SMS Based Pull Campaign Processing System

Suresh Reddy Chenreddy
Akbar Ali Baig Mirza

September 2007
Author, Programme and Year/Författare, program och år:
Akbar Ali Baig Mirza, Digital Communication Design 2005
Suresh Reddy ChenReddy, Digital Communication Design 2005

Instructor/Handledare:
K.Shashank Reddy, Ample Technologies
Daniel Einarson, HKr

Examination/Examen:
This graduation work is a part of the requirements for a Degree of Master of Digital Communication Design (as specified in the English translation).

Detta examensarbete ingår i examenskraven för Magisterexamen med inriktning digital kommunikationsdesign

English Title:
SMS Based Pull Campaign Processing System

Abstract:
SMS based pull campaign processing system is a new model of directory service provided in mobile infrastructure basically to enable SMS based directory services Mobile directory services is a brand new phenomenon, which hold strong promises to become the best targeting advertising medium delivering new means of services to users with message other than traditional channels (e.g.: print, television, radio and email etc). The main purpose of this system is to bring customers and business organizations closer. It enables SMS based directory services. It not only acts as directory services but also as an advertising medium between the user and the business organizations.

Language/Språk:
English

Approved by/Godkänd av:

Eric Chen
Examiner/Examinator
Acknowledgement

The satisfaction that accompanies the successful completion of any task would be complete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts with success.

We are greatly thankful to “Mr. K.Shashank” Project Manager of Ample technologies, Who have helped us to get this work out of the door.

We are very much thankful to our Examiner “Mr. Eric Chen” and Instructor “Mr. Daniel Einarson” for Instructing and examining the Project.

We are also very much thankful to all staff members of Kristianstad University.
SUMMARY

The First chapter tells about the project description and which include the Software & Hardware requirements. The Second chapter tells, about the SRS and how it is helpful to our project and the purpose of the SRS, scope that is where we have to implement our applications and we also having the definitions, acronyms, and abbreviations.

The Third chapter tells, about the Analysis phase which includes what are the services provided by the existing system, and our current using services [Directory Services] is one of the better solution the existing system, and the proposed system will describe about the solution to our Directory Services.

The Forth chapter tells, about the specification of the system, which include the modules which is represented in “UML” and also tells the environment of the system. The Fifth chapter tells, about the Design Phase which includes the Class diagram, use case diagram, activity diagram, sequence diagrams are drawn in UML, the system specifications are input for this Design Phase. The Sixth chapter tells about the database which include tables

The Seventh chapter tells, about the implementation phase which includes-what ever we design in the Design Phase, using that tables and other information we have to develop the code and implementation of this code and Screen Shots. The Eighth chapter tells, about testing phase which include what ever we implement the code in previous phase we have to testing that code using various testing strategies like unit test, validation test, integration test, system test & reliability test and the testing techniques are Black Box (or) White Box testing.

The Ninth & Tenth chapters are tells, about the what we have include for our application and what are the future aspects of our application, references for developing our application and conclusion.

The entire project is the procedure for building the Directory Services through SMS. Then we will develop the efficient project in the further phases of SRS as we mention the above.
# Contents

Acknowledgement..................................................................................3

Summary.................................................................................................4

1. Introduction.........................................................................................7
   1.1 Purpose.........................................................................................7
   1.2 Scope.............................................................................................7
   1.3 Definition and Acronyms and Abbreviations.................................9

2. System Analysis..................................................................................10
   2.1 Existing System...........................................................................10
   2.2 Problem Statement......................................................................10
   2.3 Proposed System..........................................................................11

3. System Specification..........................................................................14
   3.1 Architecture Diagram.................................................................14
   3.2 Network Diagram.........................................................................15
   3.3 System Environment.....................................................................16
   3.4 SMSService Implementation.......................................................24
   3.5 Dataflow Diagram........................................................................25

4. Detail Design......................................................................................27
   4.1 UML Diagrams............................................................................27
      4.1.1 Class Diagram.......................................................................27
      4.1.2 Use Case Diagram.................................................................29
      4.1.3 Activity Diagram....................................................................32
      4.1.4 Sequence Diagram...............................................................40

5. Tables.................................................................................................46

6. Implementation....................................................................................50
   6.1 Implementation Procedure..........................................................50
   6.2 Further System.............................................................................56
   6.3 Problem and Solution...................................................................56
   6.4 Screen Shots...............................................................................57
7. Testing.........................................................................................71
  7.1 Unit Testing..............................................................................71
  7.2 Integration Testing.................................................................71
  7.3 Test Suite ................................................................................72
  7.4 Test Cases.................................................................................72
    7.4.1 Functional Test Case..........................................................72
    7.4.2 Structural Test Cases.........................................................72
    7.4.3 Integration Test Cases.......................................................72
  7.5 Implementing Test Cases........................................................73

8. Conclusion.......................................................................................75

9. Scope For expansion .........................................................................75

10. Bibliography......................................................................................76
1. Introductions

The introduction of the **Software Requirements Specification (SRS)** should provide an overview of the entire **SRS**. It should include the purpose, scope, definitions, acronyms, abbreviations, references, and overview of the **SRS**.

The Software Requirements Specification (**SRS**) captures the complete software requirements for the system, or a portion of the system and it provides a typical outline for our project using only traditional natural-language style requirements – with no use-case modeling. It captures all requirements in a single document, with applicable sections inserted from the Supplementary Specifications (which would no longer be needed). For a template of an **SRS** using use-case modeling, which consists of a package containing **Use-Cases** of the use-case model and applicable Supplementary Specifications and other supporting information, and This **SRS** is provide the use with the **Rational Unified Process**

This **SRS** is the fundamental process of understanding why an information system should be built. The **SRS** will also determine how the project team will go about building the information system.

1.1 Purpose:

The **SRS** plays a major role in the development of the application which satisfies the customer requirements and maintain the **Quality of service (QOS)** to the customer.

The present role of **SRS** in the context of the “**SMS Based Pull Campaign Processing System**” is to specify the functionality of the application, scope & all the pre-requisites that are necessary for the development of the system, which could be the identification of the functional & non-functional requirements. The **SRS** also identifies the starting and ending of the various phases like Analysis, Design, Implementation and maintenance phases. The **SRS** also defines the standards that are followed gives the reader to a clear idea of what is going on and which is where in the application.

The “**SMS Based Pull Campaign Processing System**” is a unique of its services and in this we are using the directory services to develop our project. This **SRS** gives the basic skeleton for the development of our project.

1.2 Scope:

The scope in the **SRS** gives you the idea of to which extent the present development of project is giving to be implemented and for whose concerns. One project is “**SMS Based Pull Campaign Processing System**” which is one of the several services in the mobile communication and it is on of the best, cheap and efficient means of the communication that is **SMS**.

Now a days **SMS** can be used for the many purpose, and it is easy to communicate with other person. In today’s Metro’s there is a great demand for local information. **For example** people would like to know information about different types of events, and locations where the events are happening & the description of those events, and also people who are interested in various discounts, entertainment events etc.
In olden days to send the message we are using the **pagers** but using pagers we are send a textual message only, in now a days we are sending not only text also we are sending images, pictures, ring tones, movie clips and also using for other services.

Sending a **SMS** which is having text it contains only **160** characters per a page and it is not allow the **formatted** text that is which is bold, italic and underlined text.

**SMS** can also used for several problems and solutions. As the numbers of mobile users are increasing exponentially, the scope of the usage of **SMS** for solving the problem (or) for providing an attentive solution to the problem is a new and efficient idea. **SMS** can be used for various applications with the same Base line.

**SMS - Where, When, and Why**

Europe designed GSM (Global System for Mobile communications) from the ground up as a system for analog voice with modulated digital capabilities built in.

SMS took full advantage of these digital data capabilities to send short bursts of text-based messages across the GSM (and later TDMA and CDMA) networks.

SMS provides a point-to-point (and broadcast-to-point) mechanism for transmitting short messages (up to 160 text characters) from wireless handsets.

SMS was initially bundled into GSM as a sop to network carriers, promising them a way to increase marginally the carrying capacity of a network. SMS messages traverse the less-populated, higher-frequency bands of the network.

**Some of the applications that are run on SMS are**

- SMS for voting,
- SMS for Email and paging internetworking
- SMS for enquiring,
- SMS for polling,
- SMS for playing games,
- SMS for servers,
- SMS for registration, And

Our current solution provides **Directory services** with the help of **SMS** as the communication mechanism. The basic scope of this application is to provide **Directory Services** to any user by requesting a simple key code. This can also be implemented based on based **Directory Services**.

To extend our application into global that is not based on the location the user can access any where in the world it gives the information.
1.3 Definitions, Acronyms and Abbreviations:

All the definitions, Acronyms and Abbreviations used in the entire application are described in this section.

**Publisher:**
The person who publishes messages about his business/service is called **Publisher**.

**Administrator:**
The person who manages the whole application is called Administrator.

**Acronyms:**

- SRS : Software Requirement Specification
- SMS : Short Messaging Service.
- SMPP : Simple Message Peer-to-Peer.
- SIM : Subscriber Identity Module.
- SMSC : Short Messaging Service Controller.
- GSM : Global System for Mobile Communication.
- HTTP : Hyper Text Transfer Protocol.
- SDS : System Design Specification
- DB : Database
- JSMS : Java Short Messaging Service
- Modem : Modulation and Demodulation
2. SYSTEM ANALYSIS

2.1 Existing System:

Campaigning plays a vital role in the development of any business organization and it helps the consumer community a lot by providing a scope to decide while purchasing the goods. In the present system, campaigning is mainly based on the traditional means like newspapers, radio, television, and vinyl banners. These modes of campaigning involve a lot of investment by the organization to reach the consumers regarding their business. But there is no guarantee of reaching the every consumer regarding business.

2.2 Problem Statement:

The main purpose of this system is to bring customers and business organizations closer. There are several solutions for Directory Services such as Yellow Pages, Internet Based Directory Services, etc. But our “SMS Based Pull Campaign Processing System” provides a different way of solution to the Directory Services and much more easy to use by any one in any place/ time and any where. Directory Services are the process of searching data/information. It not only acts as directory services but also as an advertising medium between the user and the business organizations. This is not a solution to the existing system. This is a newly proposed System. This proposed System is an end-to-end and Complete Solution to run.

Now a days Mobile users are growing tremendously, for them a solution like this is an adoptable one. This system is to introduce highly reliable and sophisticated strategies in campaigning and in providing better beneficial to both customer and business organization. This has a better scope in implementing advancements and compatible for future developments. SMSme server does the maintenance and administration of the entire process of campaign. To develop software in such a way that is flexible and does not wear out. It should be developed in such a way that consumer can understand the system and use it efficiently. We use the software, which is portable, and compatible i.e. it can be run any where. So we can use the system in better way.

Choosing sms is more advantageous:

- Relatively Low cost
- Receiving/sending messages at any possibilities
- Easy and Comfortable to use
- We cannot neglect the message
- Quick response
- Reduced a lot of time and manual work.
- Any time Any where.
Main aim of this project is to build Directory Services through an SMS. User sends SMS with a stipulated Message been typed with a specific code. Here SMS is a best way of communication.

First, Admin has to Login. Publisher has to submit Register details with SMS Service Code and Login with Username and Password. Username must be validate. Admin has to activate / Inactivate Registered Publishers. Publisher can’t access with SMS portal unless Admin authorizes him. Mobile User sends an SMS with Service Code to Server. It validates Service Code and processes User’s request then sends it with Service Code to User. One message can allow up to 160 characters.

The software should be developed in such a way that the consumer should understand the operations which are in campaigning process and also the campaign codes should also be clearly given. Providing SMS and sending reply to the consumer should be done in a simple and optimistic way. The application should be developed in such a way that it should be done within given time and with minimum cost. As, now days, everyone is having mobile we don’t require much of the paper correspondence for communications between administrator, publisher and consumer.

2.3 Proposed System:

The main aim of the project is to access directory service over the existing mobile phone whenever a user invokes a particular application. It’s just like services provided by the yellow pages as internet, but in a new medium i.e. through SMS

Till now mobiles were used for only voice based calls as for SMS’s. This application adds one more dimension to the existing mobiles. The main objective of this SMS based pull campaign processing system is to provide directory services through new medium of SMS’s other than the regular ones like the yellow pages as internet. This application provides directory service only when the user himself invokes an application or requests for the particular services to be granted.

SMS based pull campaign system is a new model of directory services which enables mobile users to search for specific information. SMS based DIRECTORY service Can be delivered as Cross Carrier and Services only on basis of customer interest. It includes hosting services, managing the services and provisioning service information to the mobile users along with promotions, coupons, discounts etc.

We are building different way of creating Directory Services through an SMS. For this context, most of the organizations proceed with an “APPLICATION MODEL”. That means, developing an application with GSM-MODEM.

GSM modem is one of the most exciting and innovative electronic products ever developed. With it you can stay in contact with your office, your home, emergency services, and others, wherever service is provided. GSM modem is used for more no. of operators
**Falcom TWIST GSM-Modem**

GSM is a newer radio frequency («RF») technology than the current FM technology that has been used for radio communications for decades.

With the Falcom TWIST GSM-Modem, connections can be made to the GSM 900/1800 (1900) mobile radio network via the computer and the supplied GSM-Communicator.

After connecting the modem, the following telecommunication services are available:

- making and receiving voice calls
- sending and receiving short messages

All technical components required to establish the above communication links are integrated in the Falcom TWIST GSM-Modem.

The serial port is connected to the computer by the data cable. Power is supplied to the modem from the power supply socket. The internal SIM card reader serves for holding the SIM card given to you by your provider, and it is closed by a cover cap.

To make telephone calls you require suitable application software or We can use the application software contained in the GSM-Communicator.

More and more organizations are realizing the benefits of SMS, and using it as a new media within their traditional marketing communication strategies the reason for this increase in the business usage is clear – SMS is an extremely cost–effective, high-response–rate vehicle, which can help to acquire and retain consumers, sell and promote products, drive loyalty, and reinforce branding efforts.
Mobile directory services is a brand new phenomenon, which hold strong promises to become the best targeting advertising medium delivering new means of services to users with message other than traditional channels (e.g.: print, television, radio and email etc.) The main purpose of this system is to bring customers and business organizations closer. It enables SMS based directory services. It not only acts as directory services but also as an advertising medium between the user and the business organizations.

This Includes:

- Hosting Services
- Managing the Services
- Provisioning Service to the Mobile User

How this project/Application is different from other system:

- Cross Carrier Solution
- It Enables the Directory Services
- It is used to build a gap between a customer & Organisation
- It is not a generic Solution to Business but is a massive application to enhance the Business profits
- Before this System we never found a Directory based SMS Service.
3. SYSTEM SPECIFICATION

3.1 Architecture Diagram:

![Architecture Diagram]

- **User Interface**
  - Mobile Device Function
  - MESSAGES
- **Request Creation**
  - Campaign Code
- **Mobile Device**
- **Publisher Portal**
  - Web Client
- **Publisher Portal**
  - Web Client
- **SMSme Admin**
  - Web Client
- **SMSme Service Provider**
  - SMS Request Processor
  - Campaign Code Validator
  - Campaign msg Sender
  - Service Access Log
- **User Authentication**
  - Publisher Register
  - Campaign Manager
  - Campaign Access Reporter
- **User Authentication**
  - Campaign Category Manager
  - Publisher Manager
  - Campaign Controller
  - Reporting Engine
- **J2EE Framework**
- **DB Connection Manager**
- **DB**
3.2 Network Diagram:

- Wireless Operator Network
- SMS Gateway
- Receiving Service Response
- Application Server
- Database
- User Sending SMS
- Publishing Services
3.3 System Environment

The System Environment of the SMS based Pull Campaign Processing System are as follows:

**Hardware Environment:**

- **Processor**: Pentium 4
- **Hard disk**: 80 GB
- **RAM**: 256 MB

Modem or network card to access the Internet.

**Software Environment:**

- **Operating System**: Windows 2000
- **Database**: Oracle 9i
- **Languages Required**: Servlets, JSP, JDBC, HTML & JSMS-API
- **Web Services Used**: Tomcat 5.5
- **Software’s Used**: Jdk1.5 and Internet Explorer 5.0

The front end of the Pull Campaign Processing System was developed using J2EE (Servlet, JSP) and the back end using Oracle 9i.

**TECHNOLOGIES USED**

The following technologies are used in the development of the project:

- ✓ JAVA
- ✓ JAVA SERVER PAGES
- ✓ JAVA DATABASE CONNECTIVITY
- ✓ ORACLE
JAVA
Java is a general purpose, object-oriented programming language developed by Sun Microsystems of USA in 1991. Originally called Oak by James Gosling. Java was designed to solve the problem of connecting many household machines together. Then it was redesigned to work with cable TV. When the World Wide Web became popular in 1994, Sun realized that java was the perfect programming language for the web.

The most striking feature of the language is that it is a platform-neutral language. Java is the first programming language that is not tied to any particular hardware or operating system. Programs developed in java can be executed anywhere on any system. Java is a dynamic language. Java is capable of dynamically linking in new class libraries, methods and objects.
Java can also determine link or abort the program, depending on the response.

Java Virtual Machine (JVM):
All language compilers translate source code into machine code for a specific computer. Java compiler does the same thing.
Java compiler produces an intermediate code known as byte-code for a machine that does not exist. This machine is called Java virtual machine and it exists only inside the computer memory.
The process of compiling a java program into byte-code is also referred as virtual machine code.

Java Virtual Machine code is not machine specific. Java interpreter generates the machine specific code by acting as an intermediary between the virtual machine and real machine.
JAVA DATABASE CONNECTIVITY

Java Database Connectivity is the software abstraction that interfaces your java application and the database we want to access.

It sits between the application and the database we want to access. It provides a set of API’s, which is uniform across all databases, windowing systems, operating systems and hardware environments.

**JDBC ARCHITECTURE:**

If we focus further on the JDBC layer, we will get the fig. below, the JDBC interface layer defines all the APIs used by our application programs.

The driver manager loads the required driver, directs calls to a specific database to the corresponding driver and provides information about the driver to the application, if the application wants that information.
JDBC-ODBC Bridge

JDBC and ODBC are similar in design. The JDBC-ODBC Bridge enables us to access databases using ODBC drivers from JDBC. This bridge effectively translates the JDBC API calls into the corresponding ODBC calls. ODBC drivers are available for almost all types of databases.

Fig: JDBC architecture when JDBC-ODBC Bridge is incorporated.

Types of JDBC Drivers

Java Soft has divided JDBC drivers into four categories, based on their construction and the type of database they are intended to support. These four categories are explained below.

1. JDBC-ODBC Bridge Driver

The JDBC-ODBC Bridge is an asset of C libraries and client-side libraries that translates the JDBC API calls into ODBC cells and connects to the ODBC driver manager.
2. NATIVE API Driver

These types of drivers use vendor supply C language libraries to implement JDBC API calls into vendor specific functionalities. These types of drivers translate the JDBC API calls into a DBMS independent protocol and will send it to middle-tier server over a socket connection.

3. Native Protocol Driver

The driver communicates with database server directly over a socket connection. They can be used in java applications that access local databases or in a client server environment. Pure java JDBC drivers are very effective for use in Internet.

4. Java’s Support for JDBC

Java’s support for JDBC comes from a set of interfaces and classes defined in java SQL package. These interfaces and classes are listed below.

- Interface Callable Statement
- Interface Connection
- Interface Database Meta Data
- Interface Driver
- Interface Prepared Statement
- Interface Result Set
- Interface Result Set Meta Data
- Interface Statement
- Class Date
- Class Driver Manager
- Class Driver Property Info

SERVLETS:

A Servlet is a Java programming language class that is used to extend the capabilities of servers that host applications access via a request-response-programming model. Although Serves can respond to any type of request, they are commonly used to extend the applications hosted by Web Servers. For such applications, Java Servlet Technology defines HTTP-specific Servlet classes. The javax.servlet and javax.servlet.http packages provide interfaces and classes for writing Serves. All Serves must implement the Servlet interface, which defines life-cycle methods. When implementing a generic service, you can use or extend the Generic Servlet class provided with the Java Servlet API. The HttpServlet class provides methods, such as doGet() and doPost(), for handling HTTP-specific services.
Servlet Life Cycle:

The three methods are central to the lifecycle of a Servlet:

1. init()
2. service()
3. destroy()

They are implemented by every Servlet and are invoked at specific times by the server.

The life cycle of a Servlet is controlled by the container in which the Servlet has been deployed. When a request is mapped to a Servlet, the container performs the following steps:

1. If an instance of the Servlet doesn’t exist, the web container
   a. Loads the Servlet Class.
   b. Creates an instance of the Servlet Class.
   c. Initializes the Servlet instance by calling the init() method.

2. Invokes the service() method, passing request and response objects. The Servlet remains in the server’s address space and is available to process any other requests received from clients. The service() method is called for each client.

3. The server calls the destroy() method to relinquish any resources, such as file handles that are allocated for the Servlet and its object can then be garbage-collected.
Java Server Pages (JSP):

JSP technology provides an easy way to create dynamic web pages. JSP uses a component based approach that allows web developers to easily combine static HTML for look and feel with Java components for dynamic factors. The simplicity of this component-based model, combined with the cross platform power of Java, allows a web development environment with enormous potential.

JSP shows how to develop Java based web applications without having to be a hard core programmer. Web page authors will benefit from chapters on generating dynamic content, handling session information, accessing Database, authenticating users and personalizing content. JSP is a technology for developing web pages that include dynamic content. Unlike a plain HTML page, which contains static content that always remains the same, a JSP page can change its context based on any no of various items, including the identity of the user, the user’s browser type, information provided by the user and selections made by the user, functionality such as this can be used to create web applications like shopping carts and employee directories.

A JSP page contains standard markup language elements, such as HTML tags, just like a regular web page. However, a JSP page also contains special JSP elements that allow the server to insert dynamic content in the page. JSP elements can be used for a wide variety of purposes, such as retrieving information from a database or registering user preferences. When a user asks for a JSP page, the server executes the JSP elements, merges the results with the static parts of the page and sends the dynamically composed page back to the browser.

JSP defines a number of standard elements useful for any web application such as accessing JavaBeans Components, passing control between pages and sharing information between requirements, pages and users.

Programmers can also extend the JSP syntax by implementation application-specific elements that perform tasks such as accessing databases and Enterprise JavaBeans, sending email and generating HTML to present application specific data.
**ORACLE:** Oracle is based on the relational data model and uses SQL (Structured Query Language) as its query language. The relational data models:

- It is a model that is easily understood and visualized.
- Integrity laws that is easy to develop and understand.
- Reduced data redundancy.
- Independent physical storage and logical database structures.

**Advantages of ORACLE**

- Oracle supports a large number of concurrent users.
- Oracle supports a client-server environment. It enables processing to be split between the database server and client application programs.

**JSMS:** JSMS stands for Java Simple Message Service. It is an API. JSMS supports various transport facilities for sending (and receiving) Short Messages (SMS) and Multimedia Messages (MMS). JSMS provides a small footprint SMS and mail API that can easily be applied on embedded devices. The footprint of the JSMS API including drivers for several devices is just above 150k.
3.4 SmsService Implementations

JSMS currently provides the following SmsService implementations:

- **GsmSmsService** (for GSM devices supporting **GSM 03.38, 03.40 & 07.05**)
- **TapSmsService** (Implementation of the **TAP/IXO** (Paging) protocol)
- **UcpSmsService** (**UCP** Universal Computer Protocol)
- **Cimd2SmsService** (**CIMD2** - Computer Interface to Message Distribution)
- **SmppSmsService** (**SMPP** - Short Message Peer-to-Peer Protocol)

Additionally, the following protocols are implemented:

- **MM1** (Multimedia Messaging using WAP/WAP PUSH and GPRS) and jSMS also contains a small footprint **SMTP** client for sending e-mails.

1. **GsmSmsService**

   This class may be used to send and receive GSM Short Messages (SMS) using a **GSM mobile device** (e.g. a Mobile Phone). The device may be attached to the serial port or to a TCP/IP capable terminal server.

2. **TapSmsService**

   This class may be used to send Short Messages (SMS) / Pager messages to a mobile recipient using **TAP/IXO** protocol.

3. **UcpSmsService**

   This class may be used to send (and receive) Short Messages (SMS) to a mobile recipient using **UCP** (Universal Computer Protocol).

4. **Cimd2SmsService**

   This class may be used to send (and receive) Short Messages (SMS) to a mobile recipient using **Nokia's CIMD2 protocol** (Computer Interface to Message Distribution).

5. **SmppSmsService**

   This class may be used to send (and receive) Short Messages (SMS) to a mobile recipient using the **SMPP protocol**.
3.5 Data Flow Diagrams:

**LEVEL 0**

- Mobile user
  - SMS
  - Deliver SMS
  - Campaign message
  - SMSme server
  - Manage publisher
  - Administrator

**LEVEL 1**

- Mobile user
  - Set service request
  - Service Access DB
  - SMS processing
  - Campaign details
  - Register Campaign
  - Store details
  - Administrator
  - Access

**Level 0 diagram**

**Level 1 diagram**
4. DETAIL DESIGN

4.1 UML Diagrams

4.1.1 Class Diagrams

**SMSme Admin Class Diagram**
Publish Portal Class Diagram

- **Register Campaigner**
  - Campaigner_name : String
  - organization_name : String
  - org_type : String
  - address : String
  - city : String
  - zip : String
  - contactNo : long
  - mobileNo : long
  - contact_person : String
  - email : String
  - gender : String
  - username : String
  - password : String
  - register() : void

- **CheckUserNameAvailability**
  - isUserNameAvailable() : void

- **LoginCampaigner**
  - username : String
  - password : String
  - login() : void

- **CampaignPublisher**
  - campaignName : String
  - campaignCode : String
  - promotionMessage : String
  - launchDate : Date
  - expiryDate : Date
  - publishCampaign() : void

- **CampaignManager**
  - editCampaign() : void
  - deleteCampaign() : void
  - suspendCampaign() : void
  - resumeCampaign() : void
  - statusOfCampaigns() : void
  - changePassword() : void

1..*
4.1.2 Use Case Diagrams

**Use case diagram for campaigner process**

- Campaign request
- Authenticate
- Login
- <<Extend>>
- Campaign management
- View campaign statistics

**Use case diagram for end user processes**

- Send SMS
- <<Include>>
- Process service request
- Receive SMS
Use case diagram for SMSme administrator

- Manage campaign and Campaigners
- <<Extend>>
- Admin
- <<Extend>>
- Activate registered campaigners
- Control campaigns
Use case diagram for the SMS based pull campaign processing system

- **Campaigner request**
- **Authenticate**
- **Login**
- **Campaign Management**
- **Manage campaign and Campaigners**
  - **Control campaign**
  - **Activate registered campaigners**
- **View campaign statistics**
- **Send SMS**
- **End user**
- **Process service request**
- **Receive SMS**
4.1.3 Activity Diagrams

Publisher Portal Registration

This diagram (Figure: 1) describes the publisher registration with SMSme server. If a publisher wants to register with SMSme server, he has to enter his profile details along with SMS code. These details are entered in the database and an acknowledgement is sent to the registered publisher.

SMSme admin support system category management

This diagram (Figure: 2) describes SMSme admin support system for category management. If an administrator wants to login, then the administrator has to enter username and password and an authentication process takes place, which fetches data from database. Various admin services are displayed where the admin can control and manage company categories i.e. he can add a category, remove a category or update a category. The data is updated in the database accordingly.

Publisher Portal Publishing Service

This diagram (Figure: 3) describes the publisher publishing the service. If a publisher wants to login, he has to enter username and password and an authentication process takes place, which fetches data from database. If he is a valid publisher, publisher services are displayed where he can submit his service information. This information is stored in the database and an acknowledgement is sent to the publisher.

SMSme Service Access

This diagram (Figure: 4) describes the SMSme service access by the mobile user. Whenever a mobile user wants to request for some information, he will send an SMS with the SMS short code. Then the SMSme server will process the request by validating the code and then fetching the appropriate data from the database corresponding to the code. The server will send the service information to the user who requested for it.

SMSme admin support system publisher management

This diagram (Figure: 5) describes SMSme admin support system for publisher management. If an administrator wants to login, then the administrator has to enter username and password and an authentication process takes place, which fetches data from database. Then the administrator can view registered publishers and can change their status i.e. active or inactive. The status is updated in database accordingly.
SMSme admin support system campaign controller

This diagram (Figure: 6) describes SMSme admin support system for campaign control. If an administrator wants to login, then the administrator has to enter username and password and an authentication process takes place, which fetches data from database. Various admin services are displayed where the admin can control and manage campaigns i.e. he can suspend or resume the campaigns. The status is updated in database accordingly.
Publisher Registration Activity Diagram

Figure 1
Category Management Activity Diagram

Figure :2
Publisher publishing campaigns

![Diagram showing the process of publisher publishing campaigns]

Figure :3
SMSme Service Access Activity Diagram

<table>
<thead>
<tr>
<th>SMSme Service User</th>
<th>SMSme Server</th>
<th>DB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request Service Info</td>
<td>process request</td>
<td>fetch DB</td>
</tr>
<tr>
<td>Send SMS with service Short code</td>
<td>is Short code valid?</td>
<td></td>
</tr>
<tr>
<td></td>
<td>invalid</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Valid</td>
<td></td>
</tr>
<tr>
<td></td>
<td>send service into as SMS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Recieve SMS</td>
<td></td>
</tr>
</tbody>
</table>

Figure :4
Publisher Management

![Diagram of Publisher Management Process]

Figure:5
**Campaign Controller Activity Diagram**

```
<table>
<thead>
<tr>
<th>Admin</th>
<th>SMSme Server</th>
<th>DB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enter UserID</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Process Login Request</td>
<td>Fetch DB</td>
<td></td>
</tr>
<tr>
<td>Valid or Not?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Valid</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display Admin Services</td>
<td></td>
<td></td>
</tr>
<tr>
<td>View Published Servers</td>
<td>Process Request</td>
<td>Fetch data</td>
</tr>
<tr>
<td>Suspend/Resume</td>
<td>Send response</td>
<td></td>
</tr>
<tr>
<td>Change status published</td>
<td>Update Status</td>
<td>Store data</td>
</tr>
<tr>
<td>Send ACK</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

**Figure :4**
4.1.4 Sequence Diagrams:

SMSme Admin Support System
Publisher Management Sequence Diagram

1. login with user id and pwd
   1.1 Process request
      2. get login data
         2.1 fetch data
         2.2 result
         3. validate login
      4. display admin service
5. valid registered publisher
6. process request
   6.1 Process request
   7. get valid publisher
      7.1 fetch data
      7.2 send response
8. modify registered publisher status
   8.1 Process request
   9. update status
      9.1 store data
      9.2 send acknowledgment
SMSme Admin Support System
Campaign Controller Sequence Diagram

1: Login with UserID, Password ( )
1.1: Process Request ( )
2: Get Login Data ( )
2.1: Fetch Data ( )
2.2: Result ( )
3: Validate Login ( )
4: Display Admin Services ( )
5: View Published Services ( )
6: Process Request ( )
7: Send Acknowledgement ( )
7.1: Fetch Data ( )
7.2: Send Request ( )
8: Change Published Status ( )
8.1: Display Status ( )
9: Update Status ( )
9.1: Store Data ( )
9.2: Send Acknowledgement ( )
Publisher portal service public sequence diagram

Publisher

1. Login ()

1.6. SendAck ( )

2. Publisherservicemsg ()

1.1. Process request ( )

1.2. GetLoginData ( )

1.4. Result ( )

1.5. Validatelogin ( )

1.3. Fetch data ( )

SMSme Server

2.1. Process request ( )

2.2. Storemsg ( )

2.3. Storedata ( )

2.4. Result ( )

DataBase

2.5. SendAck ( )
Publisher portal registration sequence diagram

Publisher

1. Register with SMS Code ( )

1.6. Send Ack ( )

SMSme Server

1.1. Process request ( )

1.2. GetKeycode ( )

1.4. Result ( )

1.5. Valid key code ( )

1.7. Create Record ( )

1.8. Store Data ( )

1.9. Result ( )

DataBase

1.3. Fetch Data ( )
SMSme service SMSme Server Access Sequence diagram

1. Send SMS with Short Code ( )
   1.1. Process request ( )
   1.2. GetService Information ( )
   1.3. Fetch Data ( )
   1.4. Result ( )
   1.5. Validate Service Short code ( )
   1.6. Send Ack ( )

2. Send service into SMS ( )
   2.1. Store Data ( )
   2.2. Result ( )

SMS me Service User

SMSme Server

DataBase
Category management sequence diagram

1: Login with UserID, Password ( )
1.1: Process Login Request ( )
2: Get Login Data ( )
2.1: Fetch Data ( )
2.2: Result ( )
3: Validate User ( )
4: Display Admin Services ( )
5: Send Ack ( )
5.1: Manage Campaign Categories ( )
5.2: Add or Remove Categories ( )
6: Process Request ( )
7: Send Ack ( )
7.1: Handle DataBase Request ( )
7.2: Send Ack ( )
5 TABLES

1. TABLE NAME: SERVER_USER_MASTER

<table>
<thead>
<tr>
<th>FIELD NAME</th>
<th>DESCRIPTION</th>
<th>DATA TYPE</th>
<th>CONSTRAINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>U_NAME</td>
<td>USERNAME</td>
<td>VARCHAR2</td>
<td>PRIMARY KEY</td>
</tr>
<tr>
<td>PWD</td>
<td>PASSWORD</td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>NAME</td>
<td>ANY NAME</td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
</tbody>
</table>

SQL STATEMENT:

CREATE TABLE SERVER_USER_MASTER (  
U_NAME                     VARCHAR2 (15) PRIMARY KEY,  
PWD                                 VARCHAR2 (8) ,  
NAME                         VARCHAR2 (25));

2. TABLE NAME: PUBLISHER_MASTER

<table>
<thead>
<tr>
<th>FIELD NAME</th>
<th>DESCRIPTION</th>
<th>DATA TYPE</th>
<th>CONSTRAINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>U_ID</td>
<td>USER IDENTIFICATION</td>
<td>VARCHAR2</td>
<td>PRIMARY KEY</td>
</tr>
<tr>
<td>U_NAME</td>
<td>USERNAME</td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>PWD</td>
<td>PASSWORD</td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>KEY_CODE</td>
<td>KEY CODE</td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>U_STATUS</td>
<td>USER STATUS</td>
<td>NUMBER</td>
<td>-</td>
</tr>
</tbody>
</table>

SQL STATEMENT:

CREATE TABLE PUBLISHER_MASTER (  
U_ID                         VARCHAR2 (4) PRIMARY KEY,  
U_NAME                    VARCHAR2 (15) ,  
PWD                     VARCHAR2 (8) ,  
KEY_CODE                VARCHAR2 (4),  
U_STATUS                NUMBER (1));
1. **TABLE NAME : CATEGORY**

<table>
<thead>
<tr>
<th>FIELD NAME</th>
<th>DESCRIPTION</th>
<th>DATA TYPE</th>
<th>CONSTRAINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAT_ID</td>
<td>CATEGORY IDENTIFICATION</td>
<td>VARCHAR2</td>
<td>PRIMARY KEY</td>
</tr>
<tr>
<td>CAT_NAME</td>
<td>CATEGORY NAME</td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>DESC</td>
<td>DESCRIPTION</td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
</tbody>
</table>

**SQL STATEMENT:**

```
CREATE TABLE CATEGORY (
    CAT_ID    VARCHAR2(4),
    CAT_NAME    VARCHAR2(15),
    DESC    VARCHAR2(160));
```  

4. **TABLE NAME: PUBLISHER_REGISTER**

<table>
<thead>
<tr>
<th>FIELD NAME</th>
<th>DESCRIPTION</th>
<th>DATA TYPE</th>
<th>CONSTRAINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>U_ID</td>
<td>USER IDENTIFICATION</td>
<td>VARCHAR2</td>
<td>PRIMARY KEY</td>
</tr>
<tr>
<td>ORG_NAME</td>
<td>USERNAME</td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>ADDR</td>
<td>PASSWORD</td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>CITY</td>
<td>KEY CODE</td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>STATE</td>
<td>USER STATUS</td>
<td>NUMBER</td>
<td>-</td>
</tr>
<tr>
<td>CTRY</td>
<td></td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>ZIP</td>
<td></td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>PHNO</td>
<td></td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>PRSN_NAME</td>
<td></td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>MOBILE_NO</td>
<td></td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>EMAIL_ID</td>
<td></td>
<td>VARCHAR2</td>
<td>-</td>
</tr>
<tr>
<td>DOR</td>
<td></td>
<td>DATE</td>
<td>-</td>
</tr>
</tbody>
</table>

**SQL STATEMENT:**

```
CREATE TABLE PUBLISHER_REGISTER (
    U_ID                        VARCHAR2 (4) REFERENCES PUBLISHER_MASTER (U_ID),
    ORG_NAME               VARCHAR2 (50),
    ADDR                 VARCHAR2 (200),
    CITY                        VARCHAR2 (30),
    STATE                     VARCHAR2 (30),
    CTRY                       VARCHAR2 (30),
    ZIP                          NUMBER (6),
    PHNO                VARCHAR2 (15),
    PRSN_NAME
```
5. TABLE NAME: CAMPAIGN_MASTER

<table>
<thead>
<tr>
<th>FIELD NAME</th>
<th>DESCRIPTION</th>
<th>DATA TYPE</th>
<th>CONSTRAINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMPN_ID</td>
<td>CAMPAIGN IDENTIFICATION</td>
<td>VARCHAR2 2</td>
<td>PRIMARY KEY</td>
</tr>
<tr>
<td>U_ID</td>
<td>USER IDENTIFICATION</td>
<td>VARCHAR2 2</td>
<td>-</td>
</tr>
<tr>
<td>CMPN_TITLE</td>
<td>CAMPAIGN TITLE</td>
<td>VARCHAR2 2</td>
<td>-</td>
</tr>
<tr>
<td>CMPN_MSG</td>
<td>CAMPAIGN MESSAGE</td>
<td>VARCHAR2 2</td>
<td>-</td>
</tr>
<tr>
<td>CMPN_DOC</td>
<td>DATE OF</td>
<td>DATE</td>
<td>-</td>
</tr>
<tr>
<td>CMPN_DOE</td>
<td>DATE OF</td>
<td>DATE</td>
<td>-</td>
</tr>
<tr>
<td>CMPN_LAST_UPDATED</td>
<td>LAST UPDATED MESSAGE</td>
<td>DATE</td>
<td>-</td>
</tr>
<tr>
<td>TOT_CMPN_RESPONDENTS</td>
<td>TOTAL CAMPAIGNS</td>
<td>NUMBER</td>
<td>-</td>
</tr>
<tr>
<td>CMPN_STATUS</td>
<td>MESSAGE STATUS</td>
<td>NUMBER</td>
<td>-</td>
</tr>
</tbody>
</table>

SQL STATEMENT:

```
CREATE TABLE CAMPAIGN_MASTER (
    CMPN_ID                      VARCHAR2 (4) PRIMARY KEY,
    U_ID                         VARCHAR2 (4) REFERENCES PUBLISHER_MASTER (U_ID),
    CMPN_TITLE             VARCHAR2 (50),
    CMPN_MSG                   VARCHAR2 (160),
    CMPN_DOC                   DATE,
    CMPN_DOE                   DATE,
    CMPN_LAST_UPDATED   DATE,
    TOT_CMPN_RESPONDENTS NUMBER (5),
    CMPN_STATUS             NUMBER (1))
```
6. TABLE NAME: SUBSCRIBER_LOG

<table>
<thead>
<tr>
<th>FIELD NAME</th>
<th>DESCRIPTION</th>
<th>DATA TYPE</th>
<th>CONSTRAINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBSCRIBER_ID</td>
<td>NUMBER</td>
<td>VARCHAR2</td>
<td>PRIMARY KEY</td>
</tr>
<tr>
<td>U_ID</td>
<td>USER IDENTIFICATION</td>
<td>VARCHAR2</td>
<td></td>
</tr>
<tr>
<td>CMPN_ID</td>
<td>CAMPAIGN ID</td>
<td>VARCHAR2</td>
<td></td>
</tr>
<tr>
<td>MDN</td>
<td>MESSAGE NUMBER</td>
<td>NUMBER</td>
<td></td>
</tr>
<tr>
<td>RES_DATE</td>
<td></td>
<td>DATE</td>
<td></td>
</tr>
</tbody>
</table>

SQL STATEMENT:

CREATE TABLE SUBSCRIBER_LOG (  
SUBSCRIBER_ID NUMBER (4) PRIMARY KEY,  
U_ID VARCHAR2 (4) REFERENCES PUBLISHER_MASTER (U_ID),  
CMPN_ID VARCHAR2 (4) REFERENCES CAMPAIGN_MASTER (CMPN_ID),  
MDN NUMBER (12),  
RES_DATE DATE);

7. TABLE NAME: ADMIN_LOGIN_MASTER

<table>
<thead>
<tr>
<th>FIELD NAME</th>
<th>DESCRIPTION</th>
<th>DATA TYPE</th>
<th>CONSTRAINT</th>
</tr>
</thead>
<tbody>
<tr>
<td>USERNAME</td>
<td>USER NAME</td>
<td>VARCHAR2</td>
<td>PRIMARY KEY</td>
</tr>
<tr>
<td>PASSWORD</td>
<td>PASSWORD</td>
<td>VARCHAR2</td>
<td></td>
</tr>
</tbody>
</table>

SQL STATEMENT:

CREATE TABLE ADMIN_LOGIN_MASTER (  
USERNAME VARCHAR2 (15) PRIMARY KEY,  
PASSWORD VARCHAR2 (15));
6. Implementation

6.1 Implementation Procedure.

The system is divided into 3 major modules:

1. SMS Me server admin support system
2. Campaign Publisher Port
3. SMSme Service Delivery System

- This module is used by the admin with a web interface.
- It allows managing registered publishers and controlling their published campaigns.
- The admin can access reporting on various service utilization.
How to work this….

- Server Admin has to register.
- He has to Login with User Name and Password.
- This module validates the User Name.
- Admin authorizes Registered Publishers.
- This module maintains the Publisher Manager who allows to access with the Registered Publishers.
- Admin has all the rights to stop the process done by Campaign Controller (Suspend/Resume).
- He can mange the Categories (e.g.: Restaurants, Cricket Scores, etc).
- Admin can also change the status of Registered Publishers. (Active/Inactive)
- Admin can view the Published Services with Campaign Statistics. (How many No. of Publishers registered, how many Campaigns are used etc.)

The core components of this module are:

* User Authenticator
* Campaign Category Manager
* Campaign Controller
* Reporting Engine

Campaign Category Manager:

In this all the campaigns provided by various publishers are divided and managed based on the type of request given by publisher like hospital, advertisement etc.

Campaign Manager:

In this all campaigns are checked for their validation and then activated depending on the status of the publisher.

Campaign Controller:

In this campaigns are controlled by the administrator, it prevents accessing of unauthorized publishers and their campaigning.

Reporting Engine:

It is used mainly to report the status of publisher to the administrator frequently and exactly.
This module facilitates the publisher in registering, publishing campaigns and managing published campaigns.

**How to work this….**

- Publisher has to register to publish a message.
- Publisher must submit Registration Details with SMS Code.
- He has to Login with User Name, Password.
- Validates User Name by Publisher.
- As soon as Administrator authorizes using module –1, Publisher cannot Access with SMS Portal.
- Authorized Publishers can send the Campaigns up to 160 characters. This Manages by the Campaign manager.
- Publisher manages Campaign Access Reporter, who allows to view the Statistics of Campaigns.
- All the Publisher Services are available with Campaign Schedules.
The core components of this module are:

* User Authenticator
* Publisher Register
* Campaign Manager
* Campaign Access Reporter

User Authenticator:

Checks for the validation of the user accessing the application.

Publisher Register:

Here the publisher has to be registered in order to utilize the technology in campaigning. Publisher has to be registered if he is a new user accessing the application.

Campaign Manager:

In this all campaigns are checked for their validation and then activated depending on the status of the publisher.

Campaign Access Reporter:

The activation and deactivation of the campaign takes place in this case depending on the status of publisher.
This module is responsible for providing SMS Me services.
End user is campaigner

How to work this....

- SMSme Service provides Campaigns to requested Users.
- Server Validates the Campaign Code requested by the Mobile User.
- It sends an SMS with Service Short Code.
- Server maintains one Service Access Log for Access Reporting.

The core components are:

* Campaign Code Validator
* Campaign Message Sender
* Service Access Log

Campaign Code validator:

Here administrator checks the validity and uniqueness of the key code given to the publisher for campaigning his product.

Campaign Message Sender:

This is responsible to send the message to the end user. As this message format will be checked before sending, and it will be in such a format so that it can be easily understood.
Service Access Log:

In this the validation of user, his ID, Campaign ID, expire date of campaigning will be done.

Consider the below example of “SMS Based Pull Campaign Processing System” it describes about how the system works

A full cycle example follows:

SMS Me system has published its mobile number to the public through print media and by other means. Its mobile number is: 8866695452. That is, now people know that, there is an SMS directory service provider existing by the name SMS Me, and its number is 8866695452. For any enquires, they SMS some code to this number, in order to get the services.

Now consider an example of a publisher:

1. Config Software Solutions has registered with SMS Me. They have selected CNFG as their campaign code while registration (Refer to registration screen).

2. Config Software Solutions is about to launch a seminar on Web technologies on 12-Oct-2007. They wanted to advertise this info through SMS Me.

3. Today, ie., 28-Sep-2007, Config Software Solutions, published a campaign in SMS Me as follows.
   Campaign Msg: A free seminar on Web Technologies on 12-Oct. Call us on or before 10-Oct-07 to block your entry.
   Start Date: 01-Oct-2007 (See create campaign screen shot)
   Expiry Date: 10-Oct-2007 (See create campaign screen shot)

4. An end user, say, Mr. Sam wanted to know the offers/events happening with Config Software Solutions.

5. Mr. Sam is now sending an SMS to 8866695452, from his mobile, say 9885558585..
   SMS Msg: CNFG.
   To: 8866695452.

6. SMS Me system, now receives a message 'CNFG' from mobile 7885558585. On receiving this SMS, the application will search the database table T_PUBLISHER_MASTER for publishers with KEY_CODE of 'CNFG' using the following query.

   SELECT F_USER_ID FROM T_PUBLISHER_MASTER WHERE KEY_CODE = 'CNFG';

   Assume that the U_ID of Config software solutions is 'Config'. Then the above query returns 'Config'. Now the following query will be executed to get the running campaigns of 'Config' user.

   SELECT CMPG_MSG FROM CAMPAIGN_MASTER WHERE U_ID = 'Config'
   AND SYSDATE BETWEEN F_LAUNCH_DT AND F_END_DT;

   55
The above query returns the running campaign promo messages of 'Config' publisher. Now the application will send all these promo messages (in this example, only one promo is running) as sms to the mobile 7885558585: If more than one promo exists, then, all the promos will be sent in a single SMS.

6.2 Further System

When you send the message through your mobile phone (attached to the computer), SMS- it will use the parameters (senders ID, SMS center, etc) of your phone to send the message. Sending will work the same as when sending messages from your phone, except that you type the message on your computer. Not all phone manufactures implement however a link to the computer (IR port or serial cable link). You send text, pictures and ring tones. Having the mobile phone connected to the computer has the additional advantage that you can transfer data from your phone to the computer and vice versa.

6.3 Problem and Solution

Problem and Solution describes the real-time experience about any work done. While working on “SMS Based Pull Campaign Processing System” we have faced some minor and major problems.

Minor: we have faced a lot of minor problems but at that time each minor problem was looks very terrible unless we found solution of it.

1) Some of just while writing the code we got some syntax errors like missing some parameters.

2) For the first time when we are connecting to the database, the connection is not succeeded after we got to know that we use the different driver to connect.

3) While connecting to the database, for authentication we have used 1 xml file which consist of database parameters, in XML file the port number for database we have used same as tomcat because of mismatch port number we are unable to connect the database, after a period of time we got the solution that database has its own port number.

Major: we have faced some big problems in sending and receiving sms by using GSM modem the problem was in configuration.

1) Some of other problem in integration part while combining all modules together. This problem we identified while testing the complete application in the testing phase.

2) We have faced some problems with different models of mobile devices like nokia N series its not compatible with our application. In some of this model we are able to receive the sms, but we faced some problem in replying it.
6.4 Screen Shots

Fig no.6.4.1 Admin Page
Fig no.6.4.2 Login Page
Fig no.6.4.3 Active user Page
Fig no.6.4.4 Inactive user Page
Fig no.6.4.5 New Category Page
Fig no.6.4.6 Update Category Page
Fig no.6.4.7 View Category
Screens for Campaign Self Service

Fig no.6.4.8 Create campaign
Fig no.6.4.9 Register publisher Page
Fig no.6.4.10 Profile Page
Fig no.6.4.11 Edit Profile Page
Fig no.6.4.12 Campaign List Page
### Fig no.7.2.13 Edit Campaign Page

**Edit Campaign**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Campaign ID</td>
<td>P001</td>
</tr>
<tr>
<td>Campaign code</td>
<td>Apollo</td>
</tr>
<tr>
<td>Category ID</td>
<td>Hospital</td>
</tr>
<tr>
<td>Promo message</td>
<td></td>
</tr>
<tr>
<td>Launch date</td>
<td>09/25/2007</td>
</tr>
<tr>
<td>Expiry date</td>
<td>10/25/2007</td>
</tr>
</tbody>
</table>

Update button
Logged out successfully

Fig no.6.4.14 Logout Page
7. Testing

Testing of an application is undertaken to ensure that software functions correctly and consistently. The Testing procedure includes the development of test cases, each of which is intended to test a particular element of the application. A good test case is one that has a high probability of finding an as yet undiscovered error. The test results are analyzed to check the correctness of the software. A successful test is one that uncovers as yet undiscovered error. The test suite for a class will consists of test cases chosen to satisfy specific testing requirements.

Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct then the goal will be achieved successfully. Inadequate testing or non-testing to errors and creates problem like:

1. The time lag between the cause and appearance of the problem.
2. The effect of system error on files and records with in the system.

The best program is worthless if it is does not respond to user needs. The first test of a system it to see whether it produces the correct outputs.

The testing phase in software development involves unit testing and integration testing.

7.1 Unit Testing:

Unit testing involves testing individual classes or hierarchies of classes, and is typically performed by the software developer. The tester checks the outcome of each test case in the suite by executing a feature (method) and comparing the actual results with the expected results. Unit testing is used to check the correctness of the algorithms implemented in the class and conformance of return values to expected results. The “Unit” can be either a class hierarchy, a class, or a method.

7.2 Integration Testing:

Integrating testing deals with how well the single units work together. In object-oriented systems, although we test the individual features (methods) within a class. A neophyte in the software world might ask a seemingly legitimate question once all modules have been unit tested. “If they individually work, why do you doubt that they will work when we put them together?” The problem of course, is “putting them together” interfacing. Data can be lost across an interface, one module can have an inadvertent, adverse affect on another; sub functions, when combined, may not produce the desired major function; global data structures can present problems. Integration testing is a systematic technique for constructing the program structure while conducting tests to uncover errors associated with interfacing.

There is often a tendency to attempt non-incremental integration. That is, to construct the program using “big bang” approaches. All modules are combined in advance. The entire program is tested as a whole. Incremental integration is the antithesis of the big bang approach. The program is constructed and tested in small segments, where errors are easier to isolate and correct, interfaces are more likely to be tested completely and a systematic test approach may be applied.

We are more interested in class testing, which combines some aspects of unit testing and integration testing.
7.3 Test Suite

The basic goal of testing is to create a minimum number of test cases to uncover the maximum number of errors for a given amount of effort. The test suite must include test cases to consider the functional, structural and integration aspects of a class under test. The functional aspect is an external viewpoint that focuses on the behavior promised by the interface of the class whereas the structural perspective is an internal viewpoint and is based on the actual internal program structure. The tester can then determine the coverage for test aspect in question.

7.4 Test Cases

The extended data aspect represents the data manipulation and to be tested to check the implementation behavior. A complete test case both starts and ends at the initial state. Test cases can be produced from EFSM using a combination of data flow testing and control flow testing.

Test case selection usually involves defining the purpose of the test and creating tests to satisfy each test purpose. Test purposes are derived manually such as from use-case descriptions or from checklists of common programming mistakes. Test selection techniques require categorizing tests as either structural tests, functional tests or integration tests, and recognizing the level of testing supported by this technique.

7.4.1 Functional Test Cases

Functional test cases, also known as black box test cases, are constructed based on the specification of the component. The specification of a class includes a class invariant and specification with pre and post conditions for each method. Coverage is shown as a percentage of the class specification that is covered. A method is executed to verify conformance to its assertions. More test cases are used to check invariants.

7.4.2 Structural Test Cases

Structural test cases, also known as white box test cases, are constructed by identifying individual paths through the code. This allows us to catch errors/bugs that were not detected in functional testing. Coverage is treated as a percentage of the lines of code that have been tested based on the test cases. It is not usually possible to execute all paths of the code, although every line of code should be executed at least once.

7.4.3 Integration Test Cases

Integration test cases deal with interactions between methods within a class (intra-class testing) and interactions between methods in different classes (inter-class testing). It focuses on checking methods, which may cause errors due to interactions. Test cases are produced by checking values of attributes used or set by these methods, or parameters passed between these methods. During integration tests, can needs to be taken with inheritance, polymorphism, and dynamic binding, which affect the dynamic run-time type of an object in a strongly typed object-oriented language.
7.5 Implementing Test Cases

Manual Testing

There are three main methods of manually implementing test cases. These are building test cases into your classes, creating a test script and creating a separate testing hierarchy.

Building Test Cases into your Classes

When building test cases into your classes, create test methods such as Test Class, test Objects Foxx for verifying the workings of classes and its instances. The test Class methods would test the associated class methods, create instances of the class, put these objects into default or specified states, and test the objects with related test cases. One can inherit and reuse test cases from the super class for subclass testing.

Creating a Test Script

This method involves creating an entirely separate test application. This application should: create a set of objects, send them messages, and compare the actual results to the expected results. This approach is simple but it makes inheriting test cases difficult.

Creating a separate Test Hierarchy

The benefits of creating a separate test Hierarchy is that it separates out the testing code from real code. Reuse of test cases can easily be achieved through inheritance, however this means that the two sets of code, the test code and real code, must be maintained together.
**TEST CASE:** To check the validity of user.

<table>
<thead>
<tr>
<th>TEST CASE# :-</th>
<th>User Validity</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEST OBJECTIVE:-</td>
<td>To Verify the validity of user</td>
</tr>
<tr>
<td>TEST DESCRIPTION:</td>
<td>This test is performed to check whether the data</td>
</tr>
<tr>
<td></td>
<td>entered by the user is properly getting entered into</td>
</tr>
<tr>
<td></td>
<td>the database in order to view his required</td>
</tr>
<tr>
<td></td>
<td>information</td>
</tr>
<tr>
<td>REQUIREMENTS VERIFIED:-</td>
<td>Yes</td>
</tr>
<tr>
<td>TEST ENVIRONMENT: -</td>
<td>Oracle database.</td>
</tr>
<tr>
<td>TEST SETUP/PRECONDITIONS:-</td>
<td>Connection should be established</td>
</tr>
<tr>
<td></td>
<td>between servlet and the database.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ACTIONS</th>
<th>EXPECTED RESULTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>User will enter</td>
<td></td>
</tr>
<tr>
<td>1. UserID</td>
<td>a. All the data in the form is taken and checked in the table with the help of</td>
</tr>
<tr>
<td>2. Password</td>
<td>servlet. If userid and password are properly matching then the user will view the</td>
</tr>
<tr>
<td>3. Registration details</td>
<td>main screen of the system.</td>
</tr>
<tr>
<td>4. The admin has the category</td>
<td>b. If the user does not enter field values properly then it gives an invalid</td>
</tr>
<tr>
<td>maintenance</td>
<td>screen with a message and returns to the same form.</td>
</tr>
<tr>
<td></td>
<td>c. Here the user should enter the all the necessary details that are specified in</td>
</tr>
<tr>
<td></td>
<td>the form and can not leave any box empty else an msg appears requesting the</td>
</tr>
<tr>
<td></td>
<td>missing info.</td>
</tr>
<tr>
<td></td>
<td>d. Here the publisher should select the category to which his champin belongs</td>
</tr>
<tr>
<td></td>
<td>else of new he should go for the new category. And that shall be updated by the</td>
</tr>
<tr>
<td></td>
<td>admin.</td>
</tr>
</tbody>
</table>
8 Conclusions

Application software system has been developed to case out the problems faced in the present system. The system described in this book have designed, Tested and documented to same greater extent. This package is expected to prove to be very useful to the “Online Shopping”.

The goals that are expected to achieve by the software are:

- Simplification of the operation.
- Less processing time and increasing the productivity.
- Each transaction is updated and processed immediately.
- Avoiding errors by minimizing human interaction through user friendly screens to enter data and Enquirers the information from the tables on like message.
- Portable and flexible for further enhancement.

After developing the system, it has been tested with sample data and the results are found to be in satisfaction. The system is user friendly are and easy to use prescribed by the standards have been followed in designing the input screens lending to consistency. Across forms and reduced user strains.

9 Scopes for Expansion

The system is developed in a modular fashion with different function performing their jobs. If any changes have to be done to the system then they can be easily achieved by making changes to the functions and they won’t affect the remaining part of the system.

The scope for the future enhancements in this particular application could be depending of the scope of the number of the subscribers/users the application can be made location dependent...That is the response can be send with the SMTP protocol via net and the resultant message could be of the particular area specified by the user ..not a predefined msg for all the queries.. This is a complex process which needs huge infrastructure support and a great network support at back end.
10 Bibliography


http://www.uml.org/

http://www.uiq.com

http://www.rational.com/uml

http://www.objectxp.com

http://www.exforsys.com/