (Field)	Type	NULL	Default
TicketNo	VarChar(12)	No	
RecieptNo	VarChar(12)	No	
TellerNo	VarChar(12)	No	
Fines_amount	Decimal(10,0)	No	
Custody_fee	Decimal(10,0)	Yes	NULL
Towing_fee	Decimal(10,0)	Yes	NULL
Payment_date	Date	No	
Payment_Bank	Text	No	

e. **Owners_register Table:** Table 3.5 is for keeping personal record of every vehicle owner that registered and obtained a vehicle registration number for his vehicle(s). This table has one-to-many relationship with vehicle_register table in the sense that one, person can have one or more vehicle_registered. OwnerID is the primary-key of the table and the rest of the attributes and structure are shown below;

Table 3.5: Owner Register Table.

Attribute (Field)	Type	NULL	Default
OwnerID	VarChar(15)	No	
Owner_name	Text	Yes	NULL
Address	Text	Yes	NULL
Occupation	Text	Yes	NULL
State_of_origin	Text	Yes	NULL
Local_Govt	Text	Yes	NULL
Town_residence	Text	Yes	NULL
Gsm_No	Int(25)	Yes	NULL
Thumb_Print	Blob	Yes	NULL
Photo_Passport	Blob	Yes	NULL
Remarks	Text	Yes	NULL

f. **Vehicle_register Table:** Table 3.6 is for keeping record of every registered vehicle, it has

many-to-one relation with owner_register table and one-to-many relation with offenders_register table, such that many vehicle(s) could be owned and registered by one person and each off the vehicle could be involved in one or more road traffic violation (arrested). This meant that for every vehicle owned by a person, there is possibility that it could be arrested one or many times for traffic offence violation. The table has the following attributes and structure with VehicleNo as the primary-key.

Table 3.6: Vehicle Register Table.

Attribute (Field)	Type	NULL	Default
OwnerID	VarChar(15)	No	
Owner_name	Text	Yes	NULL
Address	Text	Yes	NULL
Occupation	Text	Yes	NULL
State_of_origin	Text	Yes	NULL
Local_Govt	Text	Yes	NULL
Town_residence	Text	Yes	NULL
Gsm_No	Int(25)	Yes	NULL
Thumb_Print	Blob	Yes	NULL
Photo_Passport	Blob	Yes	NULL
Remarks	Text	Yes	NULL

g. **Drivers_record Table:** Table 3.7 is where personal information of all driving license holders' is kept. It has LicenceNo as the primary-key and one-to-many relationship with renewal_register and offenders_register tables. The relationship is such that a person (license holder) can renew his licence one or many times and can as well be arrested for

road traffic violation one or many times. The structure of the table is shown below;

Table 3.7: Drivers Record Table.

Attribute (Field)	Type	NULL	Default
LiccenseNo	VarChar(15)	No	
Name	Text	No	
Address	Text	No	
Sex	Varchar(6)	No	
Date_of_Birth	Date	No	
Blood_group	VarChar(3)	Yes	NULL
Height	Decimal(10,0)	Yes	NULL
Weight	Decimal(10,0)	Yes	NULL
GSM_No	Int(25)	Yes	NULL
Photo_Passport	Blob	Yes	NULL
Thumb_Print	Blob	Yes	NULL

h. **Renewal_register Table:** Th Table 3.8 has LicenseNo as Index-Key and is for keeping record of license renewal and has many-to-one relationship with Drivers_record table such that every license owner can only renew his license one or many times. The following is the table structure and attributes

Table 3.8: Renewal Register Table.

Attribute (Field)	Type	NULL	Default
LicenseNo	VarChar(15)	No	
Issuing_state	Text	No	
Issuing_date	Date	No	
Expiry_date	Date	No	
Last_issuing_state	Text	Yes	NULL
Licence_class	VarChar(2)	No	

i. **Booking_sheet_register Table:** Table 3.9 keep record of all booklet of offence sheets. In FRSC every command is given a booking sheet. A Booking Sheet is a Booklet that contains 50 Offence Sheets, in Triplicates. This information is stored in booking_sheet_register table. The table has PIN (Personal Identification Number) as index and many-to-one relation with employee_record table such that one employee can collect the Offence Sheet one or many times. The structure of the table is shown below;

Table 3.9: Booking_sheet Register Table.

Attribute (Field)	Type	NULL	Default
PIN	VarChar(12)	No	
startingNo	Int(12)	No	
StoppingNo	Int(12)	No	
Number_used	Int(3)	No	

Table 3.10: is the Issuance Register Table; it used for keeping record of booking sheet booklets issued to FRSC Commands (Formations) nationwide. This table has one-to-many relationship with offenders_register table.

Table 3.10: Issuance Register Table.

Attribute (Field)	Type	NULL	Default
Command_code	VarChar(20)	No	
Command_name	VarChar(25)	Yes	NULL
Zone_code	VarChar(1)	Yes	NULL
Zone_name	Int(10)	No	Auto_increment

j. **Employee_record Table:** Table 3.11 is an excerpt of the entire table's attributes and structure. Record of the entire FRSC employee is stored in this table. It has one-to-many relationship with offenders_register and Booking_sheet_register tables, such that one employee can arrest one or more traffic offenders and can also collect Offence Sheet Booklet one or many times. It has PIN as primary-key and other attributes and structure as shown in tables 3.11 below;

Table 3.11: Employee Record Table.

Attribute (Field)	Type	NULL	Default
PIN	VarChar(12)	No	
Name	Text	No	
Sex	VarChar(6)	No	
Rank	VarChar(7)	No	
Command_code	VarChar(9)	No	
Entry_qualification	Text	Yes	NULL
Add_qualification	Text	Yes	NULL
State_of_origin	VarChar(35)	No	
Local_govt	VarChar(50)	No	
DateofBirth	Date	No	
Dateof_FirstAppt	Date	No	
Dateof_preAppt	Date	No	

k. **Stopover Table:** Table 3.12 is the stopover table. it used to store the decomposed sms message extracted from the inbox table of the SMSD database. It has ID as primary-key that increases automatically every time message enters the table.

Table 3.12: Stopover Table.

Attribute (Field)	Type	NULL	Default
SenderNumber	VarChar(20)	No	
MsgText	VarChar(25)	Yes	NULL
Selectvar	VarChar(1)	Yes	NULL
ID	Int(10)	No	Auto_increment

3.2.4 GAMMU SETTING

Gammu is the SMS Gateway that we used to pass SMS messages from the GSM modem to the PC Server. In order for SMS to be passed, we configured the gammu to allow the PC Server and the GSM modem communicates with each other. We used two ini type files known as gammurc and smsdrc configuration files.

Gammurc is a configuration file that reads information concerning the connected GSM Modem and the type of connection in order to communicate with the modem. Several modem or combination of modem and cell can be connected but each must be configured in separate section. Each section is identified by square bracket with the word gammu inside i.e. [gammu], [agmmu1]….[gammuN].

Gammurc file has parameter that must be sets in order for communication to take place. Below is brief explanation on parameters of our gammurc configuration file;

Connection: it is a protocol used to talk to the GSM Modem e.g COM1. Fbus, mbus, dku5, at etc.

Port: this parameter sets the address of the modem as connected to the computer e.g COM1, USB1, /dev/ttyUSB0 etc.

Model: sets the model of the GSM Modem parameter e.g E156G, Fastrack, E226G etc.

Synchronize time: used to set time from computer to GSM Modem during initializing a connection.

Logfile: used to set location where communication records between computer and the modem is kept.

The settings of the gammurc configuration file for this research work is given below;

[gammu]

port = /dev/ttyUSB0:

connection = at115200

model = E156G

smsdrc configuration file is for gammu short message service daemon (gammu-smsd). Gammu-smsd is a program that periodically scans GSM Modem for message received, it fetch and stores them (message) into a specified storage or location as defined in the smsdrc configuration and also sends the message from the specified storage to a specified destination. So we used this file to configure our storage location which is smsd database.

Smsdrc configuration file has two part: [gammu] and [smsd], thus the gammu and sms daemon part. The gammu part has the same setting as that of gammurc excluding the logfile parameter. The smsd part consist of several parts e.g [exclude-number] for blocking message from some Cell Phones, [include-number] for allowing message from some Cell Phones etc. It has option for setting message storage which could either be in file or database. Many databases are supported. Below are brief explanations on some of the smsdrc file parameters;

Service: used to set storage media e.g files (store message in file), mysql (store message in mysql database) and if left blank i.e NULL (does not store message).

PIN: to set the SIM cards security pin is in used

LogFile: to set the location where smsd action will be stored.

MaxRetries: to set the maximum number of times smsd will try to resend message if sending fails. The default of this setting is 1

DeliveryReport: for keeping the delivery report, either log, sms or no report should be set

RunOnReceive: to set the program to execute or run after receiving message.

User: to set the username of the database to connect to

Password: password used to connect to the database

PC: the address of the database

Database: the database name to use.

The smsdrc settings for our project is given below;

gammu]

port = /dev/ttyUSB0

connection = at115200

model = E156G

[include numbers]

08065908729

08094360788

[smsd]

Service = mysql

Driver = native_mysql

PIN = 0000

LogFile = /usr/smsdlog1

User = root

Password =

PC = 127.0.0.1

Database = smsd

debuglevel = 255

RunOnReceive = /usr/SearchInfo.bh

#inboxpath = /home/says/sms/inbox/

#outboxpath = /home/says/sms/outbox/

#sentsmspath = /home/says/sms/sent/

#errorsmspath = /home/says/sms/error/

#inboxformat = standard

#transmitformat = auto

3.2.5 IMPLEMENTATION OF SECURITY MECHANISM

We implemented the following security mechanism, to protect the system against the above perceived threats.

3.2.5.1 PROVISION OF LOGIN MECHANISM

Logging mechanism was implemented when developing the Message Processing Application by constructing the application to open only to users' who supply correct logging and user name. Users are required to supply their logging information before they can have access to the system. This was provided during implementation of the SMS messaging application. We constructed validateUser **()** class (method) implements this mechanism.

validateUser() accepts logging information from users' during login into the system, it compares the information with one supplied earlier supplied by the same users' when they registered. If the information matches, the system open for the user otherwise it displays error message. Users' are allowed to try three times before the system shutdown if attempts failed successfully.

3.2.5.2 IMPLEMENTATION OF INFORMATION ACCESS CONTROL NUMBER

Information Access Control Number (IACN) is a number that is included with a message.

User typed the number immediately after the typing (composing) their message and comma (,) as in AE584-KUR,1. Information access control number determines the type of information the system return to user. For example;

a. If user need: **Multiple Traffic Offenders Information**, type: **message body, 1**

b. If user need: **Penalty Points Information**, type: **message body, 2**

c. If user need: **Vehicle Owner Information**, type: **message body, 3**

d. If user need: **Driving License Information**, type: **message body, 4**

Failure to include the information access control number, the system returns an error message of **No Selection.** This procedure is confidential information and it is to be made known only to the authorized users.

3.2.5.3 IMPLEMENTATION OF MESSAGE FILTERING

We employed the gammu SMS messages filtering to implement this mechanism. Gammu SMS gateway has an inbuilt mechanism for filtering message that is set in the smsdrc configuration file. This mechanism is among the parameters we set in our smsdrc configuration file. It allowed processing of SMS messages from the GSM numbers listed under the parameter and blocked all SMS messages from the GSM numbers not listed. This allows us to control the Cell Phones that would access the system.

FRSC has existing GSM communication facilities (Customized User Group) Cell Phones and Billing System in partnership with Globacom GSM Company. The system is configured to accept messages from only the CUG Cell Phone Numbers.

3.3 CONCLUSION

This chapter presents the detail design of the various sub-systems or components of the SMS Based Road Traffic Offence Tracker and Profiling System and the implementation of the various design that were coupled to made up of the overall system. The System was successfully designed and implemented.

CHAPTER FOUR: TESTING AND EVALUATION

4.0 INTRODUCTION

The system is meant to assists FRSC Personnel access information on Road Traffic issues ubiquously, for quick and effective decision making in order to facilitates;

a. Apprehending of Multiple RTOs
b. Check RTOs penalty point to warrant disqualification of erring motorist from driving along Nigerian roads that accumulated sufficient penalty points.
c. Verification of Vehicle Owner and genuineness of vehicle plate number from patrol locations.
d. Verification the genuineness of a Driving License from FRSC Patrol point.

4.1 FUNCTIONAL PROCESSESS OF THE SYSTEM

The Users of the system are expected to be familiar with sending and receiving SMS Message from Mobile Cell Phone.

Users are expected to type suspected traffic offender's vehicle or license number in the text field with a comma (,) and then number (1, 2, 3 or 4) digit, depending on which function user want use. Example;

a. When finding multiple offenders, user should type: vehicle number, comma (,)and 1 digit numeral, example AE128-ABC,1
b. When checking penalty points, type: License Number, comma (,) and 2 digit numeral, example EE0100213, 2
c. When finding vehicle owner, type: vehicle number, comma (,) and 3 digit numeral, example XA232-KUJ, 3 or

d. When verifying genuine of driving license, type: license number, comma (,) and 4 digit numeral, example AA2341005, 4

After typing the message, user click the OK (Send) button to send the message, which is forward to the GSM Modem attached to the PC Server at the backend. Once the message enters the Modem inbox, it is automatically fetched and transfer to the smsd database inbox tables by gammu sms gateway. Communication between gammu and PC Server is established by the gammurc configuration file and communication with the smsd database is established by smsdrc configuration file. Gammu smsd daemon RunOnReceive parameter had been set to runs the SearchInfo.bs program whenever message enters the smsd database inbox table.

RunOnReceive: is a parameter of the smsdrc configuration file that we set to call and execute the SearchInfo program. The SMS Processing Application starts processing the message in the inbox when the SearchInfo program execute.

SearchInfo contains series Linux Bash commands or statements, in batches which are executed sequentially.

The first statement in the SearchInfo file is a call to **ExtractUserMsg**.sql file. ExtractUserMsg file contained SQL statements that are structured to selects sender number and message content from the smsd inbox table whenever it is executed and write the results to **MyMsg**.txt text file. The next statement is a call to **LoadUserMsg**.sql file, it also contain an SQL statement. When it is executed, it reads the message earlier written to **MyMsg** text file, word by word as separated by comma (,) and inserts each word (segment) into a column in **stopover** table. This is where the body of users' SMS message is decomposed into three parts and each part is inserted into **msgtext, sendernumber** and **selectvar** columns in **stopover** table.

After decomposing the message, the next statement calls and executes the ExtractReplyMsg.php

script file.

ExtractReplyMsg.php is a program file that contains PHP script program or statement which is executed from the command prompt and not in the usually method or way of executing PHP program, which is normally via Web Server. When statements in the ExtractReplyMsg.php file are executed, the following tasks are performs;

a. It opens connection to the smsd database.

b. It extract the values in the last row of the **stopove**r table and

c. The values retrieved from the stopover table are used to determine the module to execute, the type of information to extract from the Tomdatabase and the destination to send the reply SMS message. For example, values retrieved from;

   - **SenderNumber column** is used to send reply message to users'.
   - **msgtext colmn** is used as search parameter to determine the needed information to retrieve from TomDatabase.
   - **selectvar column** is used to determine the module that would be executed.

d. Retrieved information from TomDatabase is inserted into the outbox table of the smsd database and forwarded to user as reply SMS message via the values retrieved from the sendernumber column of stopover table.

The following five options are used to determining the function to be executed;

Case 1: when selectvar = 1

Retrieve Multiple Traffic Offenders' Information

Case 2: when selectvar = 2

Retrieve Penalty Points Information

Case 3: when selectvar = 3

Retrieve Vehicle Owner Information

Case 4; when selectvar = 4

Retrieve Driving License Information

Case 5: when selectvar **Is Not Equal to** 1, 2, 3 or 4:

Return Record Not Found Message

4.3 SYSTEM TESTING AND EVALUATION

After successful implementation of the system, we carried out testing to ensure that all the individual function that made up of the overall system meet its expected requirement and are performing accordingly. The following tests conducted are;

a. **Functional Testing:** we conducted this test by running each function separately and ascertained the level of its performance and compliance with our specified requirements before integration. Test conducted are;

   i. Frontend Application Testing

   ii. Function Processes Testing.

The results we obtained are displayed in Table 4.1.

b. **Reliability Testing:** this we conducted by running the system (Application) to ascertain that it can run continuously for 24 hours without breaking and also to ensure that it can sustain the optimal load expected to be handled, which is at most 10 sms messages per second. The test was conducted and system was found to be capable of performing at

optimal capacity without degradation.

c. **Evaluation:** here we evaluated the performance of the Overall System's Performance in respect to its response to user requests. Our aim is to find out if the System's Response Time is within acceptable level. We carried out the test by increasing the load (Number of Records) in the TomDatabase and then measure the System's Response time with each increase. See Table 4.2.

Table 4.1: Testing of Messaging Application (Functions).

S/N	Test	Pre-condition	Test Step	Expected Results	Status
1.	Login a. With wrong usename and password		▪ User select and click the Application ▪ Fill in the login form with wrong username or password or both	System returns incorrect login info alert	Passed
	b. With correct username and password	User has been register to use the system previously	▪ User select and click TomdApplication ▪ Fill in the Login Form with correct username and password	User is logged in and presented with Compose Message Form.	Passed
2.	User compose sms message a. With wrong text format and send to backend	The sms message delivered to the Backend	User type sms message in the message textfield and then clicked the OK Button.	Required information is not retrieve from TomDatabase and the System returns "No Info Alert" to user	Passed
	b. With correct sms text format and send to backend	The sms message delivered to the Backend	User type incurred formatted sms message in the message textfield and then clicked the OK Button.	Required information is retrieved from TomDatabase and sends to user as reply sms message.	Passed
3.	Leaving Compose Message Window and returning to Login Window	User already in compose message window.	User select and click Exit button	User is return to Login Window.	Passed
4.	Login Out of the System	User Logged In and is in the login window.	User select and click the Exit button	User is Logout of the system	Passed

Table 4.2: Functional Testing Results.

S/N	Test	Pre-condition	Test Step	Expected Result	Status
1	FindOffender Function	User login, in the compose message window and has typed and send sms message.	a. User typed in the suspected offender's vehicle registration number, followed by comma and 1 digit number. b. User typed in the suspected offender's vehicle registration number without the comma or 1 digit number or both.	Retrieved the suspect's arrest records from the TomDatabase and sends to user as reply message. The System is unable to retrieve the Suspect's arrest records. It send back reply message to user alerting him of the inadequacy of the text message.	Passed Passed
2.	CheckPenaltyPoints Function	User login, in the compose message window and has typed and send sms message.	a. User typed in the suspected offender's License Number, followed by comma and 2 digit number. b. User typed in the suspected offender's License Number without comma or 2 digit number or both.	Retrieve the suspect's cumulative penalty points from the TomDatabase and sends to user as reply message. The System fail to retrieve the suspect's cumulative penalty points from the TomDatabase and it send back alert message to user as reply informing him about the inadequacy of the sms text message.	Passed Passed
3.	FindVehicleOwner Function	User login, in the compose message window and has typed and send sms message.	a. User typed in a Vehicle Registration Number followed by comma and 3 numeral digit. b. User typed in a Vehicle Registration	Retrieve the Vehicle Owner's Name, Address and Phone Number from the TomDatabase and sends to user as reply message. The System fail to retrieve the	Passed Passed

			Number without comma or 3 numeral digit or both.	Vehicle Owner's Name, Address and Phone Number from the TomDatabase and it send back alert message to user as reply informing him about the inadequacy of the sms text message.	
4.	FindLicenseOwner Function	User login, in the compose message window and has typed and send sms message.	a. User typed in a License Number followed by comma and 4 numeral digit.	Retrieve the License Owner's Name, Address and Phone Number from the TomDatabase and sends to user as reply message.	Passed
			b. User typed in a License Number without comma or 4 numeral digit or both.	The System fail to retrieve the License Owner's Name, Address and Phone Number from the TomDatabase and it send back alert message to user as reply informing him about the inadequacy of the sms text message.	Passed

Table4.3:SMS message snap short of the Evaluation Results.

S/N	SenderNumber	Message Sent	Sending Time	**Reply Message**	DestinationNumber	Reply Message Time
1	+2347055394123	AE0100213,1	09/11/2013 11:36:11	The point is:57	+2347055394123	09/11/2013 11:36:06
2	+2347055394123	AE0100213,1	09/11/2013 11:46:11	The point is:57	+2347055394123	09/11/2013 11:46:30
3	+2348065908729	AE0100213,2	09/11/2013 10:56:22	No Record!	+2348065908729	09/11/2013 10:56:22
4	+2347055394123	AE0100213,1	09/12/2013 11:15:33	The point is:57	+2347055394123	09/12/2013 11:46:12
5	+2347055394123	AA001-ABC,3	09/14/2013 22:32:32	No Record!	+2347055394123	09/14/2013 22:32:48
6	+2348065908729	AA001-ABC,3	09/16/2013 08:48:09	No Record!	+2348065908729	09/16/2013 08:51:58
7	+2348065908729	AA001-ABC,3	09/16/2013 09:27:25	No Record!	+2348065908729	09/16/2013 09:56:57
8	+2348034572357	UX855-KJA,3	09/16/2013 09:53:07	Aigbokhaode Helen Alekedo, Plot C, Flat E 46, New Creation Phase 2/1, Kubwa, Abuja.	+2348034572357	09/16/2013 09:57:00
9	+2348036422468	FV442-ABJ,3	09/16/2013 10:13:21	Aruwa Monday unubi, FRSC Officers' Qtrs., Mararaba1, Karu LGA, Nassarawa State.	+2348036422468	09/16/2013 10:17:14
10	+2348065908729	AE584-KUR,3	09/17/2013 11:30:50	Auwalu Magaji Mohammed, FRSC Officers' Qtrs, Mararaba 1, Karu LGA, Nassarawa State.	+2348065908729	09/17/2013 11:30:55
11	+2348038415405	AA139-AGL,3	09/17/2013 11:33:02	Larai Williams, Dawaki Qtrs, Dutse Alhaji kubwa, Abuja.	+2348038415405	09/17/2013 11:33:02
12	+2347055394123	AA003-ABC	09/17/2013 01:14:47	No Selection, Record not found	+2347055394123	09/17/2013 01:13:08
13	+2348065908729	AA003-ABC, 5	09/17/2013 01:30:20	No Selection, Record not found	+2348065908729	09/17/2013 01:28:40
14	+2348065908729	AA001-ABC, 6	09/18/2013 02:37:57	No Selection, Record not found	+2348065908729	09/18/2013 02:53:12

4.4 LIMITATION OF THE SYSTEM

The System uses GSM modem to sends and receives SMS messages to and from users. A GSM modem has a serious constrain in the sense that it cannot send more than ten (10) SMS messages per minute. This limitation of GSM modem SMS sending rate directly impacted on the system. It implies that the system can only process only 10 requests from users in every minute. This is a serious draw back considering the size of FRSC as an organization with more than one thousand Patrol Teams operating daily. Also, the system's response time is dependent on the GSM Carrier Network Speed, this is another limitation as the system's speed would degrade when the GSM network degraded and vice versa.

4.5 CONCLUSION

This chapter described in detail, the outline of the system processes and the message structure. The threats to the system and the methods used in protecting the system against the perceived threats were explained. Testing of the various systems' components as well as the overall system performance evaluation were also described in the chapter.

CHAPTER FIVE: SUMMARY, RECOMMENDATIONS AND CONCLUSION

5.0 INTRODUCTION

The advent of mobile computing makes mobile computing device like Cell Phone and other hand held devices become like computer such that whatever can be done with computer system could possibly be done with these devices. In order for the devices to behave and supports services like computer, they need to be program just like the computer.

There are several platforms or programming language for developing applications for Mobile Cell Phones and other hand held devices that include J2ME, Android Development Tools, Wireless Markup Language (WML) and so many more. In this research work, we employed XML, J2ME, Android, SQL, PHP and Bach Scripts Programming and technologies to develop a mobile application that is used to tracks Road traffic Offences and access Vehicle and License Owners profiles, such as RTOs penalty-points, name, address, cell phone number etc.

5.1 SUMMARY

The objectives of this Research Work was to develop a system that would enable FRSC Operative track Road Traffic offences have access to Vehicle and License Owner personal profiles such as RTOs accumulated penalty-points, address, GSM Numbers' etc, from the FRSC Operatives' patrol locations via SMS message using mobile cell phone. A system that achieved the objectives was successfully developed, tested and evaluated and was found to be working satisfactorily.

The system consists of two parts; the mobile cell phone application referred to as SMS messaging or Frontend application which was designed and implemented, in two versions, with

J2ME technology using NetBean_IDE 7.1.2, Eclipse and Android Manager 2.2 development environment. The SMS messaging application is installed in Users' Mobile Cell Phone; it is used for composing and sending SMS message to the system's backend application which we referred to as SMS processing application. The system is made such that only Mobile Cell Phones that have the application installed in them and were already configured to be used for the system, can sends an SMS message to the backend application. However, even cell phone that does not have the application can sends message to the backend it has been configured to be used for the system.

The backend comprises of a computer that house the traffic offenders management database (Tomdatabase), the GSM modem which is connected to the PC server's via USB Port, gammurc and smsdrc configuration files interface the modem with computer. Gammu SMS gateway is used for retrieving the received SMS message from the modem inbox, stored it into the inbox table in the smsd database. Once message enters the inbox table, the SMS Processing Application processes the message by decomposing it into various parts and uses the parts to decide the module to execute and corresponding information to extract from the TomDatabase. Message processing is handled by SMS processing application which is a combination of bash, SQL and PHP scripts. The scripts are structured into modules; each module extracts particular information from the TomDatabase depending on the sent message. The information extracted from the TomDatabase is inserted into the smsd database outbox table from where it is sends to user as reply message. The system has been constructed and is working successfully.

5.2 CONCLUSION

Sequel to the testing and evaluation of the developed system, it is clear that the constructed

system achieved all the stated objectives of the research and it has provided the answer to the questions we set out in chapter one. Evidently, a system for tracking road traffic offences, extracting of vehicle and license owner profiles via SMS message was successfully developed. The tracking and profiling is done by sending a SMS message from FRSC Operatives location to the GSM modem attached to the PC Server for checking information on traffic related issues such as finding penalty points of a traffic offender, checking and verifying the validity of driving license and its genuineness as well as obtaining real time information on vehicles' and driving license owners.

In this system, communication between the user device and the PC Server is via GSM Network (Globacom), while communication between the PC Server and the GSM Modem is via gammu SMS gateway, specifically made possible by the gammurc and smsdrc files.

Messaging processing done using SQL and PHP Scripts structured which executed by a Bash scripts program. Message extracting is done based on modules, where each module is responsible for extracting particular information required from the TomDatabase. Information extracted is based on the content of message sent by users.

Message has been structured in a specialize format, just like a command-like format. For a message to satisfied being process by the system and in order for the correct module to be executed, it must complied with the acceptable structure or compose according to the acceptable format. When user sends message, it is retrieved from the modem's inbox and stored in the smsd database. From there, the message is extracted and decomposes into three parts; each part has its specified function in determining the module to be executed, the information to be extract and the destination to send the reply.

The system has been constructed, tested and found to be working satisfactorily. We recommend

it for used by the intended organization which is the Federal Road Safety Corps, Nigeria.

5.3 LIMITATIONS

An SMS Based Road Traffic Offence Tracking and Profiling System (Tracker) had been successfully built and it is working accordingly. However, system's performance as per number of requests it process per minute is strictly defendant on the SMS sending rate of the GSM Modem per minute which is between 6 to 10 sms/minute. This has constrain the system performance to process only 10 requests per minute maximally.

Moreover, the efficiency of the Tracker is defendant on the GSM Network such that the faster the network, the faster the tracker replies users' requests.

5.4 RECOMMENDATIONS FOR FRSC PROCESSES IMPROVEMENT

Based on our study of the FRSC Functional Process and in order for the System to perform optimally, we recommend as follows;

a. FRSC system of formalizing Vehicle Registration and Transfer of vehicle Ownership should be de-centralize by introduction of One Stop Shop orientated Service, establishing mechanism for licensing Vehicle Dealers nationwide in order to control vehicles' sale and facilitates quick vehicles ownership transfer.
b. FRSC should also ensure proper and timely archiving of convicted road traffic offenders records from the main offenders_register table in order to avoid being charged and prosecuted more than one for the same offence.

5.5 RECOMMENDATION FOR FURTHER WORK

This research work is concerned with access to the textual information of Road Traffic Offences, Vehicle and License Owner information which is stored in the TomDatabase. However, equally stored in the database is the vehicle and license owner biometric information and the database is stand alone database system. In order to improve the system, we recommend as follows;

a. Improvement of the system to retrieval biometric information of RTOs, vehicle and license owner information using Multi-media Message Service (MMS) and provision for use of Scratch Card.
b. Improvement of the TomDatabase from Standalone to Distributed System.

REFERENCES

AbdWahab, H. M., Abdullahi, N., Ayob, J., & Herdawati, A. K. (2010). GSM Based Control System for Smart Home Appliances. *Journal of Convergence Information Technology* , Volume5, Number 1.

Alomary, A. (2007). A System for Controlling Remote Devices through the Internet, SMS and Telephone Network. (pp. 19-27). International Conference on Communication and Computer (CCCP'07).

Barroon, I. A. (2011). *Remote Monitoring and Management of Servers from Mobile Devices. M.Sc Thesis.* Zaria: Ahmadu Bello University, Zaria, Nigeria.

Bhalla, M. R., & Bhalla, A. V. (2010). Generation of Mobile Wireless Technology: A survey. *International Journal of Computer Applications* , 26-32.

Billy, D. (2009). *Development of Wireless Application using Java 2 Micro Edition Platform.* Sun Microsystem Inc., www.billydey.com.

Blaha, R., Premerlani, W., & Rumbaugh, J. (1998). *Relational Database Desigh Using an Object Orientated Methodology.* Communication of the ACM, Vol. 3, No. 4.

Ciubotaru, P. B., Chiciudean, D., Cioarga, R., & Stanescu, D. (2006). *Wireless Solution for Telemetry in Civil Equipment and Infrastructure Monitoring.* Romainan-Hungarian Joint Symposium on Applied Computational Intelligence (SACI). http:/www.bmf.hu/conferences/saci2006/ciubotarus.pdf.

Das, C., Sanaulla, M., Sarowar, H., & Hassan, M. (2008). *Development of a Cell Phone Based Remote Control: An Effective sqitching for Controlling Home and Office Appliances.* International Journal of Electrical and Computer Sciences (IJECS), Vol 9, No. 10.

Delgado, A., Picking, R., & Grout, R. (2006). Remote Controlled Home Automation Systems with different Technologies. *6th International Networl Conference (ICN 2006)* (pp. PP 357-366). University of Playmouth: Proceeding of the 6th International Networl Conference (ICN 2006),.

Developer, W. (n.d.). *Build a Smart J2ME Mobile Application, part 1.* Retrieved December 2011, from www.ibm.com.

Developer's, G. (2011). *Oracle Java Micro Edition Development Kits*. Retrieved December 2011

DeveloperWorks. (n.d.). *J2ME: Step by Step.* Retrieved August 15, 2010, from ibm.com/developWorks: ibm.com/developWorks

Dey, A., & Abroul, G. (2000). CyberMinder: A Context-aware System for Supporting Reminders. *Symposium on Handheld and Ubiqitious Computing* (pp. 187-199). Spring-Verlog: Second Symposium on Handheld and Ubiqitious Computing, LNCS 1927,.

FRSC. (2010). *Report on Road Traffic Crashes (RTC) in Nigeria, 2007-June 2010.* Abuja: Corps Standadization Office, ISBN:13978-978-482-74-0-9.

Garzeli, A. I. (n.d.). *Cellular Security.* Retrieved December 2011, from http:/cse.wustl.edu: http:/cse.wustl.edu/jain/cse57406/ftp/CellularSecurity/index.html

Google. (n.d.). *Wikipedia* . Retrieved October 2011, from ww: www.wikipedia.org

Jan Lucenius, J. S. (2009). *Implementing Mobile Access to Heteregeneous Home Environment.* Finland: VTT Technical Research Centre of Finland.

Jawakar, N., Ahmed, V., Ladhake, S., & Thakare, R. (2008). Micro-controller Based Remote Monitoring using Mobile Through Spoken commands. *Journal of Networks, 3(2). , 3(2)* (58-63), 58-63.

Jianye, L., & Jianku, Y. (2011). Research on Development of Android application. Fourth International Conference on Intelligence Networks and Intelligence Systems.

Kana, D., Obiniyi, A., & Ajiboye, O. (2009). Deployment of Information Technology in Vehicle Control and Monitoring in Nigeria. *Journal of Sustainable Development, Federal University of Technology, Abeokuta. , 6* (1), pp 21-27.

Ketaki, D., & Priyanka, L. (2010). *Remotely Controlling a Desktop Machine using Mobile Phone CS6235.* Georgia: Class Project, Georgia Institute of Technology.

Morris, P. J. (2005). *Relational Database Design and Implementation for Biodiversity Informatics.* Philadelphia, USA: The Academy of Natural Sciences 1900 Ben Franklin Parkway, Philadelphia, PA19103.

Murphy, M. (2008). *Mobile Based Primary Health Care System for Rural India.* W3C Workshop on the Role of Mobile Technologies in Fostering Social Development.

Ozeki, N. (2010). Retrieved September 2012, from Ozeki NG: http://www.ozekisms.com

Ratnesh, L. (2009). SMS Based Home Control System: An Inexpensive Approach.

Roldan, E., Monton, E., & Guillen, S. (2008). *Programming Mobile Device with J2ME.* ITACA Institute.

Roth, J. (2008). *Flexible Positioning for Location-Based Service.* Hagen, Germany: University of Hagen.

Salsano, S. (2008). *Final Architecture Specification.* Information System, Project Title; SMS, Project No 034620.

Security Engineering Report Group. (2010). *Technical Report: Analysis on Android Application Framework and Existing Security Architecture.* Peshawar, Pakistan: Institute of Management Sciences, http:/serg.imsciences.edu.pk.

Sommaire, 2. (2010). *GSM/SMS GUIDE.* Copyright (c) 2010XMcom, www.xmcom.net.

Van der Werff, M. J., Xu, W., & Gu, X. i. (2004). *Activation of Home Automation System via Mobile Technology.* Email: vanderwerff@clear.net.nz, w.l.xu@massy.ac.nz and x.gui@massey.ac.nz.

Wemmy, J. R., David, T., Lee, N. k., & Eric, P. (2003). *Case Study for Using an object Relational Paradigm in Building Web Database Application.*

APPENDICES

APPENDIX A: SMS messaging Application Source Codes.

a. TomdApplication.java

```
import javax.microedition.midlet.MIDlet;
import javax.microedition.lcdui.*;
import javax.microedition.io.*;
import javax.wireless.messaging.*;
import java.lang.String;

public class TomdApplication extends MIDlet implements CommandListener{
        private Display display;
        private TextField userName,password,toWhom,message;

        public Form form,compose;
        private Command login,cancel,send,exit;
        private Image img, imge, img2;
 private Alert alert;//sendingMessageAlert;
 MessageConnection clientConn;


public TomdApplication() {
        form = new Form("Sign in");
        userName = new TextField("LoginID:", "", 30, TextField.ANY);
        password = new TextField("Password:", "", 30, TextField.PASSWORD);
        cancel = new Command("Cancel", Command.CANCEL, 2);
        login = new Command("Login", Command.OK, 2);

compose=new Form("Compose Message");
toWhom=new TextField("To","+2348059628173",15,TextField.UNEDITABLE);
message=new TextField("Message","",600,TextField.ANY);
send=new Command("Send",Command.BACK,0);
exit=new Command("Exit",Command.SCREEN,5);
compose.append(toWhom);
compose.append(message);
compose.addCommand(send);
compose.addCommand(exit);
compose.setCommandListener(this);

        try{
                img = Image.createImage("/server.gif");
                imge = Image.createImage("/fail.gif");
                img2 = Image.createImage("/success.gif");
```

```
        }catch(Exception e){
                System.out.println(e.getMessage());
        }
}

public void startApp() {
        display = Display.getDisplay(this);
        try{form.append(img);//form.append(img);
        }catch(Exception e){}
        form.append(userName);
        form.append(password);
        form.addCommand(cancel);
        form.addCommand(login);
        form.setCommandListener(this);
        display.setCurrent(form);
}
public void pauseApp() {}

public void destroyApp(boolean unconditional) {
        notifyDestroyed();
}

public void validateUser(String nam, String passwd) {
        if (nam.equals("123") && passwd.equals("123")) {
                showMsg();
        } else {
                tryAgain();
        }
}

public void showMsg() {
        Alert success = new Alert("Login Successfully", "          Your Login
        Process is completed!", img2, AlertType.INFO);
        success.setImage(img2);
        userName.setString("");
        password.setString("");
        display.setCurrent(success, compose);
}
public void tryAgain() {
        Alert error = new Alert("Login Incorrect", "              Please try
        again", imge, AlertType.ERROR);
        //error.setTimeout(900);
        error.setImage(imge);
        userName.setString("");
        password.setString("");
        display.setCurrent(error, form);
```

```
}

public void commandAction(Command cmd, Displayable d) {
        if(cmd==cancel) {
                destroyApp(true);
                }
        else if(cmd==exit) {
    destroyApp(false);
}
else if(cmd==login) {

    String name=userName.getString();
    String passw=password.getString();
validateUser(name,passw);
}
 else if(cmd==send)
                        {
    String mno=toWhom.getString();
    String msg=message.getString();
    if(mno.equals("")) {
        alert = new Alert("Alert");
        alert.setString("Enter Mobile Number!!!");
        alert.setTimeout(2000);
        display.setCurrent(alert);
                                        }
    else
                                {
        try {
            clientConn=(MessageConnection)Connector.open("sms://"+mno);
                                                }
        catch(Exception e) {
            alert = new Alert("Alert");
            alert.setString("Unable to connect to Station because of network
        problem");
            alert.setTimeout(2000);
            display.setCurrent(alert);
                                                }
        try {
        TextMessage textmessage = (TextMessage)
        clientConn.newMessage(MessageConnection.TEXT_MESSAGE);
            textmessage.setAddress("sms://"+mno);
            textmessage.setPayloadText(msg);
            clientConn.send(textmessage);
            //String statusMessage = "Sending message to " + mno + "...";
                                //sendingMessageAlert.setString(statusMessage);
                                                }
```

```
            catch(Exception e)
                                                {
                Alert alert=new Alert("Alert","",null,AlertType.INFO);
                alert.setTimeout(Alert.FOREVER);
                alert.setString("Unable to send");
                display.setCurrent(alert);
                                                }
                                        }
                                }


            }
}
```

b. TomdApp.Java

```
package com.example.tomdapp;

import android.app.Activity;
import android.os.Bundle;
import android.telephony.SmsManager;
import android.view.Gravity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class TomdAppActivity extends Activity {

 //EditText no;
 EditText mes;
 Button send;
   /** Called when the activity is first created. */
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_sms);

      //no=(EditText)findViewById(R.id.phno);
      mes=(EditText)findViewById(R.id.message);
      mes.setGravity(Gravity.TOP);
      send=(Button)findViewById(R.id.sendsms);
```

```
        send.setOnClickListener(new OnClickListener() {

      @Override
      public void onClick(View v) {
       // TODO Auto-generated method stub
       String phNo = "08059628173"; //=no.getText().toString();
       String message=mes.getText().toString();
           //if empty send alert

       if (message.matches(""))
       {
              Toast.makeText(getApplicationContext(), "Field cannot be empty",
              Toast.LENGTH_LONG).show();
       }
       else{
              SmsManager sms=SmsManager.getDefault();
               sms.sendTextMessage(phNo, null, message, null,null);
               Toast.makeText(getApplicationContext(), "message sent successfully",
              Toast.LENGTH_LONG).show();
               //no.setText("");
               mes.setText("");
      }
      }
     });
       }
}
```

c. R.java File

```
/* AUTO-GENERATED FILE.  DO NOT MODIFY.

 *

 * This class was automatically generated by the

 * aapt tool from the resource data it found.  It

 * should not be modified by hand.

 */


package com.example.sendingsms;
```

```
public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int bg=0x7f020000;
        public static final int bg1=0x7f020001;
        public static final int ic_launcher=0x7f020002;
        public static final int logo=0x7f020003;
        public static final int logo1=0x7f020004;
        public static final int server=0x7f020005;
    }
    public static final class id {
        public static final int menu_settings=0x7f070003;
        public static final int message=0x7f070001;
        public static final int sendsms=0x7f070002;
        public static final int textView1=0x7f070000;
    }
    public static final class layout {
        public static final int activity_sms=0x7f030000;
    }
    public static final class menu {
        public static final int activity_sms=0x7f060000;
    }
    public static final class string {
        public static final int app_name=0x7f040000;
        public static final int hello_world=0x7f040001;
        public static final int menu_settings=0x7f040002;
```

```
    }
    public static final class style {
        /**
        Base application theme, dependent on API level. This theme is replaced
        by AppBaseTheme from res/values-vXX/styles.xml on newer devices.

            Theme customizations available in newer API levels can go in
            res/values-vXX/styles.xml, while customizations related to
            backward-compatibility can go here.

        Base application theme for API 11+. This theme completely replaces
        AppBaseTheme from res/values/styles.xml on API 11+ devices.

API 11 theme customizations can go here.

        Base application theme for API 14+. This theme completely replaces
        AppBaseTheme from BOTH res/values/styles.xml and
        res/values-v11/styles.xml on API 14+ devices.

API 14 theme customizations can go here.
         */
        public static final int AppBaseTheme=0x7f050000;
        /**  Application theme.
All customizations that are NOT specific to a particular API-level can go here.
         */
        public static final int AppTheme=0x7f050001;
    }
```

}

d. Activity_xml File

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:orientation="vertical" android:background="@drawable/bg" >
 <!-- android:background="#ccffff" -->
 <!--  <TextView
       android:layout_width="fill_parent"
       android:layout_height="wrap_content"
       android:text="Mobile Number"
       android:textColor="#000000"
       android:textSize="20dp"/>

   <EditText
      android:id="@+id/phno"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:ems="10"
      android:inputType="number" />

   -->

   <TextView
      android:id="@+id/textView1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Message"
       android:textColor="#000000"
       android:textSize="20dp"/>

   <EditText
      android:id="@+id/message"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:ems="10" />

   <Button
      android:id="@+id/sendsms"
      android:layout_width="150dp"
      android:layout_height="wrap_content"
```

```
        android:text="Send"
        android:layout_gravity="center"
         android:layout_marginTop="10dp"
          android:textSize="20dp"/>

</LinearLayout>
```

e. AndroidManifest.xml File

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="com.example.sendingsms"
   android:versionCode="1"
   android:versionName="1.0" >
   <uses-sdk
      android:minSdkVersion="8"
      android:targetSdkVersion="8" />
    <uses-permission android:name="android.permission.SEND_SMS"/>

   <application
      android:allowBackup="true"
      android:icon="@drawable/logo"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >
      <activity
         android:name="com.example.sendingsms.SMSActivity"
         android:label="@string/app_name" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
   </application>

</manifest>
```

APPENDIX B: SMS Message Processing Application Source Codes.

a. SearchInfo.bs

b. ExtractUserMsg.sql File

select SenderNumber,TextDecoded INTO OUTFILE 'MyMsg.txt' FIELDS TERMINATED BY ',' ENCLOSED BY '' ESCAPED BY'\t' from inbox order by ID DESC limit 1;

c. LoadUserMsg.sql

LOAD DATA INFILE 'MyMsg.txt' INTO TABLE stopover
FIELDS TERMINATED BY ',' LINES TERMINATED BY '\r\n';

d. ExtractUserReply.php File

```
#!/usr/var/php -q
<?php
$host = "Localhost";
$user = "root";
$Password = "";
$databasename = "smsd";
/* connect to database*/
$con = @mysql_connect($host,$user,$password) or die(mysql_error());
@mysql_select_db($databasename) or die(mysql_error());
/* set the query*/
$query = "SELECT SenderNumber, msgtext, selectvar FROM stopover order by ID DESC limit 1";
/* execute the query*/
$result = mysql_query($query) or die(mysql_error());
/* fetch the result into the variables*/
$mysendno = mysql_result($result, 0, "SenderNumber");
$mytext = mysql_result($result, 0, "msgtext");
$myselectvar = mysql_result($result, 0, "selectvar");
$mytext = trim($mytext);
echo $mysendno." t". $mytext."t ". $myselectvar;
mysql_free_result($result);
if($myselectvar == ""){
        $insertext = "No Selection, record not found!";
}
else{
        function ExtractInfo($mytext,$myselectvar){
```

```
$infotext = "";
if($myselectvar == "1"){
        sql = "SELECT SUM(penalty_points) AS points FROM
        offence_register WHERE LicenceNo='$mytext'";
        //echo $sql;
}
elseif($myselectvar == "2"){
        $sql = "SELECT COUNT(*) AS points FROM
        offenders_register WHERE VehicleNo = '$mytext'";
}
elseif($myselectvar == "3"){
$sql = "select CONCAT(owner_Name,', ',Address) AS points from
owner_register where owner_ID = (select Owner_ID from
vehicle_register where vehicleNo = '$mytext')";
//echo $sql;
}

$result = mysql_query($sql) or die(mysql_error());
//$row=mysql_fetch_array($result1);
$infotext = mysql_result($result, 0, "points");
//$infotext = mysql_result($result1, 0, "points");
//echo $infotext."iner";
//mysql_free_result($result1);
return $infotext;
}
/* Call the ExtractInfo function and assign its value to variable*/
$replytext = ExtractInfo($mytext,$myselectvar);
echo $replytext;
if($replytext == ""){
        $insertext = "No Record!";
}
else{
        if($myselectvar == "1"){

                $insertext = "The Offender Penalty Points is:" . $replytext;
        }
        else if($myselectvar == "2"){
                $insertext = "The Offender was arrested:" . $replytext .
"times";
        }
        else if($myselectvar == "3"){
                $insertext = "The Detail of " . $mytext . "is:" . $replytext;

        }
        else if($myselectvar == "4"){
                $insertext = "The Detail of " . $mytext . "is:" . $replytext;
```

```
                    }
                }
            }
            /*  Inser Variables into outbox*/
            $query2 = "INSERT INTO outbox SET TextDecoded = '$insertext',
            DestinationNumber = '$mysendno'";
            echo $query2;
            @mysql_query($query2) or die();
            /* Close the Database Connection*/
            @mysql_close($con);
?>
```

e. MyStartup.bs

```
#!/bin/bash
/opt/lampp/bin/mysql --user=root --password="" --database=smsd < /usr/ExtractUserMsg.sql
/opt/lampp/bin/mysql --user=root --password="" --database=smsd < /usr/LoadUserMsg.sql
/opt/lampp/bin/php -f /usr/ExtractReplyMsg2.php
rm /opt/lampp/var/mysql/smsd/MyMsg.txt
```

APPENDIX C: Gammu Configuration Files.

a. gammurc

gammu]

port = /dev/ttyUSB0

connection = at115200

b. smsdrc

[gammu]

port = /dev/ttyUSB0

connection = at115200

[smsd]

Service = mysql

Driver = native_mysql

PIN = 0000

LogFile = /home/smsdlog1

ReceiveFrequency = 2

Commtimeout = 2

User = root

Password =

PC = 127.0.0.1

Database = smsd

debuglevel = 255

RunOnReceive = /usr/SearchInfo2.sh

#inboxpath = /var/spool/gammu/inbox/

#outboxpath = /var/spool/gammu/outbox/

#sentsmspath = /var/spool/gammu/sent/

#errorsmspath = /var/spool/gammu/error/

#inboxformat = standard

#transmitformat = auto