

# **Architectures for two-way SMS services involving an SDI node**

Shelton Butau

March, 2010

# Architectures for two-way SMS services involving an SDI node

by

Shelton Butau

Thesis submitted to the International Institute for Geo-information Science and Earth Observation in partial fulfilment of the requirements for the degree in Master of Science in *Geoinformatics*.

## **Degree Assessment Board**

Thesis advisor	Dr.Ir R.A. de By Dr.Ir R.L.G. Lemmens
Thesis examiners	Chair: Dr. J.M. Morales External Examiner: Dr. A. Wombacher



INTERNATIONAL INSTITUTE FOR GEO-INFORMATION SCIENCE AND EARTH OBSERVATION  
ENSCHDEDE, THE NETHERLANDS

## **Disclaimer**

This document describes work undertaken as part of a programme of study at the International Institute for Geo-information Science and Earth Observation (ITC). All views and opinions expressed therein remain the sole responsibility of the author, and do not necessarily represent those of the institute.

# Abstract

Spatial Information remains key to any societal development and its importance is greater seen in most decisions in our society. Current trends have this information easily accessible via the internet, through different services and products. These services, though of great importance are not easily accessible to people in the southern contexts where the Internet is out of reach for many. Fortunately, the mobile phone in its basic form has an almost ubiquitous presence in these societies. Stemming from this presence, the mobile phone can then serve as an efficient communication platform. In this research project we explore the means of providing crucial spatial information to people in the southern contexts in two-way SMS services involving an SDI node. To propose the architectural framework, we review and scan current or ongoing projects defining and deriving the possible SMS and SDI functionality. The functionality derived is used to create a messaging scenario of a messaging service for rain information. This scenario is used to create a test use case for the SMS-SDI node messaging. The use case, using the systems approach in a service oriented environment, is used in the architecture formulation for the two-way SMS system involving an SDI node. The proposing the architectural framework we also highlight the limitations of the SMS technology and those of the SDI technology and how integration can be made possible. The proposed architecture is implemented for the use case and reviewed. The review leads to a general framework architecture that provides the means in which SMSs can be used in conjunction with SDI nodes to communicate spatial information in a two way manner.

## Keywords

*Short Message Service, SDI node, two-way architecture, SDI-SMS architecture, Spatial Data Infrastructure*



# Acknowledgements

I would like to thank GOD the Almighty for awarding me this lifetime opportunity to study at ITC. I extend my gratitude to my MSc sponsor, The Netherlands Fellowship, which made it possible for me to complete my course. I thank my supervisors at ITC, Dr. Ir. R.A. Rolf de By and Dr. Ir. R.L.G. Rob Lemmens for their unwavering support and advice. I thank all the ITC lecturers and supporting staff who contributed to the success of my studies. Many thanks to my colleagues from my GFM class for their support.

I would like to thank the ITC Christian fellowship support group; chaplains (Rev. Josine and Fr Wiel), leaders and congregation for the continued spiritual help and for making me feel at home away from home. I thank my family and friends for their love and prayers. God bless you all.

*“A theory is the more impressive the greater is the simplicity of its premises, the more different are the kinds of things it relates and the more extended the range of its applicability”.....**Albert Einstein***



# Contents

<b>Abstract</b>	<b>i</b>
<b>Acknowledgements</b>	<b>iii</b>
<b>List of Tables</b>	<b>ix</b>
<b>List of Acronyms</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation and Problem Statement . . . . .	1
1.2 Research Identification . . . . .	4
1.3 Method Adopted . . . . .	5
1.4 Structure of Thesis . . . . .	7
<b>2 Overview of Short Message Services and Spatial Data Infrastructures</b>	<b>9</b>
2.1 Introduction . . . . .	9
2.2 Short Message Services: What are they? . . . . .	9
2.2.1 How Short Message Service (SMS)'s generally work? . .	10
2.2.2 Where are they being used? . . . . .	10
2.3 What can be sent via SMS Formats? . . . . .	12
2.4 SMS modes, how do they work? . . . . .	12
2.5 Spatial Data Infrastructure: What is it? . . . . .	14
2.5.1 Concepts and Components . . . . .	14
2.6 SDI Architectures . . . . .	15
2.7 Overview of SMS technologies in the southern context . . . . .	15
2.8 Conclusion . . . . .	16
<b>3 Short Message Service System Functionality and Issues</b>	<b>17</b>
3.1 Introduction . . . . .	17
3.2 Positioning . . . . .	17
3.2.1 Positioning the mobile phone . . . . .	18
3.2.2 Positioning the subject of interest . . . . .	22
3.3 Message Parsing . . . . .	23
3.4 Alphanumeric Data Input Errors . . . . .	23
3.5 Resolving Grammatical Errors . . . . .	26
3.6 Cultural Issues pertaining to the Short Message Service . . . . .	27

3.7	Service access through USSDs . . . . .	27
3.8	The SMS sending or receiving . . . . .	28
3.9	Conclusion . . . . .	29
<b>4</b>	<b>Design of the two-way SMS Architecture(s)</b>	<b>31</b>
4.1	Introduction . . . . .	31
4.1.1	Use Case Identification . . . . .	32
4.1.2	Use Case Description . . . . .	32
4.1.3	Use Case Specification . . . . .	33
4.2	Requirements definition and analysis . . . . .	37
4.3	Domain Class Model: Rain Information Service . . . . .	37
4.4	System Architecture Modelling . . . . .	39
4.4.1	Layered System Architectural View of the RISS . . . . .	39
4.4.2	Structural Model View of the RISS . . . . .	41
4.4.3	Behavioural Model View of the RISS . . . . .	41
4.5	Conclusion . . . . .	43
<b>5</b>	<b>Implementation of the Two-way SMS Design Architecture System Prototype</b>	<b>45</b>
5.1	Introduction . . . . .	45
5.2	System Prototype Functionality Development . . . . .	45
5.2.1	Tools and techniques used . . . . .	47
5.3	Model Implementation . . . . .	47
5.3.1	Architecture Component Implementation . . . . .	47
5.4	Interface/ Engine implementation . . . . .	53
5.5	Review of architecture, The General Framework . . . . .	54
5.6	Conclusion . . . . .	57
<b>6</b>	<b>Discussion, Conclusions and Recommendations</b>	<b>59</b>
6.1	Introduction . . . . .	59
6.2	Discussion . . . . .	59
6.3	Shortcomings and limitations of two-way SMS involving the Spatial Data Infrastructures (SDI) node . . . . .	67
6.4	Conclusion . . . . .	68
6.5	Recommendations . . . . .	68
6.5.1	Current Research . . . . .	68
6.5.2	Future Research . . . . .	68
	<b>Bibliography</b>	<b>71</b>
<b>A</b>	<b>Use Case Template</b>	<b>77</b>
<b>B</b>	<b>Architectural views of the System by Stereotyped UML Diagrams</b>	<b>79</b>
<b>C</b>	<b>Summary of currently available products and technologies for SMS services</b>	<b>81</b>

<b>D</b>	<b>Cell-Id Look-up Application for Java ME</b>	<b>87</b>
<b>E</b>	<b>Code Extracts of the RISS Engine and its components</b>	<b>93</b>
<b>F</b>	<b>Detailed Description of the Operator Side SMS System Architecture</b>	<b>99</b>
F.1	SMS-enabled Global System for Mobile communication (GSM) Architectures . . . . .	99
F.1.1	The GSM Protocol Architecture . . . . .	101
F.2	General SMS Network Architecture . . . . .	102
F.3	SMS Protocol Stack . . . . .	102
<b>G</b>		<b>105</b>
<b>Index</b>		<b>107</b>



# List of Tables

3.1	Noisy Channel Method by example . . . . .	26
4.1	General Software Architectural View . . . . .	39
4.2	Stereotyped UML Diagrams . . . . .	40
5.1	Software and Hardware used in RISS . . . . .	46
B.1	Views of the System by Stereotyped UML Diagrams . . . . .	80
F.1	Transport Protocol Data Unit Types . . . . .	103



# List of Algorithms

1	<i>Message Parsing mechanism of the Rain Information Service System (RISS)</i> . . . . .	48
2	<i>Dictionary validation and Spell checker</i> . . . . .	49
3	<i>Profiling the RISS user</i> . . . . .	51



# List of Acronyms

<b>AGPS</b>	Assisted Global Positioning System
<b>API</b>	Application Programming Interface
<b>AOA</b>	Angle of arrival
<b>AOI</b>	Area of Interest
<b>AT</b>	Hayes AT
<b>DN</b>	Digital Number
<b>DBMS</b>	Database Management System
<b>DR</b>	Dead Reckoning
<b>GI</b>	Geographic Information
<b>GPS</b>	Global Positioning System
<b>GSDI</b>	Global Spatial Data Infrastructure Association
<b>GSM</b>	Global System for Mobile communication
<b>ICT</b>	Information Communication Technology
<b>ISO</b>	International Organisation of Standardisation
<b>J2ME</b>	Java 2 Micro Edition
<b>JSR</b>	Java Specification Request
<b>OGC</b>	Open Geospatial Consortium
<b>LOB</b>	Line Of Bearing
<b>LD</b>	Levenshtein Distance
<b>NLP</b>	Natural Language Processing
<b>PDU</b>	Protocol Description Unit
<b>RISS</b>	Rain Information Service System
<b>REST</b>	Representational State Transfer

<b>SDI</b>	Spatial Data Infrastructures
<b>SIM</b>	Subscriber Identity Module
<b>SME</b>	Short Message Entities
<b>SMSC</b>	Short Message Service Centre
<b>SOA</b>	Service Oriented Architecture
<b>SQL</b>	Structured Query Language
<b>SMS</b>	Short Message Service
<b>TDOA</b>	Time difference of arrival
<b>TOA</b>	Time of arrival
<b>UCM</b>	Use Case Model
<b>UCS</b>	Universal Character Set
<b>UML</b>	Unified Modelling Language
<b>USSD</b>	Unstructured Supplementary Service Data
<b>USSD</b>	Unstructured Supplementary Service Data
<b>UTF</b>	Unicode Transformation Format
<b>W3C</b>	World Wide Web Consortium
<b>WFS</b>	Web Feature Service
<b>WML</b>	Wireless Markup Language
<b>WQI</b>	Water Quality Information
<b>WPS</b>	Web Processing Service
<b>XML</b>	Extensible Markup Language

# Chapter 1

## Introduction

Geographic Information (GI) is being used for a vast array of purposes by a number of individuals in Africa. This need for geoinformation has resulted in the creation of SDI for the dissemination of such data. Timely access of geographic information is necessary but remains inadequate especially in the southern contexts [5]. The SDI nodes are built based on standards defined by standardisation organisation, for example, Open Geospatial Consortium (OGC) and International Organisation of Standards, like the ISO 19100 family of standards. These SDI nodes disseminate GI mainly via the Internet. Statistics on the internet access suggest that it remains a great challenge for people in the ‘Southern Contexts’ to fully harness the potential and use of the SDIs [26] due to the limited access to the internet. In the same geo-space it can be noted as well that another technology exists which is ubiquitous, and this is the Mobile phone technology.

Current SDI technology is not fully adapted for scenarios where there are limitations in internet access, possibly due to the reason that the societies are technologically challenged or the user is mobile. Another possible challenge is the technical human resource base for extensive Information Communication Technology (ICT) developments [11]. In many nations including developing economies, the challenge is to find spatial information given the scarcity of various resources, like internet access, besides human resources. Such information can be provided through the use of Mobile phone technology by linking it with an SDI node. Since mobile phone technology is omnipresent, the linkage may just be the way to reach out to a much wider user audience. Hence, the 160-character SMS technology is a logical and even necessary platform for building ICT applications for mobile scenarios in challenged societies.

### 1.1 Motivation and Problem Statement

A Short Message Service (SMS) on a SDI<sup>1</sup> node can send messages (sometimes in bulk) to targeted users, and receive messages from respondents. Based on OGC specifications and guidelines, architectural designs are implemented

---

<sup>1</sup>‘SDI-information system – normally hosted by an organisation – which provides geoinformation services (geoservices) to an environment outside of the organisation’.[21]

to create web-enabled systems that together form an SDI (the so-called SDI nodes). Coupling these web-enabled systems with SMS systems through a gateway GI can be exchanged through queries to and from an SDI node.

The text message queries use robust search engines that allow two-way, interactive text messaging applications for searching information based on simple texts. These search engines can be used by mobile phone applications using SMS to provide service delivery [32]. Application Programming Interfaces (APIs) provided by mobile frameworks like the Java 2 Micro Edition (J2ME)<sup>2</sup> allow for the creation of sophisticated applications for such queries [43]. Queries can be done using Wireless Markup Language (WML) for data integration which is implemented as an extension of Extensible Markup Language (XML) [14].

In an African perspective, the density of mobile phone usage is greater than that of the Internet due to various developmental reasons [26] hence, the use of the mobile phone versus the internet is greater. According to the World Bank, 77% of the world's population already lives "under the footprint of a mobile network" [37]. Consequently, information can be seen to be highly accessible via the use of SMS services. Furthermore, a lot of services can be implemented to work using SMS for information interchange.

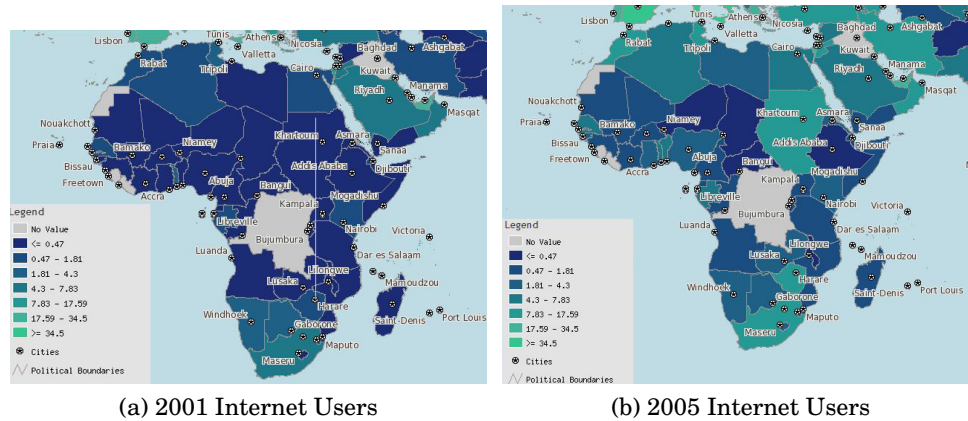


Figure 1.1: Internet Usage comparison from 2001 to 2005. The statistics are computed per 100 people (*source: [62]*)

The number of internet users in the last decade has increased, Figure 1.1, during the same period there has been a greater increase of mobile phone usage, Figure 1.2, for many African countries. The country statistics in Appendix G reiterate this showing by region the variations. This goes to show that there is greater potential in developing solutions that are mobile phone based compared to internet based solution.

<sup>2</sup><http://java.sun.com/javame/index.jsp>

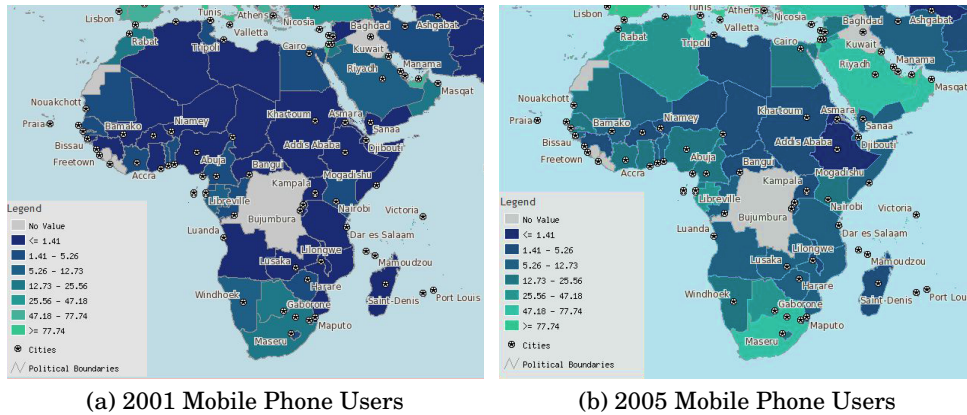


Figure 1.2: Mobile Phone Usage comparison from 2001 to 2005. The statistics are computed per 100 people (source: [62])

Therefore, the employment of SMS services involving an SDI node has considerable positive implications. Societies in developing countries are reported to have a high growth in mobile phone usage [53]. In developing societies, there is a lack of means to provide spatial information to and from SDI nodes using almost omnipresent technologies. Mobile phones can alleviate in information services to SDIs. Due to the need for real time information update and retrieval from SDI nodes, for instance, status update and process inputs, there is a crucial need to define and implement architectures for two-way messaging that incorporate ubiquitous technologies such as SMS services. Hence, the lack of accessible and low cost solution SDI interfaces can be resolved. Also, the solution can be found for the limitations caused by lack of instantaneous, updates and retrieval of spatial data from SDI nodes.

A possible *use case* scenario is one where farmers need or provide information about water bodies in their farms. A spatially informed SMS system could be used to provide this kind of information. A farmer then sends a normal text message to a predefined number asking for the quality of water of a specific pond. This SMS is transmitted through the network via the nearest cell tower to the Short Message Service Centre (SMSC). At the SMSC it is received and forwarded to the receiving number for processing then subsequent response via the same channel. Considering the region of implementation, a working assumption is the use of Second Generation (2G) phones, primarily due to the level of accessibility as over 80% of the mobile phones are reported to be 2G phones [43]. This general use case works but several technical challenges exist to implement such a system. The challenges are user and Short Message Entities (SME) related and typically are:-

- How to deal with a message recipient who is not literate,
  - How can the recipient understand the message?
  - How can the recipient send a coded message, basing on the vast literacy level?

- System users co-share mobile phone, given the southern contexts, scenario exist whereby the user of the system is not the actual owner of the mobile phone.
- Delays in message delivery, GI database updates, for instance poisoned water sources, need instantaneous transactions but SMS delivery delays can occur. How then to deal with this?
- How to position user and how accurate? The low-end mobile phone rarely comes with Global Positioning System (GPS) functionality and as such a challenge would also be to position the user, if positioned then how accurate is the positioning?
- Incorrect message syntax or semantics, stemming from the user the messages can be incorrectly typed, then a challenge would be how the system would be able to deal with such an event.
- Variations in cultural perspectives.
- Registering individual user profiles, for each user there exist a relevant context and issues of privacy. Fortunately, the 2G phone supports Unstructured Supplementary Service Data (USSD) for secure-sessions.<sup>3</sup>

Given this background, solutions to the challenges are of paramount importance and as such this research explores how such information can be disseminated using SMS and mobile phones. The research would then be of interest to people from developing countries and to geospatial information users who require instantaneous updates to and from the SDI node.

## 1.2 Research Identification

The overall objective of the research is to develop architectural solutions for two-way SMS services on an SDI node. This can be refined through the following research objectives and questions.

### Specific Objectives

1. To design an architecture of a two way SMS system involving an SDI node.
2. To develop software components to generate, receive, analysis, send, translate and store spatially informed SMS, using low cost hardware and software solutions.
3. To build a prototype system to implement SMS spatial information generation, receiving, sending and analysis.

---

<sup>3</sup>[http://en.wikipedia.org/wiki/Unstructured\\_Supplementary\\_Service\\_Data](http://en.wikipedia.org/wiki/Unstructured_Supplementary_Service_Data)

### Research Questions

1. What are the architectural components for an SMS system for handling spatial information on an SDI node?
2. What software and hardware is needed for low cost SMS messaging of spatial information?
3. What is the appropriate format for sending spatial information through SMS to allow for extendibility of current systems in a generic manner?
4. What message characters can be used to send the spatially informed SMS?
5. How can a Spatial Database Management System be integrated in an SMS system?

### Innovation Aimed at

This research has the aim of defining software architecture and to propose the technology required for creating a two-way SMS service involving an SDI node. A number of options will be looked at, with the focus of applying open source solutions to refine the architecture. The backbone of the process is the two-way provision of spatial information in a compatible SMS format from a relevant Spatial Database Management System to and from SMS clients.

## 1.3 Method Adopted

The research project is to be executed in three phases. The initial stage was a literature review and a scan of ongoing projects in a two-way SMS services. This mainly utilised the literature review approach. It involved scanning and looking at the market for modern trends in SMS service delivery and standards in message services for the location-based systems and the appropriate data formats. This is mainly an analytical search for the identification of suitable and available open source technology, which would be implemented for the architecture for two-way SMS services on an SDI node. A subsequent stage in the literature review followed to define and derive the possible SMS and SDI functionality and architectural components.

In the second stage, a use case created using the systems approach to design the architecture of the two-way SMS system. we the break down into structural elements, components, subsystems and sub-assemblies. The appropriate break-down would support the Service Oriented Architecture (SOA) [38]. A potential scenario is that of agricultural farmers in a developing country that provide and require information on specific water bodies. In this instance, the system was split into modules which has the following functionality.

- An SMS Analysis Module. This module parses the SMS before it is sent to the user sent or received from user for errors, content and its semantics. If received it is ultimately updated to a database or perform a specific task.

For example, if a user sends water quality information about a water pond. It would determine which pond the user refers to geographically and what the user wants to imply or say with the message.

- The interface, this manages or combines the various modules and the components in the system.
- An SMS sending module. This encodes the spatially informed SMS in a manner that allows it to be sent as a normal 160 character SMS. For instance, information about the water quality and other related spatial information can be encoded and sent to the user as a broadcast message or as query result.
- An SMS reading module that decodes the message which is sent by the user. Special attention is made to this part as the audience is of a vast array of background.

Finally, as a proof of concept (validation), this study realises a designed architecture(s), by implementing a demonstrator system.

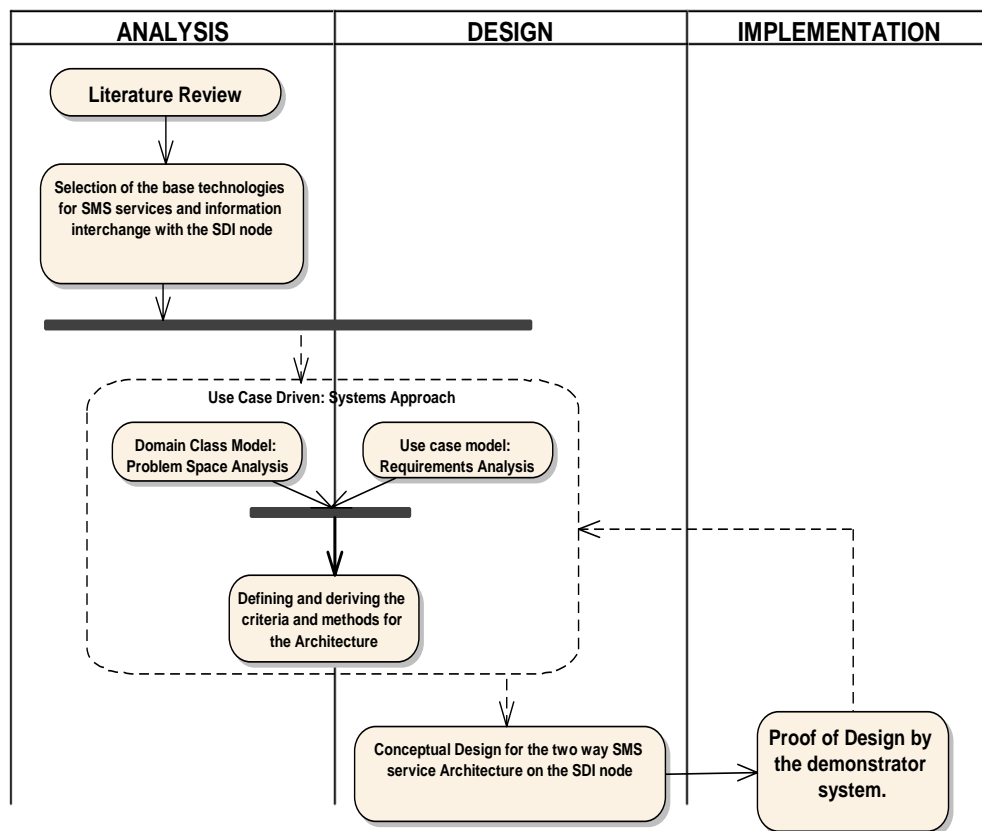


Figure 1.3: Research Approach

The Figure 1.3 illustrates the stages involved in the research work.

## 1.4 Structure of Thesis

The thesis is structured into six chapters:

- **Chapter 2** This chapter reviews the the state-of-art in the Short Message Services and the Spatial Data Infrastructure.
- **Chapter 3** This chapter identifies the system functionalities of Short Message Services
- **Chapter 4** This chapter introduces the use case to carry out the design for the architecture for the Two-way messaging system.
- **Chapter 5** This chapter presents the implementation of the designed architecture and is presented as a use case.
- **Chapter 6** This chapter focuses on the conclusion of the research.



## Chapter 2

# Overview of Short Message Services and Spatial Data Infrastructures

### 2.1 Introduction

The GSM Phase 1 European Telecommunications Standards Institute (ETSI)<sup>1</sup> introduced the service technical specifications and the technology standards which were subsequently handed over to the 3rd Generation Partnership Project (3GPP<sup>2</sup>). The widely ubiquitous technology has 100% usage from GSM handsets and the majority of GSM networks [45]. The GSM framework (GSM 03.40<sup>3</sup>) uniquely allows computers, or in this case phones, to communicate with each other without the need of a central hub. From this the service we are interested in is called the SMS. This chapter gives an overview of the SMS service and also gives an overview of SDI and its capability in section 2.5.

### 2.2 Short Message Services: What are they?

The (GSM) has a communication service which uses standardised protocols that allows for the interchange of short text messages between, normal mobile telephone devices. This standardised communication has brought about the text message popularly and internationally as SMS. This service gives a provision for text messages to be sent from one mobile phone to another mobile phone or from the Web to a mobile phone via the SMSC. Maximally 160 characters (including spaces) can be sent to and from the GSM mobile handsets. In principle this would be 160 normal characters or 1120 Bits of data, where a normal ASCII character is 7 bits. Historically this service has been in place since the 1985 as stated in the GSM series of standards [63].

---

<sup>1</sup><http://www.etsi.org/>

<sup>2</sup><http://en.wikipedia.org/wiki/3GPP/>

<sup>3</sup><http://www.3gpp.org/ftp/Specs/html-info/0340.htm>

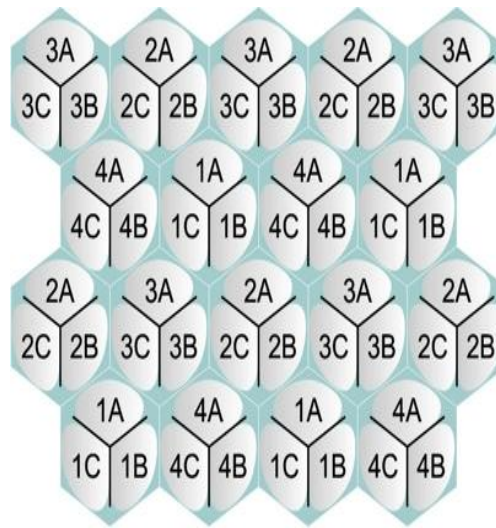


Figure 2.1: General setup of GSM cells which have homogeneous distribution and size *source: truteq.co.za*

### 2.2.1 How SMS's generally work?

With SMS, phones can locate each other; send short packets of information back and forth, through a cell phone tower over a pathway called a control channel. The GSM network creates a mesh of cells which cover entirety of the coverage area as in Figure 2.1 where a base station or repeater would denote each cell. Given this at any moment the network service provider's system would be in a position to state the exact cell in which mobile phone would be in. This occurs through data packets which are exchanged constantly between the mobile device and the surround cells. In the event of a text message being sent or call being made the communication is made through the cell that the mobile device would have reported to be residing in at that instance. To describe this, hypothetically, when a farmer X sends Y an SMS message for some known geo-service, the message is transmitted through the base station/ tower of the residing cell, flows through the cellular network operator's SMSC, then to the tower of the user or system Y, subsequently the tower at Y's end sends the message to the phone as a little packet of data on the control channel. This communication channel is not constantly active/ available as in fixed lines but is setup only when there is data or a message to be transmitted. A consequence of the infrastructure not using fixed lines is the limitation on the message size (data packet). The actual data packet for the message includes items like the length of the message, a time stamp, the target phone number and the format type.

### 2.2.2 Where are they being used?

The GSM 03.40 intended the SMS service to be a way of allowing for the interchange of a restricted amount of information between mobile entities on

the GSM network. Stemming from a few factors like the data size limitation, ubiquitous use and availability of the technology several application areas have been identified and are being exploited further. Le Bodic (2005) reported general categories where the SMS service can be used and these include consumer applications, corporate applications, and operator applications [45].

### Consumer Applications

- *Person-to-Person Messaging*, these would be for general message usage. For example, a Farmer requesting assistance from the an agricultural services officer in the form of a text message exchange or even a message amongst peers.
- *Information Services*, an information service would be a system application that is prompted to send a text when an event occurs with certain information. An example would be a rain status information service, a farmer can prompt an information service provider to send information on whether it would rain using a known text messaging protocol. The service would provide such information upon the reception and processing of such an SMS request.
- *Voice Message and Fax Notifications*, in this case a server sends an SMS to a specific number(s) in the event that a Fax has been received or a voice message has been saved on the user's number.
- *Internet Email Alerts*, this work primarily in the same manner as the Fax notification, and when an email is received an alert via SMS is sent to the email account holder's number.
- *Download Services*, this type of SMS is used when an individual is alerted of the readiness to commence a download if a queue was in existence or a process was occurring and a time wait was necessary. An example would be specific Web Processing Service (WPS) was started and when it runs to completion it can alert the user via text that data is ready to be downloaded via say a Web Feature Service (WFS).
- *Chat Applications*, these allow user to exchange several messages much like person-to-person SMSs, some allow group(s) of people to exchange SMSs interactively. Generally in such an application the SMSs, are sent, received and displayed on the mobile device in order of the date and time.
- *Social Networks*, an example here is twitter, which enables its users to send and read messages about virtually any event or occurrence in the social scene.

### Corporate Applications

- *Vehicle Positioning*, given that a mobile device can locate itself via several position technique which can include cell-id based. The location of the device which can be attached to a vehicle can be reported to the requesting

system or application thus individuals and or organisation are in a position to monitor and locate their vehicle in any area with mobile phone coverage.

- *Remote Monitoring*, this service can be built to allow GSM mobile device or GSM enabled computerised system to be able to monitor remotely with possible capability to and the systems and equipment which is based at a particular location remotely. SMS messages would be sent to change system parameters, turn on or off the equipment, request status updates and create alarms or alerts.
- *Mobile marketing*, through a central system a marketer can send a broadcast message to numbers that are accepting or registered for the service and market his/ her products or services.

#### **Operator Applications**

- *SIM Lock*, this allows the network service provider to lock/ unlock a particular handset to a specific network.
- *WAP Push*, are widely used for pushing application services and data services, for example polyphonic ringtones and wallpaper images, to the mobile phones.

The list is not exhaustive but serves to portray the vast array of application areas that can be employed through the use of SMS.

## **2.3 What can be sent via SMS Formats?**

Generally any text character can be sent via SMS but with limit on message size as alluded to earlier. For an SMS the message length is limited by the constraints of the signaling protocol to and is reported to be precisely 140 octets [32, 45]. This translates to 1120 bits of data. The SMS can be encoded using a variety of alphabets with the default alphabet being the GSM 7-bit alphabet and stated in the GSM 03.38 specifications. There also exist several alphabet options like the 8-bit data alphabet and the 16-bit Unicode Transformation Format (UTF)-16 alphabet. The alphabet denotes the number of possible characters, for example we state that an SMS is composed of *maximally* 160 characters because the GSM 7-bit alphabet would be in use. This can also translate to maximally 140 if 8-bit characters or 70 if 16-bit characters. Thus we can conclude that any character within any specified alphabet can be used [45].

## **2.4 SMS modes, how do they work?**

Generally there exist two major modes of text messaging for sending and receiving SMSs. These modes are:

**Text mode** Is a character protocol that when used is mainly based on the Hayes AT (AT) set modified for GSM. In automated environments it is described as a mode that is mainly suitable for *unintelligent systems*, and for application software built on command structures [57]. This mode is an encoding of the bit stream represented by the Protocol Description Unit (PDU) mode. The message alphabets may differ and there exist several encoding options for the display of the SMS message. When a message is received the device would display the message in a proper encoding. In text mode, the application is bound to (or limited by) the set of preset encoding options [57]. When a message is sent the message is received in plain text. The mobile phone would construct a data unit when this occurs. Refer to Table F.1 for a list of data unit types.

**PDU mode** Receiving a message in the PDU mode implies that the PDU string contains several components. These components are the message string and a lot of meta-information like the sending SMS service centre, the time stamp to mention a few. This message would be in the form of hexa-decimal octets or decimal semi-octets [12, 45]. If PDU mode is used, any encoding can be implemented.

In general any device or application capable of reading incoming SMS messages would be able to use the text mode or PDU mode.

## 2.5 Spatial Data Infrastructure: What is it?

### 2.5.1 Concepts and Components

According to the Global Spatial Data Infrastructure Association (GSDI), “The term Spatial Data Infrastructure is used to denote the relevant base collection of technologies, Infrastructure policies and institutional arrangements that facilitate the availability of and access to spatial data” [3] as illustrated in Figure 2.2.

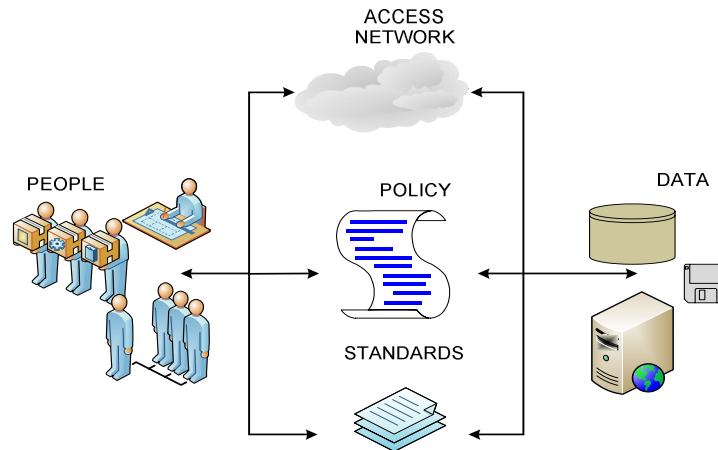


Figure 2.2: The components of an SDI (*adapted from: [58]*)

Intrinsically, to be in a position of stating the existence of an SDI, there has to be more than a single dataset or database. According to the standards and specifications as defined by the International Organisation of Standardisation (ISO)<sup>4</sup>, World Wide Web Consortium (W3C)<sup>5</sup> and the OGC<sup>6</sup>, an SDI hosts geographic data and attributes, metadata, a discovery services, visualisation and evaluation means and a methodology to access the geographic data [3]. This mainly falls under the technical aspects of an SDI. Beyond this are additional services or software to support applications of the data. To make an SDI functional, it must also include the organisational agreements needed to coordinate and administer it on a local, regional, national, and transnational scale. Although the core SDI concept includes within its scope neither base data collection activities or myriad applications built upon it, the infrastructure provides the ideal environment to connect applications to data, influencing both data collection and applications construction through minimal appropriate standards and policies.

This research mainly dwells on the data and technology aspects of the components of the SDI when looking at the SDI node in implementation.

---

<sup>4</sup><http://www.isotc211.org/>

<sup>5</sup><http://www.w3.org/>

<sup>6</sup><http://www.opengeospatial.org/>

## 2.6 SDI Architectures

The architecture of an SDI node varies depending on organisation but it can be noted that most adhere to some form of service-oriented architecture as most would want to promote the general principles of services, interoperability through an enterprise service bus, and loose coupling. The service is defined as a “piece of self-contained business functionality” [38] and from Figure 2.3 it can be seen that the components are not tightly coupled to each other and in order to interact there is a level of interoperability which is employed. The figure shows the general architecture but it should be noted that the software indicated are there to serve as examples

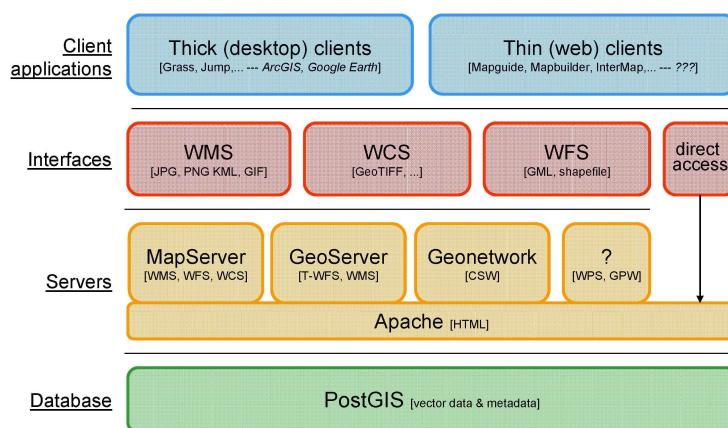


Figure 2.3: The SDI Software Architecture (*source: geonetwork-opensource.org*)

## 2.7 Overview of SMS technologies in the southern context

Several studies have been conducted in relation to the use of SMS service delivery. Donner (2008) reflects on the several projects, for instance the *Cell Life* project and *Project Zumbido* [23]. These projects explored various options to provide information to the user using an SMS platform with web-based or self contained applications [23]. For these, the domain area of application portrays that they are being used to communicate non-spatial information. Thus, the provision and the dissemination of spatial information to and from an SDI node still remains a challenge.

Emanating from the current SDI standards, developments need to be made to allow for mobile GIS services, for example, spatially informed SMSs [52]. Implementations like the “almost All QUESIONS Answered” (aAQUA) [59] tend to only provide non-spatial data and also are mainly using proprietary software solutions.

Mapham [50] further highlights the use of SMS services to provide health care information on a two-way system using messaging software called Zygo-HUBS. It is an implementation that is not built on specific information standards, though it sets a firm foundation for defining a platform for two-way messaging services. Seamless integration is also an issue in the use of mobile devices.

In [13], MySMS is reported to be built under some of the principles of this research, which are centred on the southern contexts as for application area. Campbell [13] also highlights the applicability of SMS for dissemination of information from various sources, though not necessarily spatial information through an SDI node.

FrontlineSMS [7] is an open source solution for the provision and creation of a messaging focal point through a computer for receiving and sending SMSs on a two-way communication basis. It demonstrates applicability of solutions that do not require the Internet as a direct medium of communication, and which can be said to be “webless”. This can be useful in southern contexts where Internet availability is not easily ascertained.

SMS services in the open source environments are built using open source languages, thus extending their applicability. Various approaches to designing such solutions are explored as in the Mobiles for Development (M4D) landscape. The M4D landscape is a consortium for open source solutions for mobile devices.

Apart from the above stated projects, there exists a wide range of projects employing SMS technology for message delivery. For the purpose of this research a market scan was conducted yielding a general functionality description of available SMS systems and their operations. Particular attention was given to innovations that are mainly for the southern contexts or developing nations.

A general list and an overview of the projects is given as in Appendix C.

## 2.8 Conclusion

This chapter explored the various aspects of the SMS and the SDI. It gave an overview, of the characteristics of an SMS, the message formats and where they are being used. Another aspect is the SDI in which we explored its general architecture. Lastly we provided a general view of where the SMS services are being used and how. A comprehensive market scan was introduced and detailed as in Appendix C.

## Chapter 3

# Short Message Service System Functionality and Issues

### 3.1 Introduction

In this chapter we look at the basic functionality of a system that processed SMSs. We also look at the potential aspects that are necessary to allow communication that involves an SDI node. The synthesis is drawn from the application scan done in Section 2.7. From this we obtain information on how each component works and the alternatives to it.

### 3.2 Positioning

Although Short Message Services are location independent, in many instances the location aspect is an important characteristic. Information to and from any Geospatial Dataset require a location aspect. As such, the two-way transmission of SMSs involving an SDI node is no exception. On the 3G, the location of the device is important as [65] writes, the 3G cellphone device has numerous location-dependent functions. The same level of importance can be extrapolated to low-end mobile devices, in the context of this research as geospatial information is crucial. Hence we look at how to obtain locational information as an integral functionality. Considering the mobility of the mobile phone, there exist several scenarios pertaining to location:

- There is need to determine the location of the device in order to provide the information. In this context, the mobile phone location would be directly related to the subject of interest. So, if we locate the mobile device we are also locating the subject of interest.
- The device might not be on or near the subject of interest, therefore, there is a need to locate the subject of interest separately.
- The device in some instance does not belong to the person in the subject of interest, but to another individual or to a community of users. Therefore in that case we would need to locate the subject of interest separately from the device.

Several questions can arise regarding these scenarios and we look at possible solutions, advantages and limitations.

#### 3.2.1 Positioning the mobile phone

Location technologies are principally divided into three categories. Methods to position the device are inherently derived from these principal technologies [66]. The three technologies are Stand-alone, Satellite-based and Terrestrial-radio based. These are exemplified by dead reckoning, GPS, and “C” configuration of the Long Range Navigation (LORAN-C)<sup>1</sup> system respectively. We discuss each of these below.

##### Dead Reckoning

“Dead Reckoning (DR) is the process of estimating one’s current position based upon a previously determined position, or fix, and advancing that position based upon known or estimated speeds over elapsed time”.<sup>2</sup> Though this technology was mainly used in navigation systems prior to the era of assisted-GPS, it is still an implementation possibility to locate a mobile phone. To obtain the position from a previous point the heading and distance would be required to compute the present location. Mathematically, this is given as

$$x_k = x_0 + \sum_{i=0}^{k-1} s_i \cos \theta_i \quad (3.1)$$

$$y_k = y_0 + \sum_{i=0}^{k-1} s_i \sin \theta_i \quad (3.2)$$

The  $x_k, y_k$  would be the position at time  $t_k$  from  $x_0, y_0$  given  $s_i$  and  $\theta_i$  as the length and heading of the device. A disadvantage of dead reckoning is that cumulative errors occur as the new positions are calculated solely from previous positions hence, the error in the position fix grows with time.

##### Radio-based Technology

Radio based technology uses devices that transmit radio signals, for instance base stations, satellites. The signal can be bi-directional, that is, from the mobile device to the base or conversely [65]. Essential components of the above require radio transmitters, receivers, or transceivers to provide locational information. The main assumption in this technology is that one end in the positioning system is stationary and is at an absolute, known position. A crucial aspect in its implementation is that the positioning function can be performed at either the fixed or the roving end [65]. Presently, this can be implemented using three ways discussed below.

---

<sup>1</sup><http://www.navcen.uscg.gov/Loran/default.htm>

<sup>2</sup>[http://en.wikipedia.org/wiki/Dead\\_reckoning](http://en.wikipedia.org/wiki/Dead_reckoning)

**Angle of arrival (AOA)** This method of positioning works by determining the direction of propagation of a radio-frequency wave incident on an antenna array. These antenna arrays would be fixed at the base stations. An Angle of arrival (AOA) measurement ties the source location along a line in the direction of the angle that joins the base station and the mobile phone or the user equipment. This is called the Line Of Bearing (LOB). This technique requires a minimum of two base stations to obtain a location through a form of triangulation as in Figure 3.1. In the event of multiple base stations being available and that the AOA is measured to these, the location estimate of a source is obtained as the intersection of the LOBs. Errors in this case would result in the LOBs not intersecting at the same place. Error compensation techniques must be employed to resolve the ambiguity created [22]. For AOA-based location, antenna arrays are used to measure the direction and these are put in a receiver. In this case, the receiver would be the base station. This is because of the limitations of the mobile phone and also due to mobility issues [56].

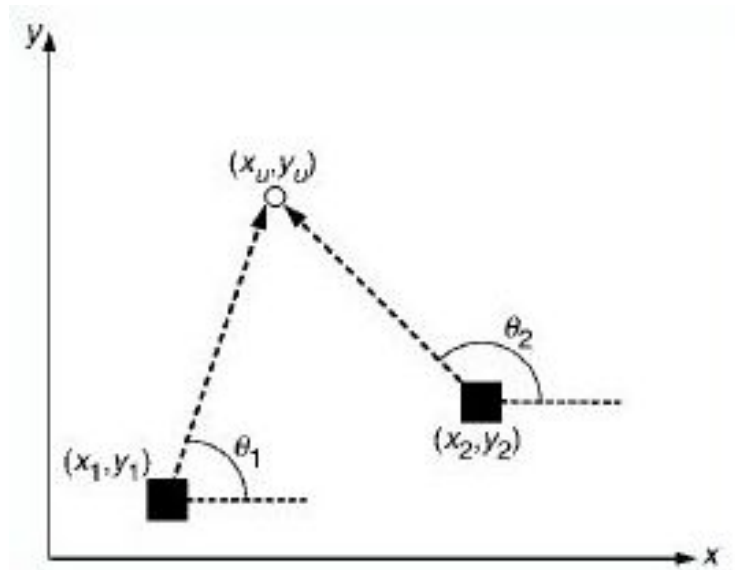


Figure 3.1: Base stations, user and angles measured to obtain user position (source: [22])

**Time of arrival (TOA)** Also known as time of flight, and it refers to time taken by a radio signal to travel from a transmitter to a receiver, which in the context of the mobile phone, would refer from the base station to the mobile phone. The basic principle evolves around intersection of range circles as in Figure 3.2. The relationship between the time taken by radio signal to travel, and the speed of the signal, that is, the speed of light is expressed by  $c$  proportional to  $\frac{1}{t}$ , where  $c$  is the speed of light and  $t$  signal travel time. The product of  $c$  and  $t$  results in the range of the mobile device to the base station antenna. Having two range measurements would then provide an ambiguous fix of the location but having a third range would then result in a unique posi-

tion. This positioning technique principle is also employed in GPS [65].

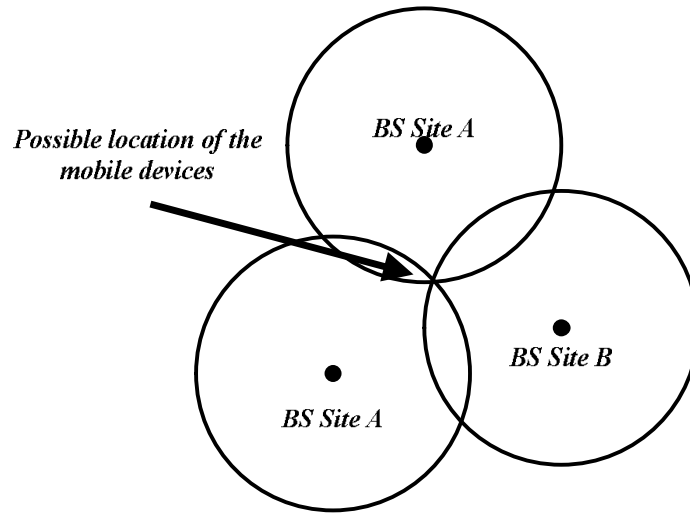


Figure 3.2: Base stations, users and angles measured to obtain user position using Time of Arrival (*source: [65]*)

**Hyperbolic positioning** is a position technique which is also known as Time difference of arrival (TDOA) or Multilateration. It uses trilateration in its positioning. As shown in Figure 3.3, absolute time measurements are not used but rather time difference measurements. The time difference is converted to constant range difference to two base stations. This defines a hyperbolic curve with a location fix being obtained by the intersection using at least three base stations in the 2D scenario.

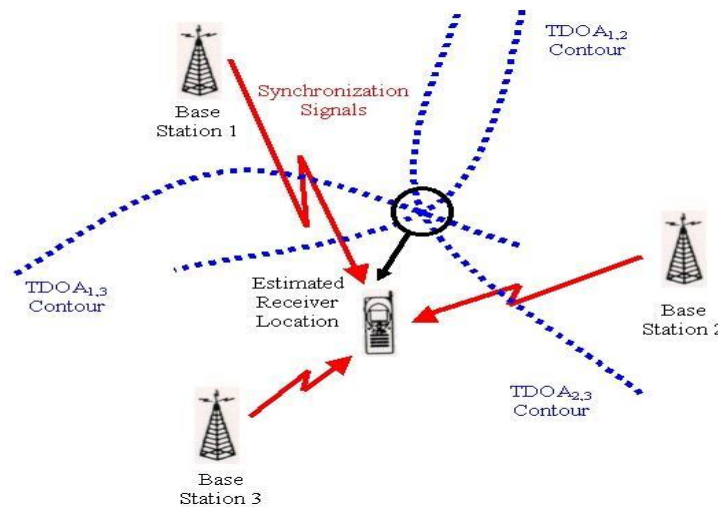


Figure 3.3: Base stations, users and angles measured to obtain user position using Time Difference of Arrival (*source: [41]*)

### The Cell-ID-Based Positioning Method

Works under the assumption that the mobile device is always connected to the closest base station, its cell, as in Figure 2.1 in Chapter 2. Generally, a service provider has base stations that form the network [65]. When in motion a mobile device changes its resident cell. A mobile device can thus be located by determining the cell that is residing at a particular moment. Each cell has a cell Id and thus can be associated to a specific location. The accuracy of this positioning technique is variable and less accurate. This is because the cell coverage does not remain static and can shrink. This method has an accuracy level directly related to the radius of the cell that the mobile device would be resident in [65]. The method does not require any special equipment and is free of charge.

In several variations the information sector can also be made available as in Fig 3.4. This would aid in enhancing the positioning accuracy of the Cell-Id method.

### The Assisted GPS Positioning Method (AGPS)

By design, GPS was not for use indoors or in urban areas. Improvements of GPS performance in such environments can be made by linking mobile receivers to a cellular, bluetooth-based, or wireless local-area network infrastructure that has a reference receiver. This form of positioning is called assisted GPS. A reference receiver provides navigation and signal timing data to a location server, which relays this information to GPS-enabled user equipment, for example, a mobile phone. Basic GPS measurements which are preprocessed along with statistical measures are returned by the mobile phone. These characterise the signal environment and are sent to the server, which performs a series of complex calculations on data received from the client to determine the user's position. This

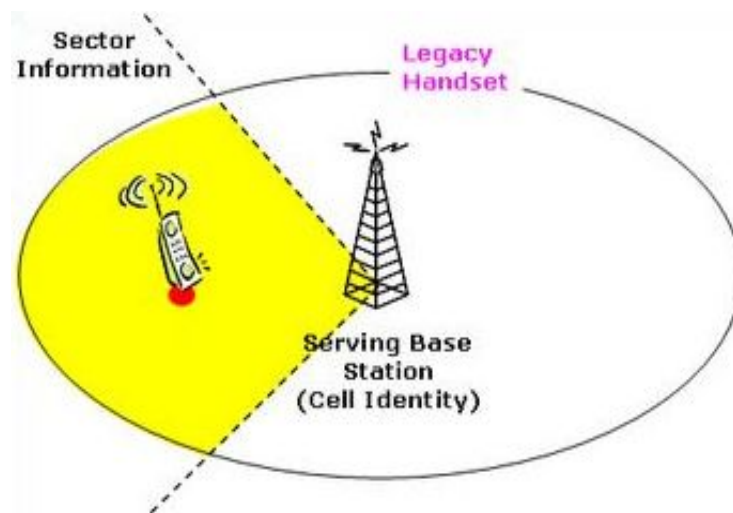


Figure 3.4: Information Sector from *cell-id* positioning. A notable aspect about this positioning technique is that it can be used by even *legacy handsets*. (source: [www.commscope.com](http://www.commscope.com))

process, through distributing data and processing, has an additional advantage in that it optimises air-interface traffic. Considering that the network assumes the burden of measurement calculations, it implies that the actual implementations on the mobile phone can be more cost effective and easier to implement [6]. Notable advantages of Assisted Global Positioning System (AGPS) are that it provides relatively greater accuracy than standard GPS, given that it is reported to have accuracies of less than 50 feet for outdoor environments and less than 50 metres feet for indoor environments [6]. Privacy issues can be advantageously enforced if the user or network provider can restrict information or service, respectively.

#### Other positioning methods

There also exist other positioning methods under research, most notably the use of radio signals, such as TV or AM/FM radio broadcast signals, in place of cellular or satellite signals. An advantage of these is in availability of radio signals compared to the cellular signal [65].

#### 3.2.2 Positioning the subject of interest

Several techniques exist in which we try to locate the mobile phone, but there are scenarios in which we need to locate the subject of interest. More often than not, in southern context, there is the aspect of shared mobile phone. That alone could mean that the mobile phone is not directly related to subject of interest does not imply the location of the subject of interest.

Given this scenario, the only way to identify the subject of interest would be by the direct reporting of the subject of interest by the user. The user would

enter in the message string the location of the subject of interest. This then is a solution to one scenario, where the phone is not linked to the subject of interest.

For a shared mobile device, there is need to identify the user first for user specific applications. In this case, one can be provided a unique identifier in the form of a name or number. This would serve as a primary key for the user details. The user then provides the identifier together with the location as a text string. This then allows for user of a shared mobile device.

### 3.3 Message Parsing

Message parsing, is an application process carried out to split up a text, which is built up as a continuous stream of characters, into tokens. A token being a segment of text. From the token generation, we then build a grammatical structure represented as a tree data structure, known as a parse tree [46]. There exist two ways to parse through text, top-down parsing and bottom-up parsing, where a parse tree is created from the top and from the bottom reducing up respectively [28].

From the token generation, syntactic parsing or syntax analysis follows. This analysis checks that the tokens form an allowable expression in reference to a context-free grammar [2, 28]. This grammar defines components that can make up an expression and the order in which they must appear. After the syntax analysis there is the semantic parsing or analysis. This is where the implications of the expression which has been validated by the syntactic analysis are extracted. If the parsing was being done by a compiler it would then generate the machine language that performs the functions that are described in the code.[2, 28]

### 3.4 Alphanumeric Data Input Errors

The principal workings of SMSs dictate that short texts are sent from one mobile phone to another. A direct result is that in many instances the user has to key in the text. As is well-known, no such system is error free and as even the text typing on the mobile phone can introduce errors. These errors are called “*Lexical errors*” and are subsequently be divided into the following categories:

- Misspelling (typographic errors or not)
- Segmentation errors (these occur due improper joining or disjoining of words)

In this context of typing text messages on mobile phones misspellings for the purpose of this research would be defined as the transformation of the word as it should be spelt [35]. This can be due to one or more of the following basic operations:-

- deletion (e.g. ‘mothr’)
- insertion (e.g. ‘motther’)

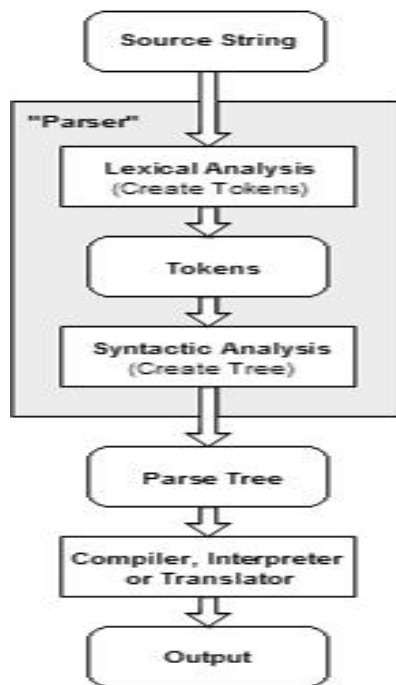


Figure 3.5: Text Parsing System *source:wikipedia*

- substitution (e.g. ‘mothar’)
- transposition (e.g. ‘mohter’)

The above operations are not primitive operations. It should be noted that they do not provide a direct means of translation between the actual word and the misspelt word but just serve as a way to describe the error. These are also used to describe lexical errors but are not very good at explaining them. Distinctions are made between typographic errors (performance errors), cognitive errors and phonetic errors (competence errors) [35]. In respect to typographic errors they are defined as mechanical failure or hand slip induced but not due to ignorance.<sup>3</sup> In general it is quite hard to determine the underlying cause of the error; (“erecieve”) for example may just as well be an accidental transposition error, and even if the cause can be established it is not certain that it will help the spelling corrector. Another, more important distinction is the one between none-word errors and real-word errors [35].

- A none-word misspelling is one that results in a string not in the vocabulary (of the system).
- A real-word misspelling is one where a valid (correctly spelled) word is substituted for the intended word.

Several strategies and techniques have been put in place to deal with such errors. According to Ingels (1997) [35] there are three main schools of

---

<sup>3</sup>[http://en.wikipedia.org/wiki/Typographical\\_error](http://en.wikipedia.org/wiki/Typographical_error)

thought in dealing with spelling error correction namely:

**The Classical Method** Damerau [19] reported that approximately 80% of all none-word misspellings were also single error misspellings. From this finding, a minimum edit distance algorithm was implemented. This algorithm works by the process of detecting the errors through a comparison of an existing dictionary and the written word. When an error is detected, the program attempts to convert the erroneous word into a *legal* word based on the assumption that exactly one of the basic error operators is responsible for the error. In the event of an occurrence of a match, the process is terminated and a possible solution match provided. In the event that the error is explained by two single-occurrence operations the process continues in the manner of an iteration until a match is found. Levenshtein's [47] similar technique defines the *Levenshtein Distance (LD)* as the distance between two words in terms of the basic operators deletions, insertions and reversals (transpositions). The LD metric algorithm works by choosing a word in the dictionary that is closest to the incorrect word. Synonymously it is used as the minimum edit distance algorithm, or rather, the term minimum edit distance algorithm refers to both ideas nowadays.

**Noisy Channel Method** This technique is based on the assumption that communication is relayed via an imperfect medium, the channel [35]. It is proposed that a word is inserted in the input end of the channel and the distorted exists from the other end. Given this, the likely correct word is estimated via conditional probabilities. These would be in the form of a set of candidate solutions, each being produced with probability of correctness. If there is also a language model that has information on what words are likely to be inserted, it provides what are known as the prior probabilities [39]. The probabilities are computed with the following formula:

$$Pr(t|c) \approx \begin{cases} \frac{del[c_p-1, c_p]}{chars[c_p-1, c_p]} & \text{if deletion} \\ \frac{add[c_p-1, t_p]}{chars[c_p-1, t_p]} & \text{if insertion} \\ \frac{sub[t_p, c_p]}{chars[c_p-1]} & \text{if substitution} \\ \frac{rev[c_p, c_{p+1}]}{chars[c_p]} & \text{if reversal} \\ \frac{rev[c_p, c_{p+1}]}{chars[c_{p1}, c_{p+1}]} & \end{cases} \quad (3.3)$$

where  $c_p$  is the  $p^{th}$  character of  $c$ , and likewise  $t_p$  is the  $p^{th}$  character of  $t$ . The five matrices are computed with a bootstrapping procedure. This procedure is self-initiating and start internal calculation mechanisms without external help. The result would be the candidate words from which a correction would be made. Using an example extracted from [39], given a word like *acress* and in the word corpus there exist alternatives for the word. In this case seven, the proposed transformations would be as in Table 3.1. Subsequently, the percentage score is obtained by multiplying the prior probability (which is proportional to 1 + *frequency* in the table below) and the channel probability ( $Pr(t|c)$ ) to form a raw score, which are normalised to produce probabilities. The final results are: acres (45%), actress (37%), across (18%), access (0%), caress (0%), cress (0%).

**Table 3.1** Shows an example of the noisy channel method computation results given a word like ‘acress’ to come up with the word with the highest probability

$c$	%	Raw	freq(c)	$Pr(t c)$
actress	37	0.16	1343	55/470 000
cress	0	0	0	46/31 000 000
caress	0	0	4	0.95/580 000
access	0	0	2280	0.98/4 700 000
across	18	0.077	8436	93/10 000 000
acres	21	0.092	2879	417/ 13 000 000
acres	23	0.098	2879	205/6 000 000

It can be concluded that a spell-checker with a basic architecture as in Figure 3.6 which can be employed within a SMS service as a server side application using any one or more of the above techniques for correction.

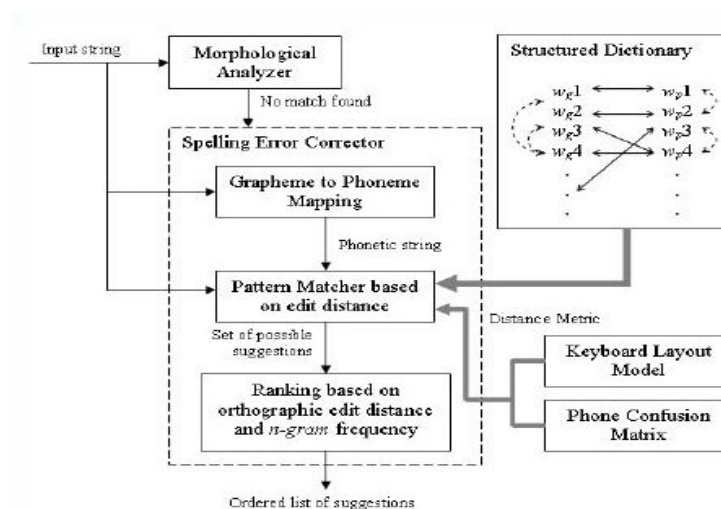


Figure 3.6: Architecture for a spellchecker (extracted from: [8])

### 3.5 Resolving Grammatical Errors

To facilitate proper communication between systems, whether human-machine interaction or human-human interaction, the tokens, syntax and semantics have to be correct in creating a properly functional two-way SMS system involving the SDI node. The rules that govern the communication have to be followed, that is, the tokens, syntax and semantics thus, grammar also plays an important role. To implement an automated check, several approaches have been proposed. A possible solution to this is that of a grammar checker as proposed by Liou (1991) in which an electronic dictionary is built containing the word stems [49]. This has an additional suffix processor to accommodate morpho-syntactic variants of each word stem. An Augmented Transition Networks

parser is used, equipped with phrase structure rules and error patterns [49]. This then would be the basis of the grammar checker and would be used to parse the text thus subsequently detect, report and correct grammar errors if possible. Auto-correction would then be possible depending on the preset conditions within the system to allow the correction.

A structured language form defined as call Controlled English<sup>4</sup> implement a mechanism of grammar checking. These have been used in Natural Language Processing (NLP) and use structured algorithms such as the (ALCOGRAM) [1]. Artwell (1987) in [4] reports of a Constituent Likelihood Automatic Word-tagging System, originally designed for the low-level grammatical analysis of the million-word LOB Corpus of English text samples. A full content parse is not done in this system, it uses a first-order Markov model of language to assign word-class labels to words. Detection of grammatical errors is subsequently done by flagging unlikely word-class transitions in the input text.

### **3.6 Cultural Issues pertaining to the Short Message Service**

Cultural issues play a crucial role in the use of technology by any society [64]. This is primary due to social benefits that might come with the technology and how it affects the culture of the society were it would be presented. The cultural issues are beyond the scope of this research but it is worth while to mention that culture is an issue worth investigation when introducing technology. In particular it is crucial where the main area of concern are people living in the so called 'southern contexts' where culture is very important. [48] reports on how the interaction is demonstrated theoretical layers of interaction.

### **3.7 Service access through USSDs**

USSD is a capability of all GSM phones. Generally a real-time message is sent from a mobile device by transmitting information over GSM signaling channels. Using this service, a user inputs a number code to access a particular service. In short, USSD are used as 'trigger' to invoke services that do not require the SMSC thus has less overhead and additional usage costs. For instance, if a farming community has this service in place. They can send a USSD like \*109\*723# and instantaneously receive rain data coming from an SDI service. The time taken for the trigger event to provide a response in the case of interactive USSD-based services is reported to be shorter compared to SMS. USSDs are used by some payment methods such as M-Pesa<sup>5</sup> in Kenya, thus portraying their secure nature. USSD Phase 1, specified in GSM 02.90, only supports device initiated operation termed pull operations.

---

<sup>4</sup>A subset of standard English with a restricted syntax and semantics described by a small set of construction and interpretation rules (source:[http://en.wikipedia.org/wiki/Attempto\\_Controlled\\_English](http://en.wikipedia.org/wiki/Attempto_Controlled_English))

<sup>5</sup>[www.safaricom.co.ke/index.php?id=745](http://www.safaricom.co.ke/index.php?id=745)

When a USSD is sent through the GSM network, a USSD Gateway routes it from the signaling network to service application(s) and returns the result.

## 3.8 The SMS sending or receiving

To send or receive messages there has to be the hardware solution in the form of a gateway. This send the SMS to and receives it from messaging network. it. In this section we discuss the various possible solutions to this as follows.

1. The first solution is to connect a mobile phone or GSM modem to the system. The message reception would be subsequently handled by the AT commands. A major advantage of this method is that there are no additional costs after setup, like additional fees from service providers apart from message sending charges. On the downside this method comes with restrictions on message numbers as it cannot handle a large amount of SMS traffic. A possible solution to this is to create a system, capable of load balancing on several GSM modems.
2. The second solutions found was through access to an SMSC or SMS gateway of a mobile phone service provider. Given this scenario, when an SMS is received in the mobile service provider's SMSC, the messages enter the system computer directly using a protocol/ interface that the SMSC or SMS gateway supports. An advantage of this message reception option is that it allows for a high messaging rate. Downside of this option comes from direct involvement of the network service provider, which usually result in high cost of setup and subscriptions usually.
3. Lastly, the third possibility found is by receiving SMS messages through a gateway connection of an SMS service provider. This distinguishes from the second option by the ways it can be handled.
  - (a) Subscriber Identity Module (SIM) hosting, in which a SIM card is hosted for the system by the service provider and SMS messages sent to the mobile phone number of the SIM card are transmitted to the system by way of a protocol /interface supported by the SMS gateway of the SMS service provider.
  - (b) Shared phone number, in which an SMS service provider gives a phone number for receiving SMS messages to several different users. Subsequently to associate with the application one has to specify one or more keywords. An example is if truck drivers in Rwanda want road information like tow gates or congestion status, which is all location-specific, they can just text '*road*' to 5556. If a message is received and its contents begins with a keyword specified '*road*', it is forwarded to a specific system using a protocol/ interface that the SMSC or SMS gateway supports.

### **3.9 Conclusion**

The components and their functionality was made leading to an understanding of the expectations of an SMS-SDI architecture. Various aspects like positioning the subject of interest, message parsing were looked at, having done this we now design the architecture in the next chapter.



## Chapter 4

# Design of the two-way SMS Architecture(s)

### 4.1 Introduction

An architecture is a description of system structures. It is a re-usable abstraction that can be transferred to new systems and serves as a platform for communication [15]. In this chapter we design an architecture through a use case, the RISS. The use case shows a set of interactions between external actors and the system under consideration. The use case is described using the Coleman (1998)'s approach [17]. The approach writes in a structured and understandable manner. From the use case present architecture described in Unified Modelling Language (UML).

**Use Case(s) Modelling** A use case model plays the role of describing the proposed functionality of a new system or an existing system. They are used as a means to document requirements, discover and understand the system's behaviour under various conditions, when there is interaction with users (the user can be another system) [27]. Since this research is use case driven, some concepts of the Unified Processing will be adopted. Generally, there are two levels of use case [42]:

**System Use Cases** These document the interactions with software and establish the system's functional requirements.

**Business Use Cases** These are part of large-scale business processes that discuss how a business responds to a customer or an event.

The modeling process comprises of descriptions in the form of text and diagrams which can be generated using a variety of programs or software [27]. For the purpose of this research, we adopted from UP, the Use Case Model (UCM) as a set of use-cases that provides a detailed description of a system from a specific view point. The model establishes functional and non-functional requirements. As such the modelling process involves use case-identification, use case scenario capturing, use-cases to user goal relation and business processes and

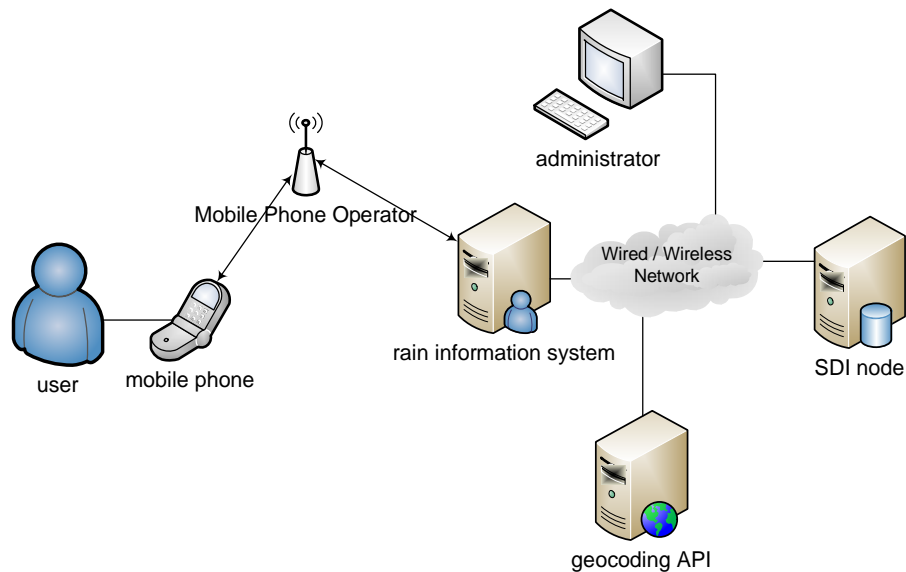


Figure 4.1: General overview of the Rain Information Service

requirement analysis [42]. The UCM is used to provide a means of developing an Architecture for two-way SMS services involving an SDI node.

### 4.1.1 Use Case Identification

A rain information service is identified as a use case. Primarily, this is due to its ability to provide a scenario where information has to be sent to and from geo-service using a communication interaction provided by SMS messages. The reason for this is that it provides spatial information to a specific user via a two-way interaction procedure through an SMS as a means of communication. an overview of the system is shown in Figure 4.1.

### 4.1.2 Use Case Description

The use case concerns an application that determines whether it is raining in an area. It also has the capability of providing a forecast for the same Area of Interest (AOI). The sequence of steps taken to provide the spatially informed SMSs are shown in this section below. The weather information is extracted via the internet from an existing web information service, Buienradar.<sup>1</sup>

An SMS is sent by a user requesting information on rain activity in a particular place, for example, Hengelosestraat 99 then send to predefined number. The message is received by the gateway and routed to the system via email. Subsequently, the email received is parsed and information is sent through

---

<sup>1</sup><http://www.buienradar.nl/>

several modules for executing the query. The query result is sent to the service requester via SMS. The modules in the system are the mail parser, for checking the email; geo-coder, for obtaining the geographical coordinates; the coordinate transformer, converting the geographical coordinates image coordinate; Image extractor, obtaining the pixel value from the image and analyser and the system checks the incoming message and parses it.

The geocoding module attempts to extract the actual geographic location in the message in the form of geographic coordinates. The Google Maps API service is used to extract the location. The extraction process is made possible by the API through using a Representational State Transfer (REST) Interface. REST allows for the use of HTTP GET/POST etc. as an API [40]. From the API call, the output is provided in the form of JSON<sup>2</sup> (JavaScript Object Notation), which is a human readable data interchange format, thus it adequately wraps the HTTP output. Having the geographic location the next step is to find the corresponding image position.

From a precalculated least-squares adjustment fit, image coordinates are obtained. This process occurs in a positioning module that provides the image position in the form of the cartesian coordinates from transformation module.

Image pixel information is extracted using the position information obtained from the transformation module. The pixel information is obtained in the form of a Digital Number (DN), a value assigned to a pixel in a digital image, which is subsequently classified. Classification of the DN value extracted, results in the state of the area of interest being known.

A report compilation module then compiles the report using the message sending number and pre-existing information. The query information and result is stored. Subsequently the report is sent to the user in the form of an SMS.

Apart from the general query to obtain instantaneous information there also exist other scenarios to that the use case follows. These are to register for a service, to cancel a service, to request a place to be monitored and to create a profile

### **4.1.3 Use Case Specification**

The rain information service use case records the behavioural operation of the system. The use case under discussion highlights the activities of the rain information service from the perspective of the two types of user and the information system. Figure 4.2, the use case diagram models the general system context and requirements [42], it illustrates the actors and the use cases and the relationships between them.

---

<sup>2</sup><http://www.json.org/>

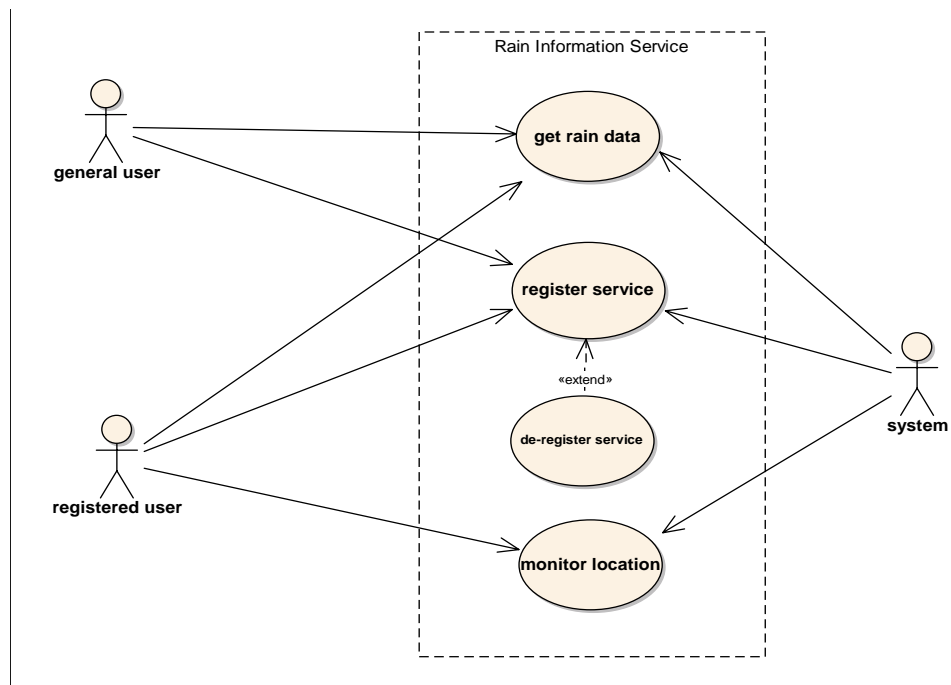


Figure 4.2: use case Diagram of the Rain Information Service

In this specification, a fully dressed description style as in Appendix A is used to describe the use case. It demonstrates a typical scenario where by multiple SMS are processed in the two-way communication with a SDI node. The fully dressed format is chosen as it allows for the inclusion of an assortment of secondary information which can be described using the pre- and post existing conditions in the system [60].

### Use Case: Rain Information Service

#### A. Characteristics

1. Primary Actor: User, who is the person sending the SMS without direct interaction with the system.
2. Goal in Context: User requests or provides spatial information through SMS in the form of a maximally 160 character alphanumeric text message.
3. Scope: *General Public* requesting rain information.
4. Level: Summary.
5. Stakeholders and Interests: General Public

**User** A person or possibly a service that requests rainfall information which is accurate and timely through an SMS sent to a predefined number.

**Rain Information Imagery Provider** A producer of remote sensing information which provides images that contain rain information. This can potentially be pre-classified or could be in raw imagery.

**Geo-coding service** A service for providing the geographical coordinates.

6. Preconditions: Geo-coding service should provide location information for places within a region provided in the rainfall weather data image.
7. End conditions: The actor's request is processed and the rainfall image is correctly calculated.
8. Failed End conditions: The service requestor is not provided with the rainfall information.
9. Trigger: Text message from Short Message Service.

## **B. Main Success Scenarios**

**Step 1** Message from a user is sent to the system via an SMS. It is sent to a pre-existing number requesting for a service. The SMS request can query for information upload or search on specific information using a specific predefined syntax. Depending on local system setting, there would be a delivery report on the success of message delivery to the rain information service's predefined number.

**Step 2** The message parsing component will successfully parse through the the message extract the query. The query information is passed to the interpretation process in the event of a query having location aspect, the information is passed to the geo-coding module.

**Step 3** The parser checks the user profile database to check if the user already exists in the database and attempts to store user information depending on query.

**Step 4** The geo-coding module extracts the geographical coordinates of the area of interest and passes the information to the transformation module.

**Step 5** The transformation module transforms the geographical coordinates into the image cartesian coordinates.

**Step 6** The system extracts the pixel values from the specified cell address and the DN value is classified by depending on the rain condition.

**Step 7** The report compilation module will compile the report and passes it to the mailing module.

**Step 8** The mailing module obtains the phone number and if possible username from the database and it sends the report via SMS and store the query and result in the database.

**Step 9** The hosting service kills the services, and releases the resources.

**Step 10** The user receives the results at proper time and finds the product satisfactory.

### C. Extensions

**Step 1-10a** In the event of any system failure the system will restart and re-process all queries not yet handled successfully.

**Step 1** User is sent an error message requesting a clearer request and will be notified of failure to process the request.

**Step 3a** System continues to provide information assuming, request is an ad-hoc query or there is a problem in database with user profiles.

**Step 4a** Location requested is out of the available rain data image and error message is sent to user of the problem.

### D. Sub- Variations (Technology)

**Step 5a** System should accept a flexible number of image types for the DN value extraction.

### E. Non-Functional

1. Priority: High
2. Performance: High
3. Fault Tolerance: Moderate
4. Frequency: Nearly continuous

### F. Open Issues

1. Explore the remote data archive service recovery issue.
2. Explore Quality of Service (QoS) in terms of performance and flexibility of dataset formats.
3. What type of customisation is needed for users having different backgrounds?
4. How does the system recover user's right such as financial transactions without delivering results?

### G. Schedule

**Due Date** : To be released

## 4.2 Requirements definition and analysis

From the use case scenario the functional requirements can be presented as in Figure 4.6. The requirement analysis describes in general the working principles of the system functions without the exact details of how this is achieved [51]. An analysis of the use case specification and use case analysis model yields the following requirements specification to be developed.

**Functional Requirements** These define the behaviour of the application. Generally, these requirements play the role of specifying what the application should do without them stating how they are realised.

- The rain information system should have mechanisms for service discovery via SMS. The query information should be handled by a single maximally 160 character SMS. The number of reserved trigger key words should be minimal.
- Provisionally, there should be a capability to handle large volume of messages to and from user. The parsing of the messages should be done as each SMS is delivered.
- System should have automated error handling and correction mechanisms, in scenarios of ambiguous requests, system should automatically interact with user to obtain further information.
- System should permit caching of results to facilitate optimal queries, given a scenario of similar queries having been received.
- System should handle multiple image formats of images for the rainfall analysis.

**Non-functional requirements** These place conditions and constraints upon the application, they specify development requirements and performance issues.

- The prototype shall be based on open source solutions.
- The prototype shall be made in a highly modular environment, whereby several sockets into each module are available thus promoting a highly interoperable and extensible environment.
- The prototype shall provide the output of query and other system processes in an optimal amount of time.

## 4.3 Domain Class Model: Rain Information Service

For this use case there is a domain class model. The primary purpose of the domain class model is to establish the context of the system through the visualisation of domain concepts. It also allows for the capturing of the concepts in the problem domain space and relationship between them [61]. From [44], it is reported to be an information model that is static and has the domain entities. It represents real-world conceptual classes, leaving software components [44]

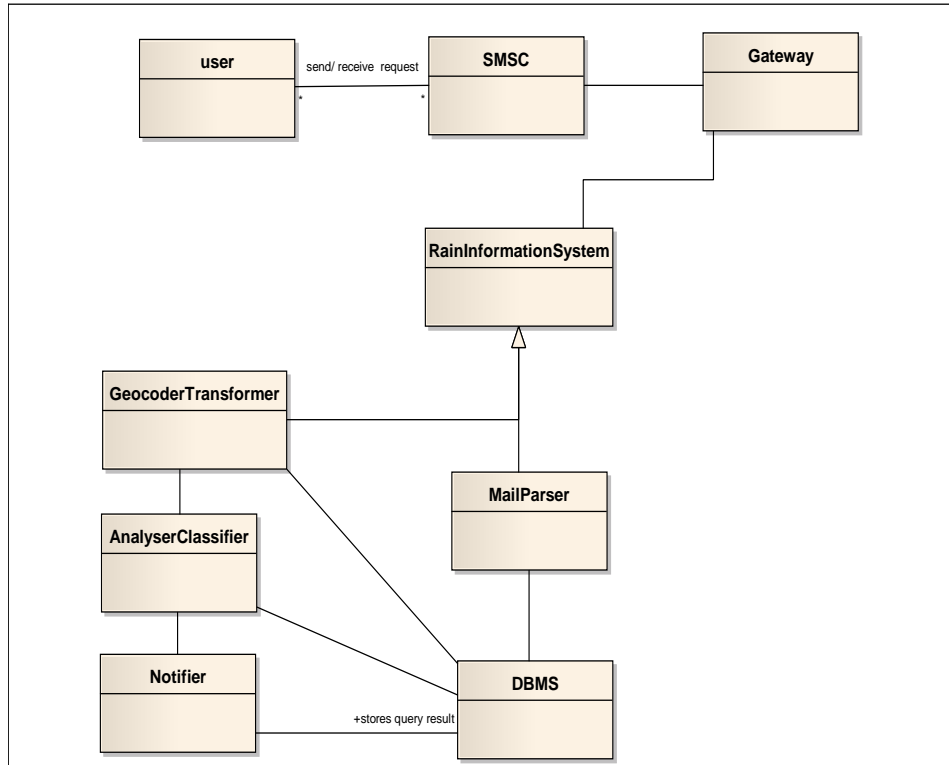


Figure 4.3: High level domain model of the rain information service

thereby providing a visual dictionary of the problem domain vocabulary and abstractions. The problem domain as reported by [18] as a set of problems that belongs to the same category and deals with corresponding information. Thus, in this context the domain class model illustrates a set of UML class diagrams, without defining operations, the problem domain whose elements are conceptual classes and relationships between these classes with attributes and association rules [44].

The following high-level domain class model, Figure 4.3, illustrates the general relationship between the abstract system components. It generally shows the meaningful conceptual classes providing the concepts related to the use case scenario under design [29]. In this model a linguistic analysis of using the noun and noun phrase in textual description of the domain is used to create the model [44].

From the domain class, as in Figure 4.3, the conceptual framework of the problem space is created, thereby allowing us to focus on semantics with a static view being created. This then allows us to convey the time invariant business rules. Hence we can describe the state of the problem domain at any instance of time.

The general requirements analysis also came with a general issues as elaborated in Section 1.1 and Chapter 3. As such, these can be listed as follows for the problem domain space.

From the domain model, there is an additional functions apart stemming

**Table 4.1** Software Architecture View *Extracted from:* [10, 29]

<b>Architectural View</b>	<b>Process</b>	<b>Focus</b>
Conceptual	Architecture diagram	Identification of components and allocation of responsibilities to components
Logical	Updated architecture diagram	Design of component interactions, connection mechanisms and protocols; interface design and specification, and providing contextual information for component users
Execution	Process view (Collaboration diagram)	Assignment of the runtime component instances to processes, threads and address spaces; how they communicate and coordinate; how physical resources are allocated to them

from the modular approach. It is the identification of a data format which allows for interoperability between system components. The general requirements would be as in Section 4.2.

## 4.4 System Architecture Modelling

The architectural view of a system translates to having met the requirements and providing a solution based on those requirements. In this case the requirements are as in Sections 4.2 and 4.3. The architectural view the structural elements of the system together with the external visible properties and relationships, which forms the design model, provides architectural decisions [29, 9, 10]. This can be presented in the form of architectural views that address the given concerns. These views are the conceptual, logical and execution views. Table 4.1 highlights the reported general process and focus of each view.

The UML is a standardised general-purpose modeling language that can be used to model the software architectural views or layers. It is used to create the structural and behavioural view diagrams. These diagrams are as in Table 4.2, and Appendix B.1 elaborates in detail on these diagrams and how they work. To describe the architecture of the two-way system involving an SDI node the Rain Information Service System, henceforth acronymed RISS, will be used. RISS's layered system architectural, structural and behavioural views are detailed in Sections 4.4.1, 4.4.2 and 4.4.3 respectively. These are created using UML diagrams as described by [60, 15, 27, 51, 44, 9].

### 4.4.1 Layered System Architectural View of the RISS

The RISS is built with distributed environments. It allows for a modular approach that is interoperable in terms of data exchange formats and has services which are discovered easily. The design of RISS is adopted from the SOA design

**Table 4.2** Stereotyped UML Diagrams source:[9, 51]

Structural Diagrams	Behavioural Diagrams
-Class -Object -Component -Deployment	-Use Case -Collaboration -Statechart -Activity -Sequence

principles. By definition, SOA <sup>3</sup> is a flexible set of design principles used during the phases of system development and integration. Hence the RISS services should be loosely coupled with the Operating System and other technologies that underlies it [25, 38]. Also from the SOA principle the functions of RISS are split into functions that are distinct units [54].

The SOA principles demand that service provision is again separated into separate units that are accessible over a network. This allows for the whole system to be combined and reused in other application by selection of the appropriate modules only. Between the services there is communication by the transference of data in a shared format that is well-defined. At a partially abstract level the services can be seen from Figure 4.4, which shows the layered system architectural view of RISS.

The bottom architectural layer of RISS provides the storage capability of the system, it consists of the spatial database which contains the geospatial datasets that are needed in information generation and specific spatially related queries. In this layer there exist the non-spatial database which can be part of the same Database Management System (DBMS) as the spatial database but in this architecture is conveniently portrayed as if is separate. The DBMS can communicate directly to the Satellite Imagery service, Geocoding API service and the main RISS engine implementation or can be accessed with service infrastructure such as Web Services and other clients.

The second layer, the Satellite Image Service, is responsible for the extraction of the satellite images for processing. This layer allows for the communication of the system to access remote data stores and provisionally communicate with the geospatial database for storage of these images, if needed. This layer provides the imagery to the main RISS engine, to facilitate image interpretation and analysis.

The third layer, the Geo-coding API Service, provides a means of extracting the geographical coordinates of the specified AOI. This is a service that accesses the information through an HTTP request. For the purpose of inter-operability, the result of this would be provisionally in the form json,<sup>4</sup> kml,<sup>5</sup> xml<sup>6</sup> and csv formats.

The forth layer is the main service layer which facilitates the communication

---

<sup>3</sup>[http://en.wikipedia.org/wiki/Service-oriented\\_architecture](http://en.wikipedia.org/wiki/Service-oriented_architecture)

<sup>4</sup><http://www.json.org/>

<sup>5</sup>[http://en.wikipedia.org/wiki/Keyhole\\_Markup\\_Language/](http://en.wikipedia.org/wiki/Keyhole_Markup_Language/)

<sup>6</sup><http://www.w3.org/XML/>

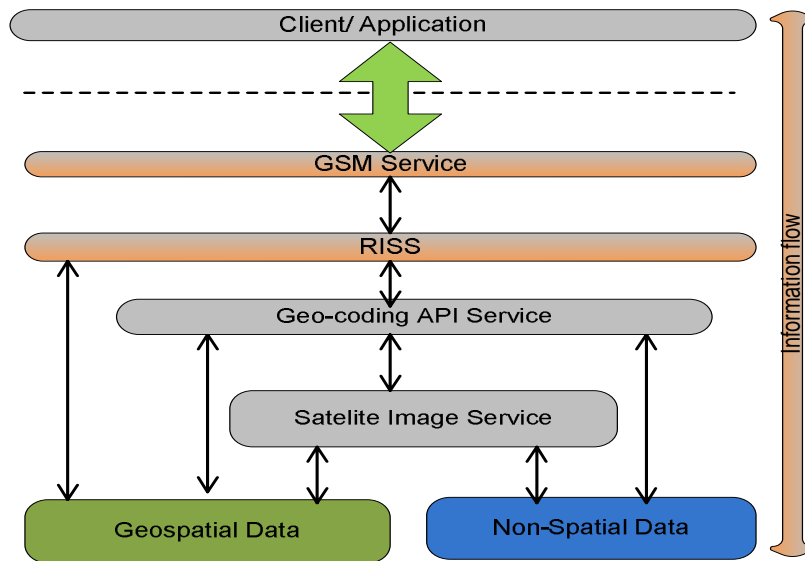


Figure 4.4: Layered Architectural View of RISS

between the various components of the system. It provides an interface between the network and the various system components of the system architecture. It provides the GSM network service, for example, the SMS queries, access to the RISS service layer. This layer communicates and provides an implementation to complete the integration of all the resources. It provides a sound mechanism for flexible, secure service provision and scheduling the resources. The last layer, the Client/ Application layer gives the user access to the system via the GSM network.

#### 4.4.2 Structural Model View of the RISS

In the system architecture for the RISS, the structural model view provides the static, structural dimensions and properties. This is presented using UML's Class, Object, Component, and Deployment Diagrams. They show the design, implementation, and deployment view of system architecture respectively [9]. The following sections provide the model system architecture of RISS at class, component and deployment level.

#### 4.4.3 Behavioural Model View of the RISS

This model view highlights the behaviour of the instances in a system including their methods and collaborations [29]. It describes the valid sequences of snapshots that may occur as a result of both external and internal behavioural effects. It is often referred to as the dynamic, process, concurrent, or collaborative view as it primarily defines how a system migrates from one epoch to another. The UML's Activity Diagram, Statechart Diagram, Sequence Diagram, and Collaboration Diagram are used to model a behavioural view of system ar-

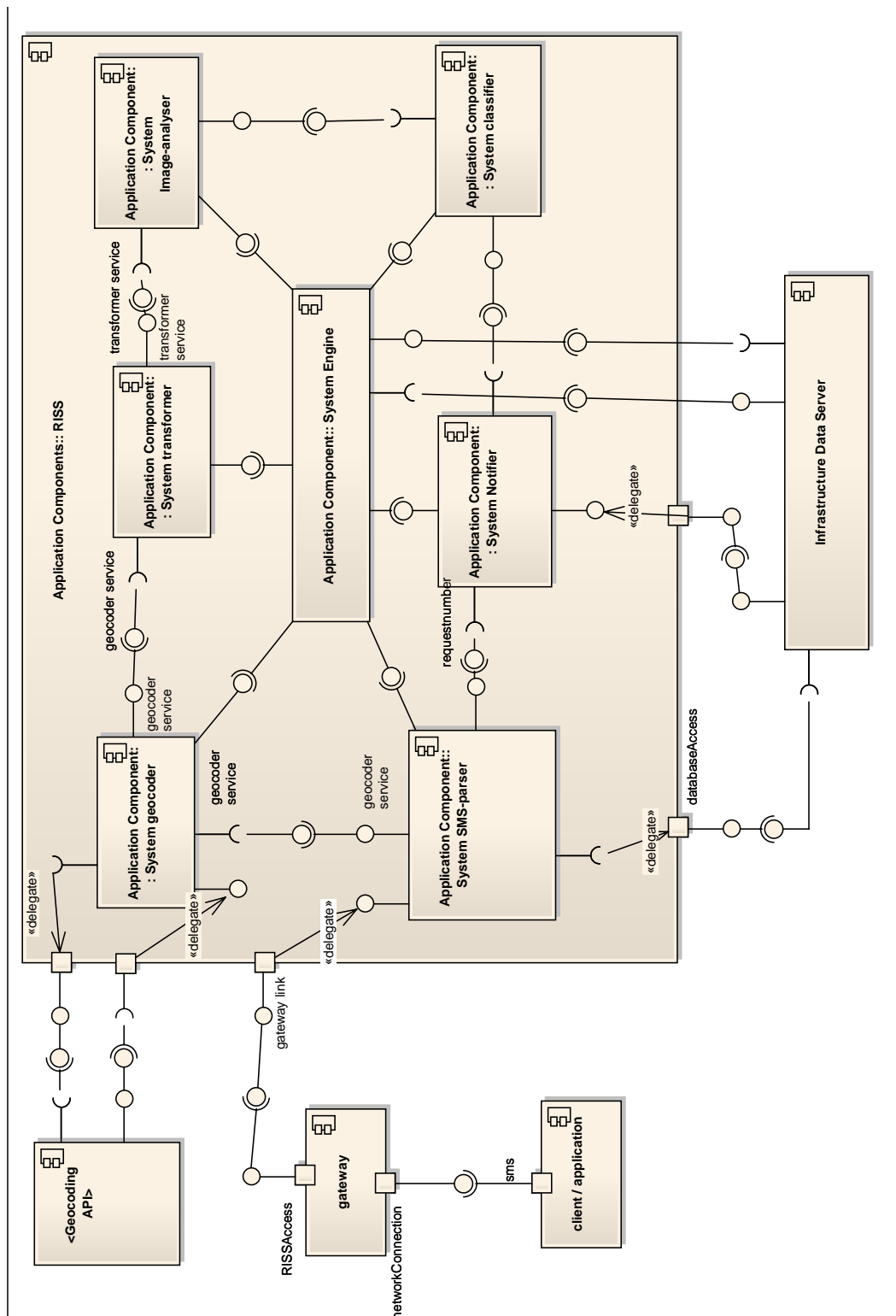


Figure 4.5: Structured Architectural View of RISS

chitecture. The corresponding diagrams capture the dynamic view of system architecture in activity-oriented, event-oriented, time-oriented, and message-oriented forms respectively. For the RISS, the UML's Sequence Diagram and Activity Diagram are used.

**UML Sequence Diagram** It models the usage scenarios and flow of both logic of methods and services. The sequence diagram as in Figure 4.6 captures the dynamic behaviour of system by its representation of the RISS's object interactions through time. The main focus in the Figure 4.6 on page 44 is to identify the behaviour of the RISS and to illustrate the typical scenarios of objects participating in an inter-action and the sequence of messages exchanged within a system architecture.

**UML Activity Diagram** An Activity Diagram is used to model business workflow and model operations, it offers the RISS a representation of the sequence of activities as it successfully captures the dynamic behaviour of system architecture [29, 44]. For the RISS Activity diagram is as in Figure 4.7

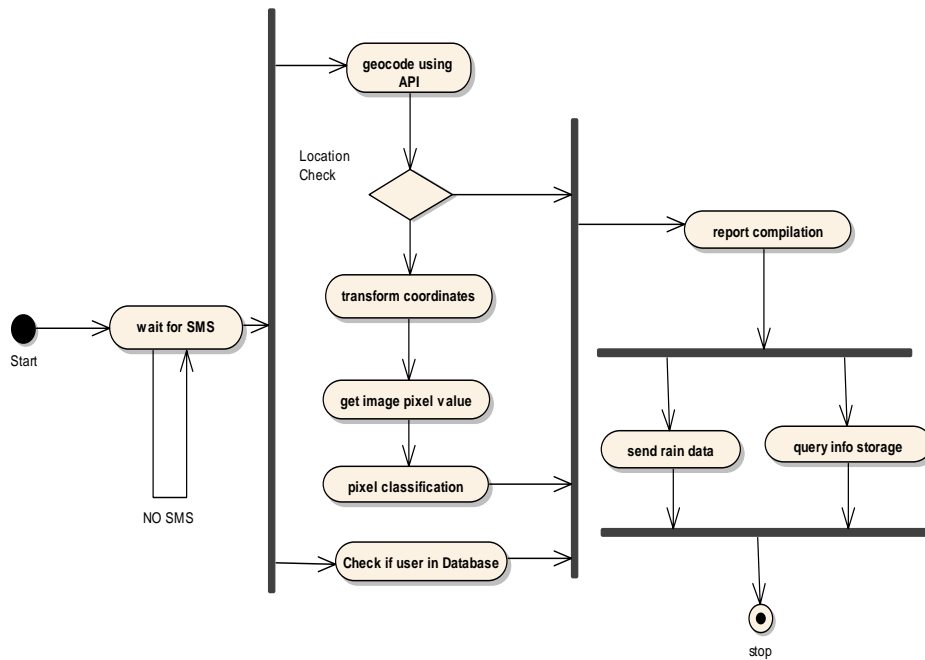


Figure 4.7: Activity Diagram of the Rain Information service

## 4.5 Conclusion

In this chapter we analysed a use case to obtain the functional requirements of an architecture for two way SMS service involving an SDI node. We proposed an architecture in UML. In the next chapter we implement the architecture to demonstrate its applicability and refine the architecture.

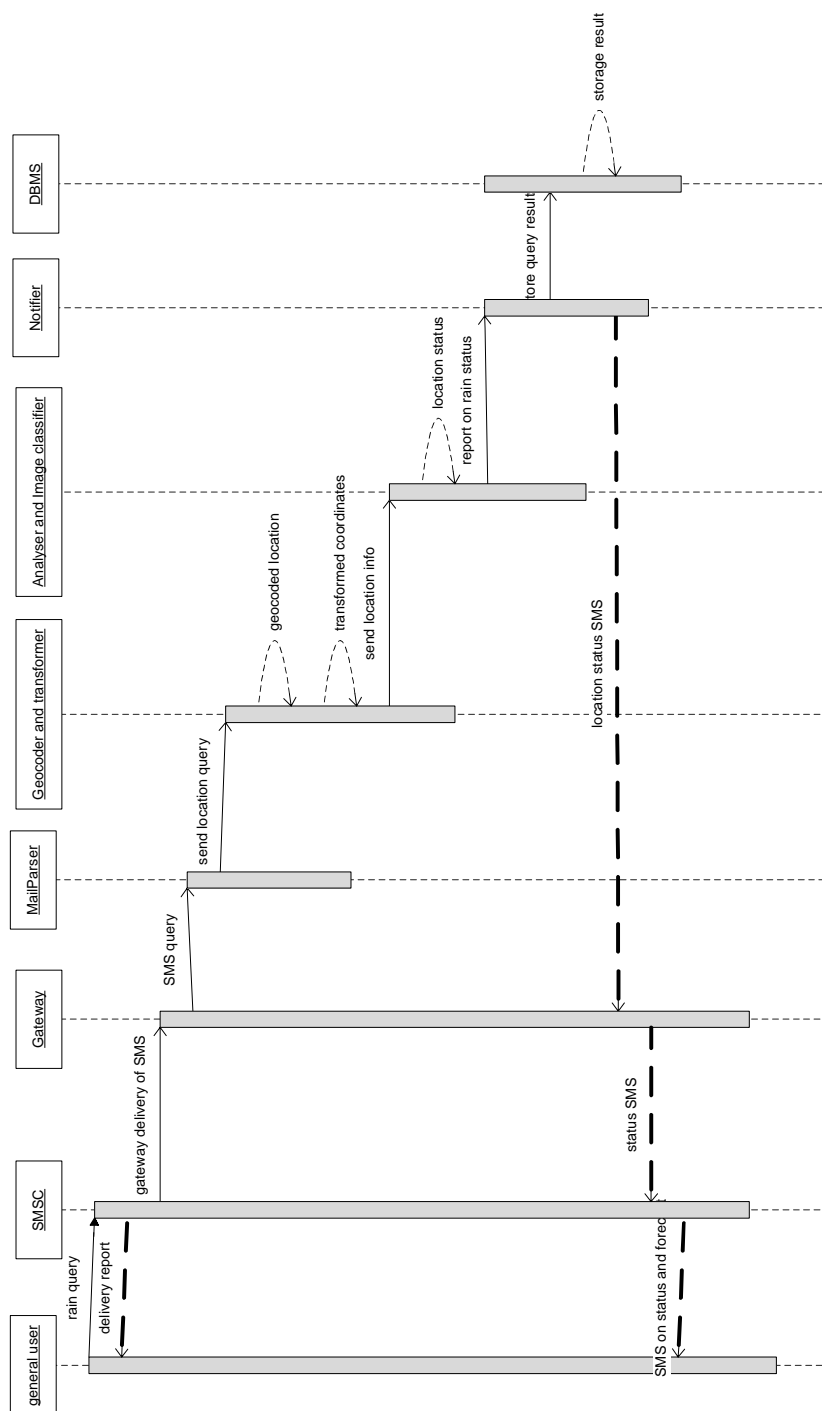


Figure 4.6: Sequence Diagram of the Rain Information Service

## **Chapter 5**

# **Implementation of the Two-way SMS Design Architecture System Prototype**

### **5.1 Introduction**

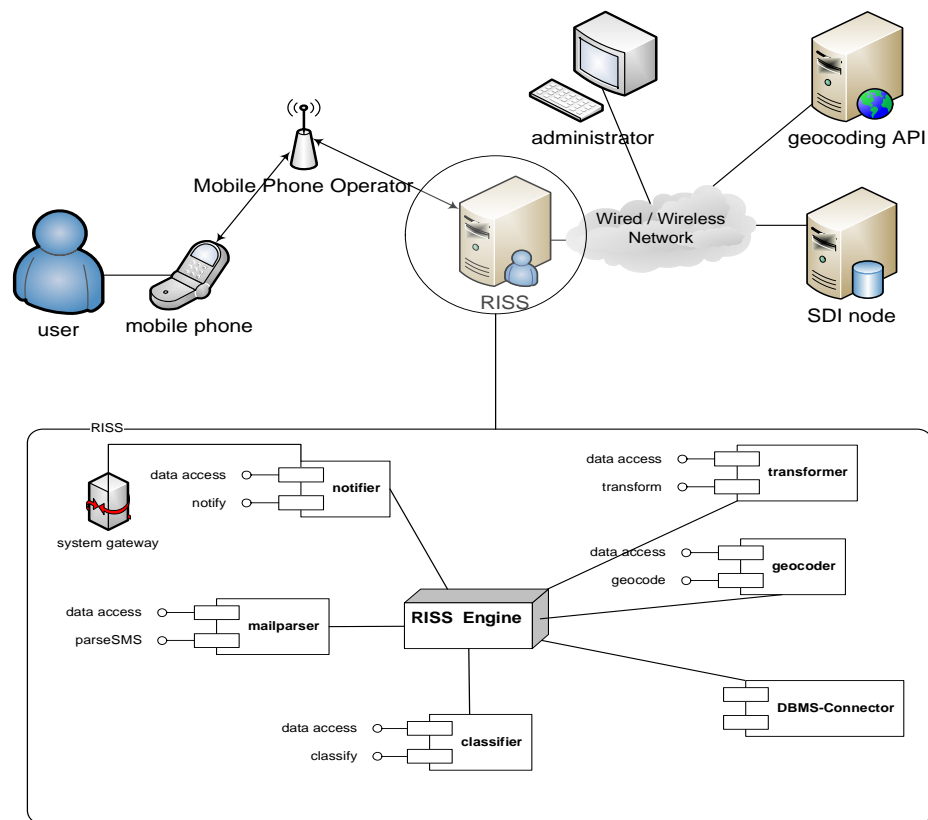
This chapter discusses the issues pertaining to the implementation of the RISS architecture. Based on the architectural views in Chapter 4 the RISS, is implemented. It was developed in accordance with the requirements in Section 4.2. To provide a system that abides with the set requirements and functionality a SOA approach was adopted. This was done by way of discoverable modular programming that facilitated the availability of module services in a potentially distributed environment through service orchestration. The orchestration is the automated coordination, arrangement, and management of the RISS systems [38]. The key element to be implemented is the RISS architecture. The chapter also re-iterates the spatial aspect of the architecture by demonstrating through the implementation steps that some of the message content and/or problem domain is spatial in nature, directly or indirectly.

### **5.2 System Prototype Functionality Development**

For the RISS, several combinations of software, hardware and programming languages were used in the development of the actual prototype. Table 5.1 summarises the software and hardware tools made available for construction of the prototype, while Figure 5.1 illustrates the proposed architecture of the system. It presents the various hardware and software components making up the system, and indicates how they interact.

**Table 5.1** The summary of the RISS software and hardware

Software/ Hardware	Description
PostgreSQL / PostGIS database management system (DBMS)	Open source DBMS with a spatial database extension.
Python Programming Language	General-purpose high-level programming language. Choice is based on the
Java Software Development kit	Provides compilers and debuggers that are necessary or useful for developing applets and applications for the java environment.
Thunderbird	Mail client to handle the mail from the gateway
SMS Gateway	Hardware device or service offering SMS transit, transforming messages to mobile network traffic from other media, or vice versa, allowing transmission or receipt of SMS messages with or without the use of a mobile phone


**Figure 5.1:** Architectural overview of the RISS and the various components that support its functionality

### 5.2.1 Tools and techniques used

With reference to Table 5.1, the tools for the implementation are listed and partially described. The general technique for the implementation is to provide component builds which are loosely coupled to any hardware solution. To support the motivation for choice of implementation platforms, tools and techniques, a greater emphasis was put on to have an implementation based on mainly free open-source solutions.

The RISS installation involved the installation of Python version 2.6 for the scripting and several non-built-in packages. Installation of PostgreSQL, which is spatially enabled by PostGIS, was done and a python package psycopg2 was used to facilitate the database connection.

## 5.3 Model Implementation

The RISS was created in a highly modular technique to support a distributed working environment. As such, each of the identified components in Section 4.2 is implemented on its own as a separate entity to be called by the RISS engine at runtime. The components implemented are described in the next section.

### 5.3.1 Architecture Component Implementation

#### Message Parsing

Message parsing is done by this module using the **mailReader.py** script. This has the function of obtaining the SMS text file from the source repository after the delivery by the messaging gateway. For instance, assume a team of medical doctors in a remote location keeping track of a disease outbreak. This information is implicitly spatial in that the location of the spread has to be known. Thus a message system to parse the SMS and extract the information has to be created and this is the underlying concept behind the creation of the mail parser. Back to the context of RISS, when the SMS is being parsed as a text file a systematic approach is applied. It parses the text file searching for particular key phrases, then if the key phrase meets a preset condition, it takes actions on that condition. The existing conditions are, to place name alone, to register for a service, to cancel a service, to request a place to be monitored and to create a profile. Algorithmically, the message parsing as a very strict parsing procedure given the likelihood of errors from text input by a user. The parsing inherently implies the need for keyword or precondition checking. The precondition checking would be the scan of the words or phrases. Given, that a word or phrase existence a condition is valid and an action is taken. Hence, we built a component to handle potential errors in the message. As stated in Section 3.4, there exist numerous ways to handle these exceptions, and these are explored. The code extract, Listing 5.1 for message parsing, shows the message extraction and keyword search. The SMS character set or alphabet<sup>1</sup> contains characters which can be read by any text file reading mechanism available in most programming

---

<sup>1</sup>[http://www.dreamfabric.com/\ac{SMS}/default\\_alphabet.html](http://www.dreamfabric.com/\ac{SMS}/default_alphabet.html)

**Algorithm 1** *Message Parsing mechanism of the RISS*

```
1: if message exist then
2:   obtain message {message in the mailbox}
3:   read message
4:   call error/spell checker
5:   get query code
6:   action query code {the query option vary depending on requests}
7: else
8:   wait for message
9: end if
```

languages, the parsing mechanism is highly customisable. After having read the message contents, there is a need to correct for detectable and rectifiable errors.

**Listing 5.1:** Message Parsing

```
...
...
def reader():
    # direct the system to the message containing folder
    if os.listdir('C:/Users/shelton/AppData/Roaming/Thunderbird/Profiles/
        b9ocalyt.default/Mail/Local_Folders/')==[]:
        print "No_Requests..... waiting..... Online....."
    ...
    ...
elif str(REQUEST.strip())=="REG":
    from profiler import profile
    REPORT = profile(NUMBER,ADDON,ADDON2)
    REPORT="ZERO"
    print "Profiling_Action_in_progress..... please_wait....."
    REQUEST="ZERO"
elif str(REQUEST.strip())=="REG":
    from profiler import profile
    REPORT = profile(NUMBER,ADDON,ADDON2)
    REPORT="ZERO"
    print "Profiling_Action_in_progress..... please_wait....."
    REQUEST="ZERO"
    ...
    ...
```

**Error handling mechanism**

Given that the query used by the system comes from an SMS, there is a great chance of mistyping on the mobile phone keypad. From the system architecture, the general structure of all queries are known, also all keywords or expressions. This implies that the number of possible keyword expressions for the RISS is fixed. We built a spell checker that works by using probabilistic concept. Assuming a mistyped word *plto*, it could assume to have meant *plot* hence be called a reversal error or *plato* as an omission error, but we do not know which one for suer initially. For example, given a *header word*, that is the first word

in a message string, it is assumed to be a keyword. This being the case, if the word is not exactly as a known keyword it is assumed to be an error and the correct word should exist within a *corpus of keywords*.

Thus we say we are trying to find the correction  $c$  out of all possible corrections, that is from the word corpus of keywords, that maximises the probability of  $c$  given the original word  $w$ . Expressed in probability terms in the Expression 5.1. Where  $P(c|w)$ , implies the probability of a word  $c$  existing as the correction given the word corpus.

$$\max_c P(c|w) \quad (5.1)$$

In probability terms using Bayes' Theorem it follows that,

$$\max_c P(c|w) = \max_c P(c|w)P(c)/P(w) \quad (5.2)$$

As  $P(w) = \text{constant}$  because the word corpus is constant, it implies that Equation 5.3 is sufficient for computation. Simply put in an algorithm as shown in the Algorithm 2.

$$\max_c P(c|w)P(c) \quad (5.3)$$

From the Expression 5.3, it can be directly inferred that  $P(c)$  the probability that a proposed correction  $c$  is a true word in the English Language thus it is called the *language model*.  $P(w|c)$ , the probability that  $w$  would be typed in a text when the user meant  $c$ , the *error model*. The  $\max_c$  which is the control mechanism, is used to obtain the best probability score of the input word [55].

#### **Algorithm 2** Dictionary validation and Spell checker

**Require:** A text word

**Ensure:** Word is in dictionary

- 1: **if** word exist correctly **then**
- 2:   return and pass to engine {message in the mailbox}
- 3: **else**
- 4:   convert to corpus case {the query option vary depending on requests}
- 5:   call spell checker
- 6:   find word highest correct probability
- 7:   **if** probability too low **then**
- 8:     abort return error for exception handling
- 9:   **end if**
- 10:   replace word
- 11: **end if**

The spelling correction mechanism subsequently works by reading from the corpus, extracting the words, then converting them to the dictionary case. We then train the model, Listing 5.2, by counting the number of occurrences of the word in the corpus [55]. Next, the **edit distance** of the words is computed, that is, the number of edits it would take to turn one into the other.

**Listing 5.2: Error handling mechanism (adopted from: [55])**

```
def words(text): return re.findall('[a-z]+', text.lower())
def train(features):
    model = collections.defaultdict(lambda: 1)
    for f in features:
        model[f] += 1
    return model
NWORDS = train(words(file('big.txt').read()))
```

Considering the fact that SMS messaging errors are usually related to typographical errors, we assumed that one edit distance suffices to correct the word. The result of this was the need to list all possible options of words that meet this criterion and this was done using the *edits1* function of Listing 5.3.

**Listing 5.3: Dictionary (adopted from: [55])**

```
def edits1(word):
    splits = [(word[:i], word[i:]) for i in range(len(word) + 1)]
    deletes = [a + b[1:] for a, b in splits if b]
    transposes = [a + b[1] + b[0] + b[2:] for a, b in splits if len(b)>1]
    replaces = [a + c + b[1:] for a, b in splits for c in alphabet if b]
    inserts = [a + c + b for a, b in splits for c in alphabet]
    return set(deletes + transposes + replaces + inserts)
```

Concluding, we can obtain the highest probability word that combines with the lowest edit distance to correct words.

### SMS Slang

In implementing the spell checker there was the realisation of a trend in SMS language to SMS Slang [33, 30]. We developed an interpreter to translate the slang into normal modern English. This works by using a dictionary which does a key-value pair matching after the phrase has been passed through the spell checker. This is displayed as in Listing E.1

### Profile building component

To interact with the user, there is a need to know who the user is and what is his subject of interest. For the RISS system, we developed a component that interacts with the user through a series of sequential text messages to obtain information about the user. This is implemented by having a series of leading questions sent to the user obtain specific responses. The correct and *validated* responses are stored in the database.

Two-way information flow occurs during this process and spatial and non spatial data is passed between the system and the user. The RISS facilitates the flow of information to and from a datastore, this data store can be the SDI node. By way of example to describe the profiler, assume the existence of a Mr Shelton Butau who resides on Hengelosestraat 99 in Enschede. To create a short profile of himself he prompts the system to register him by way of SMS with the keyword *REG*. The system welcomes the registration and asks for his name giving an example of how this is done. He sends his name and then the system asks him where he resides by another SMS.

The information is stored and his residence is geo-coded and stored in the spatial database and his additional information is then stored in the non-spatial database. During this simple interaction, there is data validation and verification and reports generated. This is done on the basis of whether other people have registered using his phone number and if there are existing registered people on his mentioned place of residence. From the given information, spatial data and non-spatial data is passed through indirect means and stored, for example, as point or polygon of his place of residence, which can be translated to the AOI depending on scenario. Algorithmically, this is shown as Algorithm 3.

**Algorithm 3** Profiling the RISS user

**Require:** keyword

**Ensure:** spell checker validation {validate the phrase to ensure that it is slang and not a typo}

```
1: if word is not a keyword then
2:   return error and quit with report to user {this informs the user of the
     error}
3: else
4:   if profile not complete then
5:     while profile not complete do
6:       check keyword word query {the query option varies depending on
         requests}
7:       process query {if query is to store name, location et cetera, it is then
         stored}
8:     end while
9:   else
10:    deny further information
11:   end if
12: end if
```

An extract of the script to do this profiling is shown in Listing E.2

**Location component**

RISS as a system can be said to have a potentially vast array of positions of the subject of interest. Naturally, spatial location can be reported in several ways via SMS. For instance, by way of coordinates, name of place, binary form for the location geometry to a mention few. But the question is, is it feasible to do so via text? To answer this, we took into consideration the context for the implementation, the user primarily and their potential ability to do so. Also, the area of interest might not be in the exact location as the mobile phone. The same question can arise in another instance like truck driver. Assuming they have a system which can query road status, they need to ask for information at a particular point or at the point they shall be at a given time. To have them use a web application can be impractical if we consider that they are in Uganda and they might not be in possession of a 3G phone, but they rightfully need the information. As alluded earlier the way out here is the SMS and the RISS

explores an option of providing geo-location via SMS of a phenomenon. The RISS provides information on rain, it is likely that the user will be querying for information on a place they are not in at that particular moment. It is impractical to request geographic coordinates of the location. To solve this and in line with the system requirements documented in Chapter 4, a component that uses the *geoloc.py* script was created.

What it does is geocode a location, that is the AOI, provided in the form of a place name. The geocoding function uses an external API from Google Maps API Services.<sup>2</sup> The API returns the geocoded result in the form of a JSON object. A script extract to illustrate the geocoding process is in Listing 5.4

Listing 5.4: Location component

```
# Function to parse a name locating string to google then output
latitude and longitude location of the place
import urllib
def geo(placename):

    #replacing the whitespaces in the placename with a +
    place=placename.replace(' ','+')

    # key from google to access the API also tying the geocoding to the
    Netherlands
    key='KEYVALUE'
    ApiURL= 'http://maps.google.com/maps/geo?q='+ place +'netherlands'+
        &output=csv&oe=utf8&sensor=false&key='+key
    sock = urllib.urlopen(ApiURL) # API call and output
    location=sock.read()
    sock.close()

    #return the location of the \ac{AOI} in geographical coordinates
    return location
```

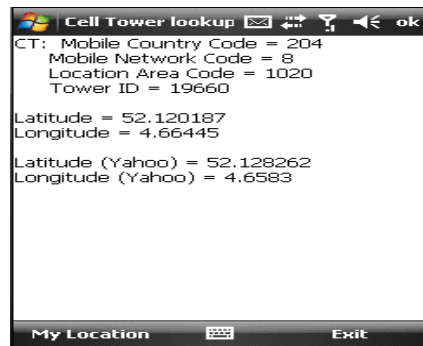
In view of the potential of requirements to locate the mobile phone is directly related to the subject of interest. A Java ME MIDlet application (MIDP 2.0) that utilises Cell-id Look-up API from Ericsson labs.<sup>3</sup> We used Eclipse with the MTJ (Mobile Tools For Java) addon. It allows for the mobile phone to extract its cell-ID, unique identifier of the radio base station that the user would be using for communication. The cell-ID is then queried from the Ericsson labs API, which has the cell spatial location. The Java code for that implementation was adopted from the Ericsson labs.<sup>4</sup> Direct implementation of this in the RISS was not done as the context of the research was for low technology mobile phones. Java enablement could not be assessed as a capability of most mobile devices. The view is that such a MIDlet, if incorporated would allow the cell-id information to be sent to the server system. The server system would in-turn query the cell-ID look-up API via local internet protocols. An example of the client side implementation of the MIDlet is as in Figure 5.2.

---

<sup>2</sup>[www.code.google.com](http://www.code.google.com)

<sup>3</sup>[labs.ericsson.com](http://labs.ericsson.com)

<sup>4</sup>[labs.ericsson.com](http://labs.ericsson.com)

Figure 5.2: Java Client-end look-up MIDlet *source:msdn.microsoft.com*

### Short Message Sending and Receiving component

The configuration of the gateway requires that SMSs are received by the RISS in the form of emails. The received emails are processed as plain text. The mailing module is split into two, mail cleaning and mail sending. The main cleaning part is to delete the inbox of the text file after it has been received and processed or compiled into a text message and sent. This is done by the ***SMS2Emailmailer.py*** module script to send the email shown by the Listing E.3. An email from the RISS engine is emailed to the gateway in the format *number@SMS.gateway-address*. The gateway subsequently sends message as an SMS using its own internal mechanism.

## 5.4 Interface/ Engine implementation


The RISS engine was created to run as a service such that at any moment it would be running on the RISS *server/ machine*. To allow for this, the RISS engine, responsible for managing and coordinating the various components was created. As Listing E.4 shows the interface responsible for coordinating the actions of the various components. This is done by the script ***RainService.py*** which has a variable refresh rate of checking for messages. Most system resources are released in the event that no process is occurring, that is when idle waiting for new SMSs. The extract in Listing E.5 shows the idle and rechecking engine that uses the scheduling package in the scripting environment.

The creation of the windows service was done in the windows environment using the *sc.exe*, a built in software in the operating system. The syntax for this creation is

```
sc create YourServiceName binPath= "c:\Program Files\
directory\appname.exe"
```

The screenshots in Figure E.1 shows the RISS when idling and waiting for a new message. The overall premise of the service being run in the system background. Figure 5.3 shows the RISS executing in the shell. It would be processing an SMS requesting the rain status for a location. The user in this

case would already be registered. It extracts additional information of the user according to his registration and stores the query and the result of the query in the database.



```
Processing next....please wait....
paexcept paye
Processing request from phone number 0645389975 about "hengelosestraat 99"
"
http://maps.google.com/maps/geo?q="+hengelosestraat+99"
+netherlands&output=csv&oe=utf8&sensor=false&key=ABQIAAAATRE6eC2ppcFR-cih1HCZSRS
EAfeKgygK8e4wC_GrNvggAwfuUxSEzu_8iSeHKykg3xuXbzeEFL9xsg
The geographic location of request is 200,8,52.2233122,6.8906508
The pixel value is 216
pvalue1 is 217 pvalue2 is 219
The general condition is most likely 'Now: No rain 30min: No rain 1Hr: No rain'
select p.username from userdata as p where p.phonenumber= '645389975'
User requesting information is sheldon butau'
Message sent successfully to 0645389975
Something went bad during the connection:
could not connect to server: Connection refused (0x0000274D/10061)
Is the server running on host "localhost" and accepting
TCP/IP connections on port 5432?
Processing next....please wait....
```

Figure 5.3: RISS processing an SMS with a general request

## 5.5 Review of architecture, The General Framework

The architecture of the two-way SMS service involving an SDI node has the purpose of documenting the ‘core’ set of operations and components. After the implementation, a reflection on the architecture proposed was made and we provide an open, vendor-neutral architecture for the service. The two-way SMS-SDI architecture has its design principles firmly ground those of a sustainable SDI. The reason being the SDI architecture is the mature backbone to the whole system. de By (2007) reports that a “spatial data infrastructure encompasses a set of interconnected, technology-neutral, user-tailored services that support the activities of the geo-information community, which in turn underpins the activities of society in general” [20]. From this, we infer that the SDI nodes are built on the underlying principles of the SDI which are in line with SOA principles and the services layered as in the SDI.

The SDI stack, Figure 5.4 shows how the SDI node is layered. Design principles of a SDI node dictate that the following.

- “SDI nodes do not provide access to an organisation’s information systems but to services that exploit the information held in those systems.

- A service provides functionality via a well-defined interface that is tailored for consumption and has an explicit quality of service.
- A service interface encompasses a number of operations that are invoked through the exchange of messages.
- A message represents the exchange of data (content) packaged as XML documents.
- A service should support invocation by other services, or by systems or applications that are not themselves services.
- A service that operates on raw data exhibits an interface that allows the consumer to provide such data from an external source.
- A service provides its functionality totally unaware of the context in which is being used.” [20]

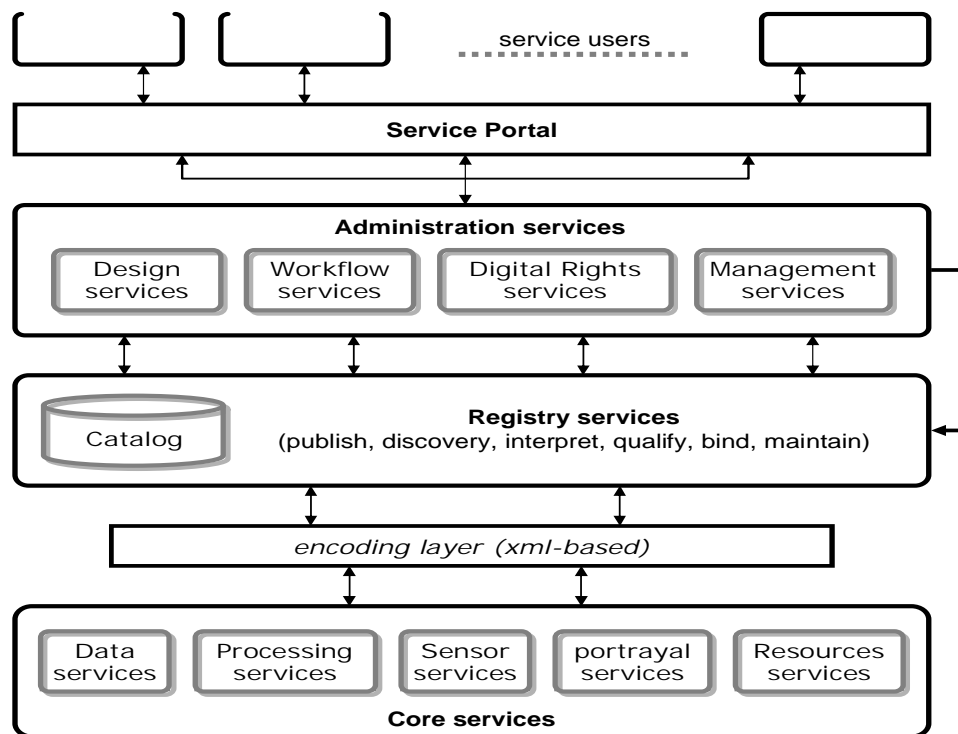


Figure 5.4: The service stack of an SDI. (source: [20])

Figure 5.4 highlights that the communication is in the form of services. From the SDI node building principle of a SDI node providing access to it in the form of services only, it implies that in the integration with a SMS system the communication protocol is in the form of services. From the SDI architecture, Figure 5.4, the access would be in the service users layer. The communication being in the form of Extensible Markup Language (XML) messages. This implies that it has to adhere to the following.

- Receive information in the form of XML documents.
- Query information in the form of an existing service.

These principles from the architecture of SDI node propagate to the architecture of SMS-SDI architecture. We present the architecture of the two-way SMS services involving an SDI node in layered view as services linked to the SDI service stack, Figure 5.5.

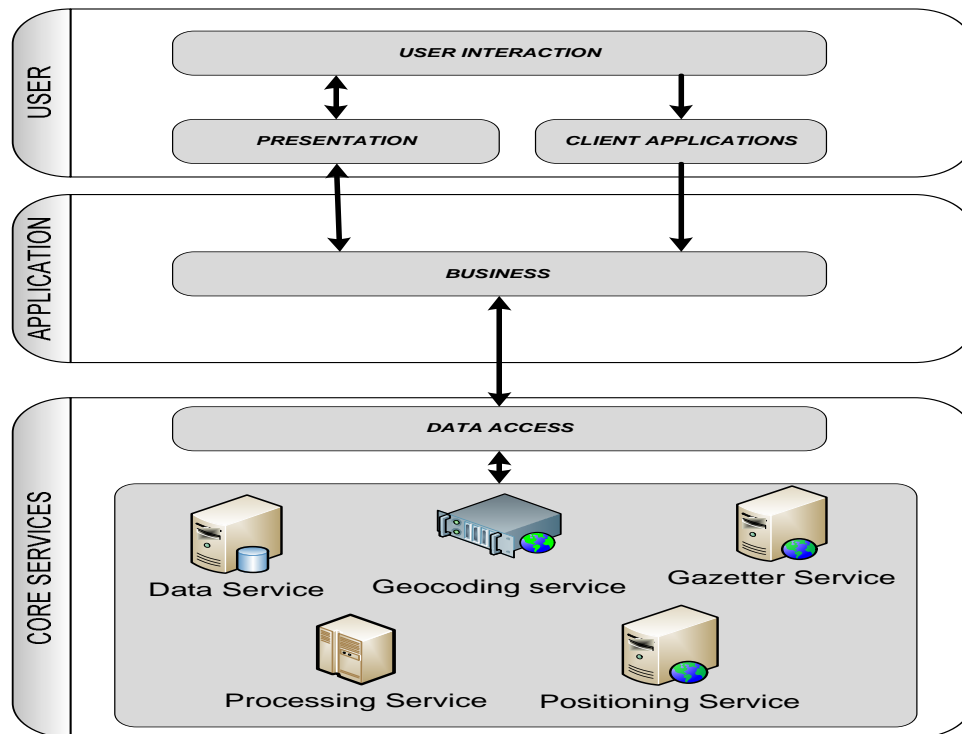


Figure 5.5: SMS-SDI Service Stack

**Core Services** represent the key services that the architecture is based on. The services can have native interfaces on them for information dissemination. These services on their own can be used for other purposes in other contexts. In this layer they exist the Data Service, this service is the one responsible for the communication with the SDI node. From Figure 5.4, the communication is through the top layer, the service users layer. This will provide the access to the node's Data services in the core services layer of the SDI node. The core services of the SMS-SDI service stack also includes the Geocoding, Gazetteer, Processing and Positioning services. The geocoding services is responsible for the converting addresses or placenames into geographic coordinates. This service can be an internally built system for the geocoding or this service can provide access to external API that provide a geocoding service. The Gazetteer service also works in the same manner as the geocoding service but allows access to search and retrieve elements of a georeferenced vocabulary of well-known place-names. This

service can be intertwined with the geocoding service to enhance service delivery.

Within the Core services layer there also exists the *Processing* service, this service provides access to the application or business layer to specific functions which might entail the chaining of other services. This depends of the general function of the SMS-SDI system. The positioning service has the responsibility of determining the location of the subject of interest. This can be a result of calling the other services within the domain, or the extraction of a location from other sources, for example, the Cell-Id databases provided by other systems as API calls.

The *data access*, follows from the SDI architecture where it is stated in OGC context, represents standard interfaces to the data sources and services. This layer provides the validation, authentication, registration, lookup of services and data types.

**Business - Application Layer** is where the integration of multiple services and or information sources takes place or a program works upon information provided by the data access service. Due to the ability of a distributed environment, these services can take place on one or more servers which can communicate by way of services. From the technical ability of the SDIs data access interfaces are used. These exposes the interface components which subsequently allows service chaining.

**The User Layer**, this tier denotes the client. In this context there is the user interaction which represents the user interacting with the system via SMS. The presentation would be the wrapping of the data to and from the user in the form of a SMS. Most SIM cards or mobile devices are Java enabled, this then provides the capability to build applications resident on the SIM/device memory. Hence communication or service call can be actuated at the device directly to the business layer.

The data communication protocol for message exchanges between the layers is by way of XML or service calls. The exception to this is in the top layer where the channel of communication is by SMS. The reason for this to create a highly interoperable environment.

## 5.6 Conclusion

In conclusion, the RISS implementation was created from the proposed architecture successfully. In its testing a reasonably sized user group was created and registered but unfortunately those tests were not sufficiently done due to gateway mail forwarding issues. The demonstrator prototype was tested with a much smaller user group and it efficiently provided good results in a reasonable response time. From the building of the demonstrator a review was made of the architecture and a general framework was described.



## Chapter 6

# Discussion, Conclusions and Recommendations

### 6.1 Introduction

Having come from the fundamental scenario of having a spatial information system and wishing to communicate with a group of users, for instance, health workers in a developing country with the mobile phone as only reliable phone. They themselves have mobile devices mainly as the reliable form of communication. They also have standard services like SMS and possibly no access to 3G services. The question was how can they use such devices to communicate spatially related information coming from a spatial node to and from and also amongst each other? This led to a series of research questions which this research project aimed to address. In this thesis, we designed an architecture for the RISS and implemented it using mainly open source solutions. The SOA approach was adopted to implement this system prototype, which allowed us to orchestrate a system to handle distributed environments. During the thesis project, several options were explored and used. As such based on questions like why people join such SMS communities, what are they getting out of it? What is it that they expect to get out of it? Fundamentally for this research the question would be how do they do it? What is it that they need to communicate using SMS when we involve an SDI node? This somehow determines what the SMS message may contain and how we get out of it. This chapter summarises the results, discusses and provides answers to such questions in fulfilling the research objectives and answering the research questions listed in Section 1.2.

### 6.2 Discussion

The background scenario, was to come up with SMS-SDI solutions that could be used in southern contexts. As argued in Chapter 1, in the southern context there is an ubiquitous access to mainly low-tech mobile devices. These mobile devices can be used to allow for two-way communication between the user and the SDI node. The architecture proposed in this research project allows for this communication between an SDI node and a user through the mobile device. As

discussed in Chapter 1, the working environment is mainly a low tech environment and use of the modern day 3G mobile phones is not a solution. Therefore, the solution would be in the SMS, which is ubiquitous, to communicate the geo-informed messages between the user and the SDI node.

In this section we outline and discuss the answers to the research questions that provide the solution to the SDI-SMS integration that are put forward in Section 1.2. This discussion summarises the achievements of our research project in accordance with the addressed research objectives, the limitations and problems encountered. The research questions are presented below together with the solutions.

*First, what are the architectural components of an SMS system for handling spatial information on an SDI node?*

The research project has a background messaging scenario of a geo-spatial data source being accessed by a community of users and interacting in a two-way manner. This geo-spatial information being conveyed in an automated real-time server-side processing, which an SDI node can have. In the design, the following components were identified.

**Message Receiving Component** There exist numerous ways to receive a text message using a computer. From the analysis of existing products as in Appendix C, the solutions for receiving text messages were found which are 1) GSM modem 2) network supported SMS gateway and 3) SIM hosting or shared phone number. In light of the southern contexts, we chose the GSM modem as the first choice solution. This is because as alluded earlier, it is the most feasible, easily implementable and cheapest solution. However, this does not mean that the other two choices are not equally good. The basis for choosing the GSM modem was in consideration of its feasibility. On the feasibility aspect, we realised that GSM modem solution demands less technical expertise to accomplish. This tallies well with the technical human resources capacity within the southern contexts which is in short supply [11]. This does not rule out the fact that in the southern contexts there also exist qualified, capable and competent technical personnel. On the other hand, this would imply that the resources in terms of finances and hardware are consumed less, thereby making this option a good possibility for receiving text messages. Having elaborated the logical side to the GSM modem, we also have to highlight the constraints posed by the chosen solution. GSM modems allow a limited number of between 6 to 10 messages that can be sent per minute. In this light, the number of queries that can be handled for a potentially large community of users is restricted. This might be interpreted as the system being less effective in service delivery. Less effectiveness is measured by the number of queries successfully executed at a give time interval. For a system to be effective, it should be able to produce the intended service. However, the merits of GSM modems outweigh the demerits considering the context and intended service to be provided.

The network supported SMS gateway can handle a large number of messages per minute depending on the network service provider. One might wonder why then did this solution is not chosen as the first option. The reason why the network supported SMS gateway being the second choice is that it is a bit expensive than GSM modems. Even though it executes more queries than GSM modems, it is not passionate with the financial constraints typical in southern contexts. This is because it requires a contractual obligation with the network service provider. That in itself poses bottleneck to the successful implementation of the system. Therefore, we might end up without a system in place to handle the two way SMS service involving an SDI node. The same can be said for the SIM hosting solution.

All the solutions discussed here are considered possible to receive/send text messages via an ordinary SMS. The message formats it supports are described in the section on message formats below. The message formats that can be handled by all the solutions do not pose notable constraints in the gateway choice.

**Parsing Mechanism** Having received the message using various ways as stated above, the next thing is to *parse* it in an automated environment. As stated in Section 2.3, the message can be in various formats. For this research, we use the SMS-to-email option. In this option, the messages is received as an SMS by the gateway and then sent to the system as an email. The reason for the SMS-to-email choice, is the availability and ease to integrate with prior existing solutions such as mozilla thunderbird and microsoft's outlook. The two text messaging modes available are the text mode and the PDU mode. It is only logical that the implementable option be the *text mode*. The PDU mode is not a feasible choice as the user community cannot be expected to text in hexadecimal octets. Messaging in hexa-decimal would mean the user, for example a general farmer should be conversant in converting alpha numeric to hexa-decimal. The hexa-decimal octets are only feasible when an external phone applications is developed to handle the encoding. For example, having a farmer called de By in Uganda to request information on rain of his farm in Kampala by saying

7261696e206b616d70616c61207567616e6461206465204279,

which translates to *rain kampala uganda de By*. Thus it can be seen for obvious reasons that the text mode is the best way of sending a text message.

Consequently, after the message is received as an email, it is subsequently forwarded to the system mailbox. In the RISS, the system was coupled with an external mail client which was used to handle mail filtering to a specific folder. Any mail client can be used to do the filtering. For implementation, we chose the Mozilla messaging Thunderbird<sup>1</sup>, as it is free opensource.

---

<sup>1</sup>[www.mozillamessaging.com/thunderbird/](http://www.mozillamessaging.com/thunderbird/)

The parsing is done by way of keyword detection within the message string, and at this stage any scripting language is able to handle this, trigger an event to occur when the keyword has been identified. For instance this can happen in our message above. With the word ‘rain’ in the string, the next thing is for the parser to check for the next component which is the location aspect, as discussed under the heading Location of subject of interest.

**Error Detection Component** In order to handle an error, first we have to detect it. To detect a message error, the simple rule of thumb obeyed initially is, *anything that comes which we do not expect is potentially an error*. During the process, if a keyword is detected all the possible variations of message syntax are known. For instance, if a health worker wants information in status of disease outbreaks by location s/he can send an SMS as *status [SPACE] cholera [SPACE] kampala*, which is a *request [SPACE] disease [SPACE] location*. So, when the system parses the message it expects such a message syntax. If not as that, then the string goes through a correction mechanism. In the event, that the correction component fails to handle the error and translates the message to a machine-understandable syntax, and preset exception routines are executed. This involves a calling routine to report back the error using a specific protocol/ interface that is pre-configured. A potential loophole can be the inability to cater for different languages. For example, different spellings pertaining to the same object. An instance of this can be place names like *Zambesi* or *Zambezi*. Both the spellings are correct but can be regarded as errors if the system cannot handle this. Another limitation is the inability of the system to be robust enough to handle multiple languages. For example, in Zimbabwe, three languages are spoken and any system created should be able to handle these languages not to mention the dialects.

**Message Correction Component** Having dealt with message parsing and error detection, we try to correct the errors in the message. The types of errors expected are described in Section 3.4. To correct, we propose techniques which exist in NLP. Considering that message parsing is a light weight variation relative to the actual domain of NLP, we propose inheriting from the classical approach of the Levenshtein [47] and a probabilistic approach described in Section 3.4 and Section 5.3, respectively. This approach has the advantage of not being computationally heavy on the server. Given that, there exist a finite corpus of keywords, the keyword correction is easily attainable which leaves the location verification. To check whether a point is geographically within bounds, the geocoding can be employed to provide the coordinates that are checked if they are within bounds. For instance, a truck driver in Kenya requests road data about Harare, Zimbabwe from a system dedicated to Kenya. This system can handle the error as the process can be given pre-known geospatial extents and anything outside this would be handled by an exception using specific protocols/ interface that are pre-configured which can involve informing the user.

Geocoding is a time and resource-intensive task and it is recommended that all geocoded location results be cached. This allows for subsequent similar location results to be extracted from the database or cache. This increases system

performance as less resources would be dedicated to geocoding. During this stage, a constraint on location outside a preset limit can handle out of bounds requests.

**Query Formulation Component** From the RISS implementation, we saw that if there is a specific query, a *canned* procedure can be created and formulate the query. When the text is being parsed, the relevant keywords are extracted and used to come up with relevant query information. In the case of the implementation, we had a database adapter which provided access to the PostgreSQL- PostGIS database. SQL queries were created by extracting a relevant section of the SMS and executed. The result returned is processed and reported back to the user. This can be done by almost all programming languages or scripting languages, thereby is not seen as a major technical problem but rather mainly conceptual as there is need a to abstractly create *parameterized* queries, whose parameters are requested within the script and executed as general queries to the Spatial DBMS.

**Message Sending Component** This component of a messaging system works in the converse manner with a message receiving component. In this instance, we have to take the query result. If need be, translate it to plain text, compile and email it to the gateway. This is then subsequently forward as an SMS message. In any situation, it was seen by the RISS implementation that this is a scripting issue. Given a situation in which all the components have the proper sockets and produce correct information, the message sending component only has the role of extracting the information from them. For instance, after a query of rain status at location Enschede. The message sending component extracts the query number from the parser, the report from the classification mechanism then compiles the message and sends an email. To send the SMS, the possible options are under receiving as described before in this section.

**Mechanism to position subject of interest** Location of the subject of interest is of paramount concern in any geospatial application or innovation. In this research project, we examined several options, as in Section 3.2. Apart from the options given there is another option in Java Specification Requests (JSRs). Given that there is an increasing number of low cost mobile phones that are Java enabled. Given that it is report that most mobiles phones are Java-enabled [34], the potential of positioning a mobile phone as using a JSR increases phenomenally. For instance the JSR 179 allows for the retrieval of physical location of mobile devices. This can occur in two ways:

- i. On a mobile device armed with GPS functionality, the physical location is directly extracted using the JSR 179. Given the scenario we focused on in this research project, that is the southern context, this type of technology remains beyond the reach of many in terms of financial cost, so developing applications in this arena would not be very appropriate.

- ii. Thus in our case, which is the low-tech mobile device that is not equipped with GPS functionality, the JSR 179 can still be used to extract the spatial location through signal triangulation using the signals from signal base stations/ towers. The JSR 179 comes with its own challenges in our research context. First of all the mobile device has to be Java enabled in order to be able to use the JSR 179. This is problematic because though most phone are said to be Java enabled there still exist a great number of mobile phones which are not, especially in the southern context. Another limitation of the JSR is that it assumes the user device to be static, this then introduces accuracy problems as the device might be in constant mobile. Depending on the user community, these limitation can be outweighed by the potential benefit of being able to obtain the location attribute of a phenomena.

After the JSR 179 has been used to obtain the location, we can also harness the potential use of the JSR120, which enables programs to send SMSs from mobile devices. Thus, several positioning alternatives exist and they primarily depend of the phone capability. The solution would not be ubiquitous. Analysing the location problem and the technology available, it is safe to conclude that the easiest and most efficient way of obtaining the location is via the user input. By this, we mean the user reports the location within the SMS. For instance, a truck driver might text the location he is in, say *road [SPACE] kampala*, and then send it. This way, provides the location on his own. From such a scenario, it can be foreseen that there are potential problems when the user gives the information and several errors introduced, can hopefully be successfully dealt with by the error detection component.

### **Bulk Message handling**

Bulk SMS text services allow for sending a *broadcast message*. It provides for the sending of a single SMS to many several recipients at the same time. In principle, the message creation can be done once and sent to the users of various numbers. A potential usage of this is in the two-way SMS communication involving an SDI node whereby a user with a relevant access level can invoke a broadcast SMS via a single request. For instance, if there is a potential spread of some disease, which, if noticed or reported late, can have catastrophic effect for instance cholera, an alert message can be sent. The message can be *cholera-alert [SPACE] 10km Kampala centre*. From the message the system selects users within the reported geographic extent and sends a bulk message as a pre-configured cholera alert message would be sent to those numbers.

By such a simple example, we observe that such a minor system can potentially have far-reaching benefits. In this research project, the Bulk Messaging as discussed is only at conceptual level as for most bulk messaging services there is a requirement of a contract with a network service provider of some specialised gateway. Thus we were not able to experiment along those lines.

*Secondly, what software and hardware needed for low-cost SMS messaging of spatial information.*

This question draws its answer from the general components and what is required to build the architecture. First, there is the gateway to send and receive the messages. Then from the RISS implementation it was seen that if the SDI node is internal, there is need for the server, and this could be a virtual server or an actual server. Another aspect in terms of the software is the system itself which can be built from proprietary software or from open-source solutions. For this research project, we chose open-source solutions provided by Python as the scripting language and the software package PostgreSQL as the Spatial DBMS which provided the SDI node.

*Thirdly, what is the appropriate format for sending spatial information through SMS?*

In essence, to answer this question we needed to understand what current SMS services support. It was seen that there exist two formats for sending SMSs. These are:

1. Text format
  - (a) **7-bit**, using maximally 160 characters, from the 7-bit default alphabet.
  - (b) **8-bit**, using maximally 140 characters, but these cannot be viewed as text but used for data exchange. The only place where we can use this form in the SMS- SDI architecture would be in the event that there is a need for machine to machine communication via SMS.
  - (c) **16-bit**, maximally 70 characters which is viewable by most phones and is based on the Universal Character Set (UCS)<sup>2</sup> or UTF-16.
2. PDU format discussed in earlier can prove to be difficult to employ. This being due to difficult human-readability of hexa-decimal. For many it would not be a natural or easy way to communicate.

At this stage and as it was implemented, the SMS text format of 160 character proves to be the best as it is a mode and format which is easy for all users who are familiar with text messages.

*Fourthly, what message characters are used to send the spatially informed SMS?*

This question is answered, given the preceding question. After looking at the possible SMS formats, it was seen that binary characters, hexadecimal character, the UTF-16 and the 7-bit alphabet as specified by the GSM 03.38, as stated in Section 2.3, can be used to send text messages. In the design of any system that may use the SMS as a means of communication between an SDI node, a **crucial** component which has to be in existence is a format converter from any data format to plain text. For instance, the internal mechanism of an

---

<sup>2</sup><http://en.wikipedia.org/wiki/UTF-16/UCS-2>

SDI node service could imply that data transfer be in XML, say a WFS. Thus the data is extracted by the node in XML, processed for the query but reported to the user in plain text. An obvious advantage of the plain text is not only direct readability but there are no compatibility issues if the communication is machine-to-machine or even to human reader, as plain text is deciphered easily.

*Finally, how can a Spatial Database Management System be integrated in an SMS system?*

A Spatial DBMS is a DBMS optimised to store and query data related to objects in space, including points, lines and polygons. Thus, in general it handles most of the general DBMS functionality, understands various numeric and character types of data and with the ability added by additional functionality to process spatial data types. Therefore to process data and query data from the Spatial DBMS, the usual routines and language supported by the DBMS for instance to Structured Query Language (SQL). Thus, in this case we can safely conclude that any Spatial DBMS can be used to serve as the SDI node. Examples include, Microsoft SQL Server, PostgreSQL and Oracle Spatial to mention a few. So therefore, how do we communicate with it to integrate it with an SMS system? For this, we need the following:

- i. A means to translate the SMS into a query that can be handled by the Spatial DBMS and this is handled by the parser as described earlier.
- ii. An implementation-dependent/independent database adapter to access the Spatial DBMS. An example we implemented here is the ***psycopg***. This adaptor works as a Python-PostgreSQL Database Adapter, allowing access to the PostgreSQL. This then allows for the formulation of the queries within the system. For this communication there exist several alternatives, which depend on the Spatial DBMS chosen.
- iii. Query result optimisation mechanism to allow for the conversion of the query result to plain text such that it can be successfully communicated through the SMS.

During this research project, we saw in Chapter 4, there was a Spatial DBMS used as an SDI node. We used PostgreSQL primarily because it is free, open-source, stable and fast. The execution of the queries was enabled by the *psycopg* adapter through python scripting for the main RISS components, to demonstrate this spatial query involving an SDI node. The RISS system, also provided a test the integration of the system with a PostgreSQL-PostGIS DBMS working as the SDI node which support spatio-analytic function executiton. Examples of the spatial analytic functions that are executed include buffering, containment and adjacency. Primarily, this minor implementation was created to demonstrate the capability of such queries in the SMS-Spatial DBMS integration.

Having answered the research questions above, ultimately it led to a fulfilment of the objectives as stated in Chapter 1. There, the general objective was stated as to develop architectural solutions for two-way SMS services on an SDI node. This is exhibited in Chapter 4 where it was developed from the requirements analysis done in Section 4.2, and illustrated in UML in Sections 4.4.1, 4.4.2 and 4.4.3.

To satisfy this general objective following specific objectives were met.

1. To design an architecture(s) of a two way SMS system involving an SDI node.
2. To develop software components to generate, receive, analysis, send, translate and store spatially informed SMS, using low cost hardware and software solutions.
3. To build a prototype system to implement SMS spatial information generation, receiving, sending and analysis.

### **6.3 Shortcomings and limitations of two-way SMS involving the SDI node**

The research questions were answered, objectives were fulfilled and an architecture for two way SMS service involving an SDI node was developed. From this research and through an analysis of the implementation, we observed and noted the following shortcomings of the architecture.

- i. Stemming from the size of the SMS we can have a query to provide or retrieve information in form of a 160 character '*data packets*', thus a limitation is that only relatively short messages can be sent at any one time, which translates to limited data transfer.
- ii. The use of phone keypad to enter information is at times is known to produce typographic errors as it is not so convenient to use. Also, this has led to the none usage of the standard English, as a form of SMS slang has emerged which introduces parsing problems, though we have tried to handle this by a translator in the RISS implementation.
- iii. For the SMS, in the event of high traffic, message delivery cannot be guaranteed as it might be minutes or longer before the message is delivered thus in the event of time dependent queries a problem might occur.
- iv. Another shortcoming noted was that in the architecture proposed if a GSM modem is used there is a limitation the number of messages that can be sent. This means that if several users request a service, a long response queue can occur as maximally 6 to 10 messages can be sent per minute.

## 6.4 Conclusion

The objectives of this research project were achieved and the research questions answered. An architecture for two-way SMS services involving an SDI was designed. A demonstrator implementation system was created using python for scripting and a PostgreSQL database as the SDI node. The design consists of a three tier architecture. The architectural design consists of the GSM modem as part of the message receiving component, SMS-to-email option as the message parsing component, the error detection component, message correction component, query formulation component, message sending component and the subject of interest positioning component. These components are accessed as services in the SMS-SDI service stack of the architecture. Free open-source solution are used to build the components except for the message sending component. The implementation of the use case (RISS) provides sufficient evidence to suggest that the designed architecture is feasible to communicate two-way spatial services within southern contexts. A use case scenario was chosen, the RISS and several users were used to test the system. We conclude that the designed architecture works and that it can successfully be built to allow for two-way SMS services involving an SDI node. The designed architecture provided spatial innovation in that SMSs could be used in conjunction with SDI nodes to communicate spatial information in a two way manner.

## 6.5 Recommendations

### 6.5.1 Current Research

In direct relation to the current research I recommend the following:

- i. The proposed architecture be implemented using a dedicated gateway. This comes from noting problems that are associated with a shared gateway as some incoming messages take longer to be delivered to the system folder. Thus causing a temporal delay which can be problematic if the incoming query is time dependant.
- ii. Community based information sharing via SMS be looked at. For instance, creating an architecture that facilitates the social affinity computation based on spatial location. Though this can be said to be a heavily server side application, we are then in a better position to come up with an architecture which provides a greater understanding of the two way communication with an SDI node.
- iii. Given that information of the phone can be obtained through a JSR, I recommend that the applicability of such technology in the information transfer via SMS be investigated, particularly in mobile devices available in the southern context.

### 6.5.2 Future Research

For future research i recommend the following:

- i. The aspect of bulk messages be researched further than it was done in this research. This would be in the arena of creating a robust messaging system which is capable of having multiple sockets and handle several messages simultaneously. This would lead use to understand how to integrate the SDI node or if successfully an SDI with a two way communication which is robust enough to handle bulk messages.
- ii. The aspect of positioning the subject of interest. As stated in this research project. With an spatial system, the aspect of location remains crucial. Therefore having looked at a number of possible ways to position the mobile phone, a crucial question is what is supported by the current mobile phone in the southern context and what components are required to position the phenomena of interest.
- iii. Though the concepts might be a bit similar, in an SDI several SDI nodes are interconnected. In this research we focused on an SDI node. Stemming from this I recommend research on the use of SMS technology to allow for
- iv. Though there was a minor discussion on free form messaging, I recommend that it be looked at further as this provides a means of messaging which has less rules and constraints thus be more robust and more user friendly.
- v. Creation of an architecture which supports ad-hoc query formulation via SMS to access the SDI or SDI node thereby using for example the Publish-Find-Bind Model as described in the OGC Reference Model<sup>3</sup>. For example, given a scenario where by a trained expert user, a doctor in the heart of a rural setup, who wishes to enquire information about a phenomena which is a combination of processes available on a the SDI. He would be able to find the services available from a registry, chose the services and datasets which had be published and execute the service. The results of the given geoprocessing being reported in a format that can be handled by an SMS

---

<sup>3</sup><http://www.opengeospatial.org/standards/orm>



# Bibliography

- [1] ADRIAENS, G., AND SCHREORS, D. From COGRAM to ALCOGRAM: Toward a Controlled English Grammar Checker. In *Proceedings of the 14th Conference on Computational linguistics* (Morristown, NJ, USA, 1992), Association for Computational Linguistics, pp. 595–601.
- [2] AHO, A., SETHI, R., AND ULLMAN, J. Compilers: Principles, Techniques, and Tools. *Reading, MA*, (1986).
- [3] ANNONI, A. Towards a SDI for Europe: An Overview and Areas for Research and Investigation. In *From OEEPE to EuroSDR: 50 years of European Spatial Data Research and Beyond Seminar of Honour* (2003), p. 31.
- [4] ATWELL, E. S. How to detect grammatical errors in a text without parsing it. In *Proceedings of the Third Conference on European chapter of the Association for Computational Linguistics* (Morristown, NJ, USA, 1987), Association for Computational Linguistics, pp. 38–45.
- [5] AYANLADE, A., ORIMOOGUNJE, I. O. O., AND BORISADE, P. B. Geospatial Data Infrastructure for Sustainable Development in Sub-Saharan Countries. *International Journal of Digital Earth* 1, 3 (2008), 247–258.
- [6] BAJAJ, R., RANAWEERA, S. L., AND AGRAWAL, D. P. GPS: Location-Tracking Technology. *Computer* 35, 4 (2002), 92–94.
- [7] BANKS, K. FrontlineSMS. <http://www.frontlinesms.com/what/>, August 2009.
- [8] BANSAL, B., CHOUDHURY, M., SARKAR, S., AND PRADIPTA RANJAN, R. Isolated Word Error Correction for Partially Phonemic Languages Using Phonetic Cues. In *the Proc of the Int. conf. on Knowledge based Computer Systems (KBCS)* (2004).
- [9] BOOCH, G., RUMBAUGH, J., AND JACOBSON, I. *Unified Modeling Language User Guide, The (Addison-Wesley Object Technology Series)*. Addison-Wesley Professional, 2005.
- [10] BREDEMEYER, D., AND MALAN, R. The Role of the Architect. *white paper published on the Resources for Software Architects web site, <http://www.bredemeyer.com/papers.htm>* (2002).

- [11] BRITZ, J., LOR, P., COETZEE, I., AND BESTER, B. Africa as a knowledge society: A reality check. *The International Information & Library Review* 38, 1 (2006), 25 – 40.
- [12] BROWN, J., SHIPMAN, B., AND VETTER, R. SMS: The Short Message Service. *Computer* 40, 12 (2007), 106–110.
- [13] CAMPBELL, B., RUIBO, L., POON, A., AND HILAIRE, D. Connecting Developing Regions Though SMS.
- [14] CHANG, Z. *Design and Implementation of Distributed Web Services in Mobile GIS Application*. PhD thesis, ITC, Enschede, 2003.
- [15] CLEMENTS, P., AND KAZMAN, R. *Software Architecture in Practices*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003.
- [16] COCKBURN, A. Basic Use Case Template. *Humans and Technology, Technical Report 96* (1998).
- [17] COLEMAN, D. A Use Case Template: draft for discussion. *Fusion Newsletter* (1998).
- [18] CZARNECKI, K., AND HELSEN, S. Classification of model transformation approaches. In *Proceedings of the 2nd OOPSLA Workshop on Generative Techniques in the Context of the Model Driven Architecture* (2003), Citeseer.
- [19] DAMERAU, F. J. A Technique for Computer Detection and Correction of Spelling Errors. *Communications of the ACM* 7, 3 (1964), 171–176.
- [20] DE BY, R. A., MORALES, J., AND LEMMENS, R. Methodical construction of sustainable SDI nodes. *Presented at the 8th Africa GIS conference and exhibition, Ouagadougou, Burkina Faso*. (September 2007), 17–21.
- [21] DE BY, R. A., WYTZISK, A., MORALES, J., AND VOGES, U. Construction Methods for Interoperable SDI Nodes. *GSDI-10* (February 2008).
- [22] DEMPSTER, A. Dilution of Precision in Angle-of-Arrival Positioning Systems. *Electronics Letters* 42, 5 (2006), 291–292.
- [23] DONNER, J., VERCLAS, K., AND TOYAMA, K. Reflections on MobileActive08 and the M4D Landscape. In *Proceedings of the First International Conference on M4D* (2008), pp. 73–83.
- [24] ERICSSON LABS. Mobile Location. <https://labs.ericsson.com/apis/mobile-location/>, January 2010.
- [25] ERL, T. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall PTR Upper Saddle River, NJ, USA, 2005.
- [26] FOSTER, V. Africa Infrastructure Country Diagnostic: Overhauling the Engine of Growth. *Unpublished Draft World Bank report* (September 2008).

- 
- [27] FOWLER, M. *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003.
- [28] FROST, R., HAFIZ, R., AND CALLAGHAN, P. Modular and efficient top-down parsing for ambiguous left-recursive grammars. *IWPT 2007*, 109.
- [29] GHIMIRE, D. Design of a Grid -Based Geo-Service Architecture. Master's thesis, ITC, Enschede, 2005.
- [30] GRINTER, R., AND ELDRIDGE, M. y do tngs luv 2 txt msg. In *Proceedings of the Seventh European Conference on Computer-Supported Cooperative Work ECSCW (2001)*, vol. 1, Citeseer, pp. 219–238.
- [31] GUPTA, P. Short Message Service: What, How and Where?, October 2009.
- [32] GUTHERY, S. B., AND CRONIN, M. J. *Mobile Application Development with SMS and the SIM Toolkit*. McGraw-Hill, New York, 2002.
- [33] HÄKKILÄ, J., AND CHATFIELD, C. 'It's like if you opened someone else's letter': user perceived privacy and social practices with SMS communication. In *MobileHCI '05: Proceedings of the 7th international conference on Human computer interaction with mobile devices & services* (New York, NY, USA, 2005), ACM, pp. 219–222.
- [34] HOLZINGER, A., NISCHELWITZER, A., AND MEISENBERGER, M. Mobile phones as a challenge for M-Learning: Examples for mobile interactive learning objects (MILOs). In *Proceedings of: Third IEEE International Conference on Pervasive Computing and Communication (PerCom 05), Kauai Island (HI)* (2005), pp. 307–311.
- [35] INGELS, P. A Robust Text Processing Technique Applied to Lexical Error Recovery. *CoRR cmp-lg/9702003* (1997).
- [36] INTERNATIONAL TELECOMMUNICATION UNION. Telecommunication/ICT Markets And Trends In Africa 2007. [http://www.itu.int/ITU-D/ict/statistics/material/af\\_report07.pdf](http://www.itu.int/ITU-D/ict/statistics/material/af_report07.pdf), Accessed: February 2010.
- [37] JAMES, J., AND VERSTEEG, M. Mobile phones in Africa: How much do we really know? *Social Indicators Research* 84, 1 (2007), 117–126.
- [38] JOSUTTIS, N. M. *SOA in Practice: The Art of Distributed System Design*. O'Reilly Media, Inc., 2007.
- [39] KEMIGHAN, M., CHURCH, K., AND GALE, W. A spelling correction program based on a noisy channel model. In *Proc. of the Thirteenth International Conf. on Computational Linguistics* (1990), pp. 205–210.
- [40] KHARE, R., AND TAYLOR, R. N. Extending the representational state transfer (rest) architectural style for decentralized systems. In *ICSE '04: Proceedings of the 26th International Conference on Software Engineering* (Washington, DC, USA, 2004), IEEE Computer Society, pp. 428–437.

- [41] KOETS, M. Development of Inverse Hyperbolic Positioning Using the GSM Cellular Telephone Network, 16-9222. [www.swri.org/3pubs/ird2002/16-9222.htm](http://www.swri.org/3pubs/ird2002/16-9222.htm), February 2010.
- [42] KRUCHTEN, P. *The Rational Unified Process: An Introduction, Second Edition*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2000.
- [43] LAKSHMISH, R., DEEPAK, P., RAMANA, P., KUTILA, G., DINESH, G., KARTHIK, V., AND SHIVKUMAR, K. Caesar: A context-aware, social recommender system for low-end mobile devices. *Mobile Data Management, IEEE International Conference on* 0 (2009), 338–347.
- [44] LARMAN, C. *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.
- [45] LE BODIC, G. *Mobile Messaging Technologies and Services: SMS, EMS and MMS*. John Wiley & Sons Inc, 2005.
- [46] LEIJEN, D., AND MEIJER, E. Parsec: Direct style monadic parser combinators for the real world. *Department of Information and Computing Sciences, Utrecht University, Tech. Rep. UU-CS-2001-35* (2001).
- [47] LEVENSHTEIT, V. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. In *Soviet Physics-Doklady* (1966), vol. 10.
- [48] LIND, M. A De-Construction of Wireless Device Usage. *Selected Readings on the Human Side of Information Technology* (2008), 245.
- [49] LIOU, H.-C. Development of an English Grammar Checker: A Progress Report. *CALICO journal* 9, 1 (1991), 57–70.
- [50] MAPHAM, W. Mobile phones-changing health care one SMS at a time. *Southern African Journal of HIV Medicine* 9, 4 (2008), 11.
- [51] MARSHALL, C. *Enterprise modeling with UML: Designing successful software through business analysis*. Addison-Wesley Professional, 2000.
- [52] MOBARAKI, A., MANSOURIAN, A., MALEK, M., AND MOHAMMADI, H. Application of Mobile GIS and SDI for Emergency Management. *ISPRS Commission Technique I. Symposium* (2006), 95–100.
- [53] MOMO, C. Wireless in Africa: Insights into Mobile Markets. *IT Professional* 7, 3 (2005), 34–38.
- [54] NEWCOMER, E., AND LOMOW, G. *Understanding SOA with Web Services (Independent Technology Guides)*. Addison-Wesley Professional, 2004.
- [55] NORVIG, P. How to Write a Spelling Corrector. <http://norvig.com/spell-correct.html>, December 2009.

- 
- [56] PAGÈS-ZAMORA, A., VIDAL, J., AND BROOKS, D. Closed-form solution for positioning based on angle of arrival measurements. In *IEEE PIMRC* (2002), vol. 4, Citeseer, pp. 1522–1526.
- [57] PEERSMAN, G., CVETKOVIC, S., GRIFFITHS, P., AND SPEAR, H. The global system for mobile communications short message service. *IEEE Personal Communications* 7, 3 (2000), 15–23.
- [58] RAJABIFARD, A., AND WILLIAMSON, I. Spatial data infrastructures: concept, SDI hierarchy and future directions. *Proceedings of GEOMATICS 80* (2001).
- [59] RAMAMRITHAM, K., BAHUMAN, A., AND DUTTAGUPTA, S. aAqua: A database-backed multilingual, multimedia community forum. *International Conference on Management of Data: Proceedings of the 2006 ACM SIGMOD international conference on Management of data* (2006), 784–786.
- [60] SENDALL, S., AND STROHMEIER, A. *From Use Cases to System Operation Specifications*, vol. 1939/2000. Springer Berlin / Heidelberg, 2000.
- [61] SILAGHI, R., AND STROHMEIER, A. Integrating CBSE, SoC, MDA, and AOP in a Software Development Method. In *Proceedings of the 7th IEEE International Enterprise Distributed Object Computing Conference, EDOC, Brisbane, Queensland, Australia, September* (2003), Citeseer, pp. 16–19.
- [62] THE INTERNATIONAL DEVELOPMENT RESEARCH CENTRE. ICT4D Atlas. <http://www.geog.mcgill.ca/atlas/>, February 2010.
- [63] TROSBY, F. SMS, the strange duckling of GSM. *Telektronikk* 100, 3 (2004), 187–194.
- [64] ZAPHIRIS, P., AND CHEE, S. A. *Cross-Disciplinary Advances in Human Computer Interaction: User Modeling, Social Computing, and Adaptive Interfaces*. Information Science Reference, 2009.
- [65] ZHAO, Y. Standardization of mobile phone positioning for 3 G systems. *IEEE Communications Magazine* 40, 7 (2002), 108–116.
- [66] ZHAO, Y., AND HOUSE, A. *Vehicle Location and Navigation Systems: Intelligent Transportation Systems*. Artech House (1997), 221–224.



# Appendix A

## Use Case Template

The below template is extracted from [16] and is used to model the Rain Information Service use case.

**Use Case:** <number><the name should be the goal as a short active verb phrase>

### CHARACTERISTIC INFORMATION

*Goal in Context:* <a longer statement of the goal, if needed>

*Scope:* <what system is being considered black-box under design>

*Level:* <one of: Summary, Primary task, Subfunction>

*Preconditions:* <what we expect is already the state of the world>

*Success End Condition:* <the state of the world upon successful completion>

*Failed End Condition:* <the state of the world if goal abandoned>

*Primary Actor:* <a role name for the primary actor, or description>

*Trigger:* <the action upon the system that starts the use case, may be time event>

### MAIN SUCCESS SCENARIO

<put here the steps of the scenario from trigger to goal delivery, and any cleanup after>

<step #><action description>

### EXTENSIONS

<put here there extensions, one at a time, each refering to the step of the main scenario>

<step altered><condition>: <action or sub.use case>

<step altered><condition>: <action or sub.use case>

### SUB-VARIATIONS

<put here the sub-variations that will cause eventual bifurcation in the scenario>

<step or variation # ><list of sub-variations>

<step or variation # ><list of sub-variations>

### RELATED INFORMATION (optional)

*Priority:* <how critical to your system / organization>

*Performance Target:* <the amount of time this use case should take>

*Frequency:* <how often it is expected to happen>

*Superordinate Use Case:* <optional, name of use case that includes this one>

---

*Subordinate Use Cases:* <optional, depending on tools, links to sub.use cases>Channel to primary actor: <e.g. interactive, static files, database>

*Secondary Actors:* <list of other systems needed to accomplish use case>

*Channel to Secondary Actors:* <e.g. interactive, static, file, database, time-out>

**OPEN ISSUES (optional)**

<list of issues about this use cases awaiting decisions>

**SCHEDULE**

*Due Date:* <date or release of deployment>...any other schedule / staffing information you need...

## **Appendix B**

# **Architectural views of the System by Stereotyped UML Diagrams**

**Table B.1** Architectural view of the System by Stereotyped UML Diagrams  
source:[29, 9, 51]

Architectural View	UML Diagram	Description
Structural	Class	Shows a static structure of a model with a set of classes, interfaces, and collaborations and their static relationships
	Object	Shows a set of objects and their relationships, snapshot of the detailed state of a system at a point of time
	Component	Shows a set of components and their relationships, represents a modular, deployable, and replaceable part of a system that encapsulates implementation and exposes a set of interfaces
	Deployment	Shows how instances of components and processes are configured for run-time execution on instances of processing nodes
Behavioural	Use Case	Organises the behaviours of the system, shows a set of use cases and actors, and their relationships
	Collaboration	Focused on the structural organisation of objects that send and receive messages, used to model flow control and illustrate coordination of object structure and control
	Statechart	Focused on the changing state of a system driven by events, used to model object life-cycles and model reactive objects (under interfaces, devices)
	Activity	Focused on the flow of control from activity to activity within a system, represents a state machine of a computation itself
	Sequence	Focused on time-ordering of messages, shows the explicit sequence of communications and is better for real time specifications and for complex scenarios

## Appendix C

# Summary of currently available products and technologies for SMS services

Tool	General Services	Language	Region	Open Source
W3C MobileOK Checker	null	null	global	yes
mobilisr	Bulk SMS, Multi-Media Messaging (MMS) or other Multi-Media, Premium SMS and Billing, USSD Services, Voting, Data Collection, Surveys, and Polling	English	Sub-Saharan Africa	yes
Polls (Pollimath)	Voting, Data Collection, Surveys, and Polling	English	Asia	no
CommCare	Voting, Data Collection, Surveys, and Polling, Information Resources/Information Databases	English, Swahili, Spanish	Sub-Saharan Africa	no
Agricultural Market Information Systems (AMIS)	Information Resources/Information Databases	Bengali, English	Asia	no

---

MapSwitch	Mobile Payments	null	Sub-Saharan Africa	yes
digitalics	null	Spanish, English	North America	yes
greporter	Voting, Data Collection, Surveys, and Polling	English	North America	yes
My Question	Interactive Voice Response (IVR), Information Resources/Information Databases	English	Sub-Saharan Africa	yes
komkom	null	French	Sub-Saharan Africa	no
mobile-researcher	Voting, Data Collection, Surveys, and Polling	English	Sub-Saharan Africa, Asia, Europe, North America	yes
Wannigame	null	French	Sub-Saharan Africa	no
Fluid-nexus	null	English, Spanish	North America	no
audioTagger	Voting, Data Collection, Surveys, and Polling, Location-Specific Services and GIS	English	Europe, North America	no
MyCO2Print	Voting, Data Collection, Surveys, and Polling	English	Sub-Saharan Africa	no
FrontlineSMS	Bulk SMS	English, French, Spanish	Sub-Saharan Africa, Asia	yes
GeoChat	null	English	Sub-Saharan Africa, Asia, North America	yes
Sub-Saharan Africa	Bulk SMS	English, Zulu, Xhosa	Sub-Saharan Africa	no
Souktel	Bulk SMS, Voting, Data Collection, Surveys, and Polling, Mobile Social Network/Peer-to-peer, Information Resources/Information Databases	Arabic, Kurdish, English, French, Somali, Spanish	Sub-Saharan Africa, Middle East and North Africa	yes

*Appendix C. Summary of currently available products and technologies for SMS services*

Claim Mobile	Voting, Data Collection, Surveys, and Polling, Information Resources/Information Databases	English	Sub-Saharan Africa	yes
NAFIS (National Farmers Information Service)	Interactive Voice Response (IVR), Information Resources/Information Databases	Kiswahili, English	Sub-Saharan Africa	no
Veeker	Multi-Media Messaging (MMS) or other Multi-Media	English	Sub-Saharan Africa, North America	no
minerva-mobile	Mobile Payments	English, French	North America	no
Pigeon	Interactive Voice Response (IVR), Mobile Social Network/Peer-to-peer	English, Spanish	North America	no
DEWN (Disaster Emergency Warning Network)	Bulk SMS, USSD Services	English, Sinhalese	Asia	no
The Extraordinaries	web-based application/web service	English	North America	no
Freedom Fone	Interactive Voice Response (IVR)	All	Sub-Saharan Africa	no
Ushahidi	Voting, Data Collection, Surveys, and Polling, Location-Specific Services and GIS	English, French, Arabi	Sub-Saharan Africa, Asia, Australia and Oceania, Central America and the Caribbean, Europe, Middle East and North Africa, North America, South America	yes

---

TxtAlert	Bulk SMS	English	Sub-Saharan Africa	yes
trixbox	Interactive Voice Response (IVR)	English	Sub-Saharan Africa	no
frogtex	Touchscreen interface to register transactions. Camera phone for barcode reading. Accounts are synchronized with server.	Spanish, English	South America	yes
Handheld Human Rights	Voting, Data Collection, Surveys, and Polling	null	Asia	yes
Catalista	Information Resources/Information Databases	English	North America	yes
gather	Voting, Data Collection, Surveys, and Polling	English	Sub-Saharan Africa	yes
voiceglue	Interactive Voice Response (IVR)	English	North America	yes
BeVocal Cafe	Interactive Voice Response (IVR)	null	North America	no
jvoicexml	Interactive Voice Response (IVR)	null	North America	yes
freeswitch	Interactive Voice Response (IVR)	null	Sub-Saharan Africa, Asia, Australia and Oceania, Central America and the Caribbean, Europe, Middle East and North Africa, North America, South America	yes

Asterisk	Interactive Voice Response (IVR)	null	Sub-Saharan Africa, Asia, Australia and Oceania, Central America and the Caribbean, Europe, Middle East and North Africa, North America, South America	yes
mxit	Games, Mobile Social Network/Peer-to-peer	null	Sub-Saharan Africa, Asia	no
TextMarks	Bulk SMS, Premium SMS and Billing, Voting, Data Collection, Surveys, and Polling	null	North America	yes
Avaaj Otalo	Interactive Voice Response (IVR)	Gujarati	Asia	no
GUIDE	Information Resources/Information Databases	null	Sub-Saharan Africa	no
Cryptosms	null	English, French, Spanish, Russian, Japanese, Slovenian, Portuguese, Norwegian, Polish, ...	Australia and Oceania, Central America and the Caribbean, Europe	yes



## Appendix D

# Cell-Id Look-up Application for Java ME

Listing D.1: Cell-id Look-up Application for Java ME source:[24]

```
import java.io.ByteArrayOutputStream;
import java.io.IOException;
import java.io.InputStream;

import javax.microedition.midlet.*;
import javax.microedition.io.Connector;
import javax.microedition.io.HttpConnection;
import javax.microedition.lcdui.*;

import org.json.me.JSONException;
import org.json.me.JSONObject;

/**
 * A minimal example MIDlet implemented
 * with Cell-ID look-up API.
 */
public class CellIdLookUpMidlet extends MIDlet
    implements CommandListener, Runnable {

    // **** Set the developer key here ****
    private final static String APIKEY = "";

    private Command exitCommand;
    private Cell cell;
    private Form form;

    private StringItem itemLongitude, itemLatitude;
    private StringItem itemAccuracy, itemName;
    private double longitude, latitude, accuracy;
    private String cellName;

    public CellIdLookUpMidlet() {

        cell = new Cell();

        exitCommand = new Command("Exit", Command.EXIT, 1);
```

---

```

form = new Form(null);

itemLongitude = new StringItem("longitude:_",
    Double.toString(longitude));
itemLatitude = new StringItem("latitude:_",
    Double.toString(latitude));
itemAccuracy = new StringItem("accuracy:_",
    Double.toString(accuracy));
itemName = new StringItem("name:_", cellName);

form.insert(0, itemLongitude);
form.insert(1, itemLatitude);
form.insert(2, itemAccuracy);
form.insert(3, itemName);

form.addCommand(exitCommand);
form.setCommandListener(this);

Thread t = new Thread(this);
t.start();
}

public void run() {
    updatePosition();
}

private void updateForm() {
    itemLongitude.setText(Double.toString(longitude));
    itemLatitude.setText(Double.toString(latitude));
    itemAccuracy.setText(Double.toString(accuracy));
    itemName.setText(cellName);
}

protected void startApp() {
    Display.getDisplay(this).setCurrent(form);
}

protected void pauseApp() {}
protected void destroyApp(boolean bool) {}

public void commandAction(Command cmd, Displayable d) {
    if(cmd==exitCommand) {
        destroyApp(false);
        notifyDestroyed();
    }
}

/**
 * updatePosition
 * The format of the position data is either xml or json.
 * In this example we are using json
 */
public void updatePosition() {
    StringBuffer url = new StringBuffer();
    url.append("http://cellid.labs.ericsson.net/json/lookup");
    url.append("?cellid=").append(cell.getCellId());
    url.append("&mnc=").append(cell.getMnc());
    url.append("&mcc=").append(cell.getMcc());

```

```

url.append("&lac=").append(cell.getLac());
url.append("&key=").append(API_KEY);

try {
    byte[] data = getHttp(url.toString());
    if(data!=null) {
        parseJSON(new String(data));
        updateForm();
    }
} catch (IOException e) {
    e.printStackTrace();
}
}

/**
 * parseJSON
 *
 * For JSON support we are using org.json.me
 * The JSON package can be downloaded from here:
 * <a href="http://www.json.org/java/org.json.me.zip">br />
 * title="www.json.org/java/org.json.me.zip">br />
 * >www.json.org/java/org.json.me.zip<br />
 * </a> */
public void parseJSON(String jsonString) {
    try {
        JSONObject o = new JSONObject(jsonString);
        JSONObject pos = o.getJSONObject("position");
        this.longitude = pos.getDouble("longitude");
        this.latitude = pos.getDouble("latitude");
        this.accuracy = pos.getDouble("accuracy");
        this.cellName = pos.optString("name");
    } catch (JSONException e) {
        e.printStackTrace();
    }
}

static public byte[] getHttp(String url)
throws IOException {

    HttpURLConnection c = null;
    InputStream is = null;
    byte[] data = null;

    try {
        c = (HttpURLConnection)Connector.open(url);
        int rc = c.getResponseCode();

        // handle the HTTP response codes used by the API
        if(rc!=HttpURLConnection.HTTP_OK) {
            switch(rc) {
                case HttpURLConnection.HTTP_NO_CONTENT:
                    throw new IOException("The cell could not be "+
                        "found in the database");
                case HttpURLConnection.HTTP_BAD_REQUEST:
                    throw new IOException("Check if some parameter "+
                        "is missing or misspelled");
                case HttpURLConnection.HTTP_UNAUTHORIZED:
                    throw new IOException("Make sure the API key is "+

```

---

```

        "present_and_valid");
    case HttpURLConnection.HTTP_FORBIDDEN:
        throw new IOException("You have reached the limit "+
            "for the number of requests per day. The maximum "+
            "number of requests per day is currently 500.");
    case HttpURLConnection.HTTP_NOT_FOUND:
        throw new IOException("The cell could not be found "+
            "in the database");
    default:
        throw new IOException("HTTP response code: " + rc);
    }
}

is = c.openInputStream();

int actual = 0;
int len = (int)c.getLength();
if(len>0) {
    int bytesread = 0 ;
    // the server returned a length so we can allocate
    // a byte array directly
    data = new byte[len];
    // loop until there is nothing more to read or
    // until buffer is full
    while((bytesread != len) && (actual != -1)) {
        actual = is.read(data, bytesread, len - bytesread);
        bytesread += actual;
    }
} else {
    // the server is not returning a length
    // so we need a ByteArrayOutputStream
    ByteArrayOutputStream bos = new ByteArrayOutputStream();
    // create a 256 bytes read buffer to
    // improve reading performance
    byte buf[] = new byte[256];
    // loop for as long as we get data from the server
    // and add it to the output stream
    while((actual = is.read(buf, 0, 256))!= -1) {
        bos.write(buf, 0, actual);
    }
    bos.flush();
    data = bos.toByteArray();
}
} catch (ClassCastException e) {
    throw new IllegalArgumentException("Not an HTTP URL");
} finally {
    try { if (c != null) c.close(); } catch(IOException e) {};
    try { if (is != null) is.close(); } catch(IOException e) {};
}
return data;
}

/**
 * Cell class to handle cell id parameters.
 * Note that this is Sony Ericsson specific code.
 */
public class Cell {

```

```
private String id;
private String lac;
private String mcc;
private String mnc;

public Cell() {
    update();
}

public void update() {
    try {
        id = System.getProperty("com.sonyericsson.net.cellid");
        lac = System.getProperty("com.sonyericsson.net.lac");
        mcc = System.getProperty("com.sonyericsson.net.cmcc");
        mnc = System.getProperty("com.sonyericsson.net.cmnc");
    } catch (Exception e) {
        e.printStackTrace();
    }
}

public String getCellId() { return id; }
public String getLac() { return lac; }
public String getMcc() { return mcc; }
public String getMnc() { return mnc; }
}
```



## Appendix E

# Code Extracts of the RISS Engine and its components

Listing E.1: SMS Slang to Normal English Translator

```
def multiwordReplace(text , wordDic):
    v=wordDic[ text ]

# the dictionary has target_word : replacement_word pairs
lis = []
f = open( 'csv.txt' , 'r' )
for x in f:
    lis.append(x.rstrip( '\n' ))
f.close()

dic = {}
for line in lis:
    key, value = line.split( ',' )
    dic[key] = value
wordDic = dic

# call the function and get the changed text
print "speller( 'word' )"

def speller(word):
    try:
        text=correct(word).upper()
        print text
        meaning = wordDic[ text ]
        return meaning
    except Exception:
        print 'Word_not_in_dictionary'
```

Listing E.2: Profile building component

```
...
...
else:
```

---

```

try:
    conn = psycpg2.connect("dbname=\ac{SMS}_gateway_host=itcnt07.itc
                           .nl_port=5432_user=shelton_password=victoria")
    curs = conn.cursor()
    ADDON.strip('')
    statement="UPDATE_userdata.SET_plotname='"+ ADDON2.strip('') +"'
              "+ "WHERE_phonenumber=" + NUMBER
    print statement
    curs.execute(statement)
    conn.commit()
    conn.close()
except Exception:
...
...

```

### Listing E.3: Mail Sending using the win32com.client

```

import win32com.client
def mailer(NUMBER, REPORT):

    olMailItem = 0x0
    obj = win32com.client.Dispatch("Outlook.Application")
    newMail = obj.CreateItem(olMailItem)
    newMail.Subject = "Message_Report_to_"+ NUMBER
    newMail.Body = REPORT
    newMail.To = NUMBER+"@\ac{SMS}.itc.nl"
    newMail.Send()

    print 'Message_sent_successfully_to_' + NUMBER

    return None

```

### Listing E.4: The RISS Engine

```

import os, sys
import random, psycpg2
#
=====

#Class to check for the filename and return the name of the file in the
#  directory
#
=====

class DirectoryWalker:
    # a forward iterator that traverses a directory tree

    def __init__(self, directory):
        self.stack = [directory]
        self.files = []
        self.index = 0

    def __getitem__(self, index):
        while 1:
            try:
                file = self.files[self.index]
                self.index = self.index + 1

```

```

except IndexError:
    # pop next directory from stack
    self.directory = self.stack.pop()
    self.files = os.listdir(self.directory)
    self.index = 0
else:
    # got a filename
    fullname = os.path.join(self.directory, file)
    if os.path.isdir(fullname) and not os.path.islink(
        fullname):
        self.stack.append(fullname)
    return fullname
#
=====

#           Reading the working directory and obtain the filename and its
#           contents
#
=====

def directFname():
    work_path = 'C:/cygwin/var/spool/\ac{SMS}/incoming' # setting the
        working directory

    if os.path.exists(work_path) == True:

        if os.listdir(work_path)==[]: # check if the directory is empty
            or not

            print "There are no new messages"
        else:
            for file in DirectoryWalker(os.path.abspath('C:/cygwin/var/
                spool/\ac{SMS}/incoming')):

                f = open(file, "rb") # opening the file for reading
                f.seek(0) # moving the file object to absolute position
                    0 i.e from the start of file
                placename=f.read()
                from mailReader import reader
                (REQUEST, REPORT, NUMBER)=reader()

                if REQUEST=="ZERO":
                    print "NO new messages"
                else:

                    from geoloc import geo
                    location = geo(REQUEST)
                    print "The geographic location of request is " +
                        location
#
=====

#           Writing the geocoded results to a file for \ac{SMS} reply

    out = open ('C:/cygwin/var/spool/\ac{SMS}/outgoing/'+
        str(random.randint(1, 99999))+ os.listdir(
        work_path)[0] , 'w') # The textfile created

```

---

```

t=location.split(',')
longitude=t[3]
latitude=t[2]
accuracy=t[1]
Longi=float(t[3])
Latit=float(t[2])

from transform import trans
PIX = trans(Longi, Latit)

pixelX= PIX[0]
pixelY= PIX[1]

from PiXimager import buien # obtain
(CellX, CellY, pvalue)=buien(pixelX, pixelY)
from PiXForecast import buienF
(pvalue1, pvalue2)=buienF(pixelX, pixelY)

from classifier import classified # import
classifying option

currentCond = classified(pvalue)
minCond30 = classified(pvalue1)
minCond60 = classified(pvalue2)

REPORT = 'Now:_'+ currentCond + '_30min:_'+ minCond30
        + '_1Hr:_'+ minCond60

print "The_general_condition_is_most_likely_" +
        REPORT + "''"

from phonebook import phone

pNUMBER= NUMBER.lstrip('0')

USERNAME = phone(pNUMBER)

from \ac{SMS}2Emailmailer import mailer # import
mailing program

x=mailer(NUMBER, REPORT)
#print "message Sending process here"

from mailCleaner import cleaner
clean=cleaner()

counter = random.randint(1, 99999)

try:

    conn = psycopg2.connect("dbname=\ac{SMS}_stuff_
        host=localhost_port=5432_user=\ac{SMS}_
        password=\ac{SMS}")
    curs = conn.cursor()

```



---

```
sc.enter(1, 1, do_something, (sc,))
s.enter(1, 1, do_something, (s,))
s.run()
...
..
```

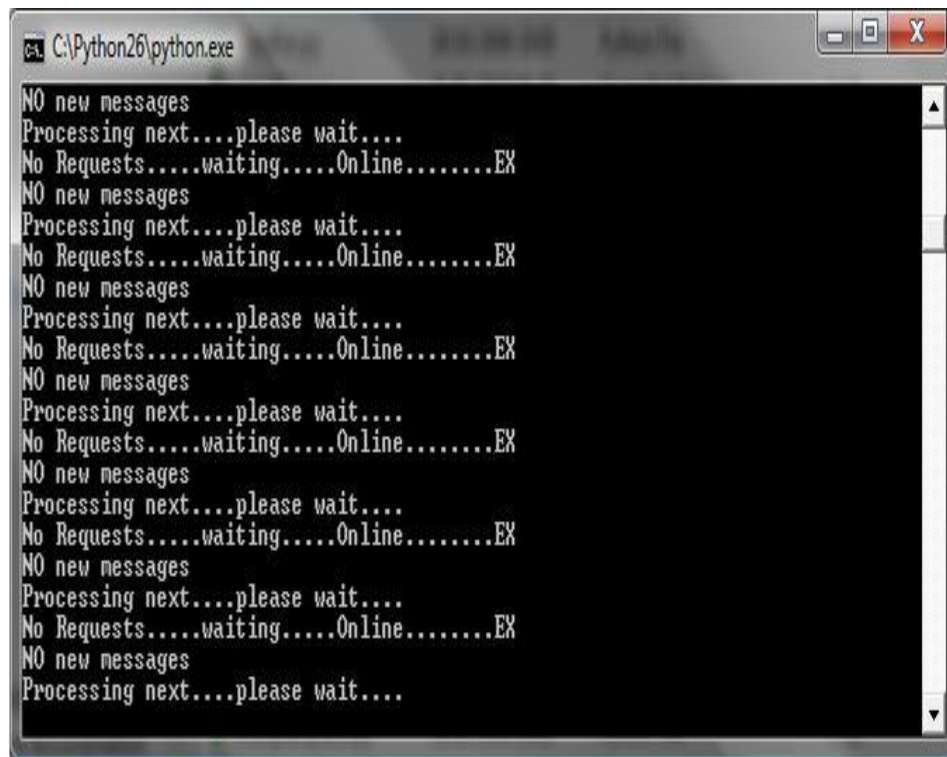


Figure E.1: RISS in idle mode waiting for SMSs

## Appendix F

# Detailed Description of the Operator Side SMS System Architecture

### F.1 SMS-enabled GSM Architectures

The SMS is part of the GSM network and to clearly define its role, we look at its position in an SMS enabled GSM network. With reference to Figure F.1, the Short Message Centre (SMC) is the component that has the role for the storage and transmission of messages to and from the mobile device i.e the short message entity (SME) located within the GSM network. In a bid to integrate with other networks the SMS gateway MSC (SMS GWMSC) receives messages from other networks and submits the message to the serving SMSC using the Signalling System Number 7 (SS7 or C7). From the home location register(HLR) profile information of the mobile and also about the routing information for the subscriber is obtained using the SS7. The HLR is the main database in a mobile network also holds the information on the area (covered by a MSC) where the mobile is currently situated. Using this information the GMSC would be able to pass on the text message to the correct Mobile Switching Centre (MSC). Within the GSM network the MSC is responsible for switching connections between mobile stations or between mobile stations and the fixed network [31].

Temporary information about the mobile like the mobile identification cell identification is contained in the Visitor Location Register(VLR) and corresponds to each MSC . Using information from the VLR the MSC is able to switch the information (short message) to the corresponding Base Station System (BSS), which transmits the short message to the mobile. The BSS consists of transceivers, which send and receive information over the air interface, to and from the mobile station. a consequence of passing the the information over the signaling channels is that the mobile can receive messages during a data or voice call [31].

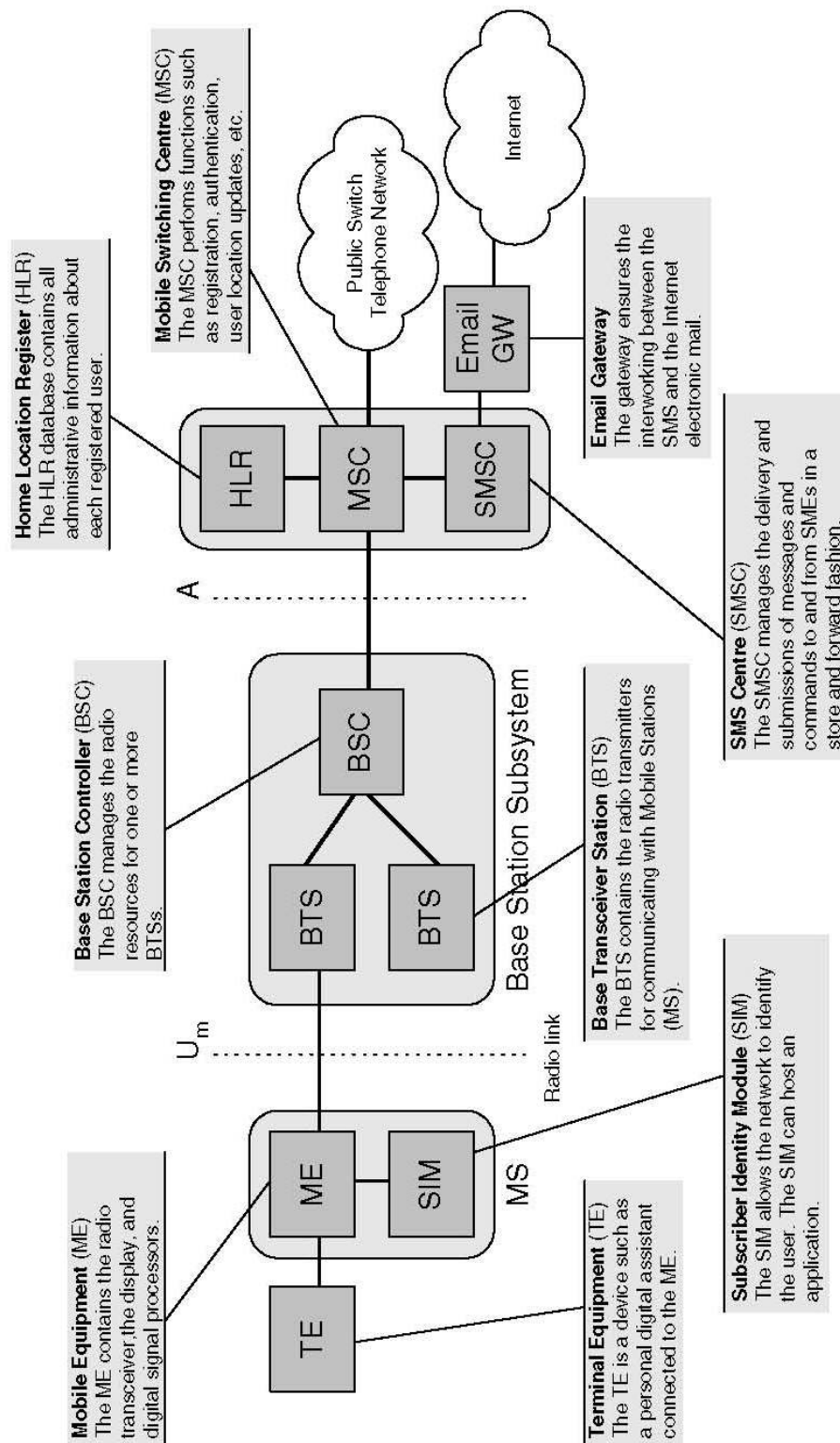


Figure F.1: Short Message Service-enabled GSM Network Architecture *source: [45]*

### F.1.1 The GSM Protocol Architecture

The architecture consists of three layers, the Physical, Data link layer and Message Layer as shown on Fig F.2.

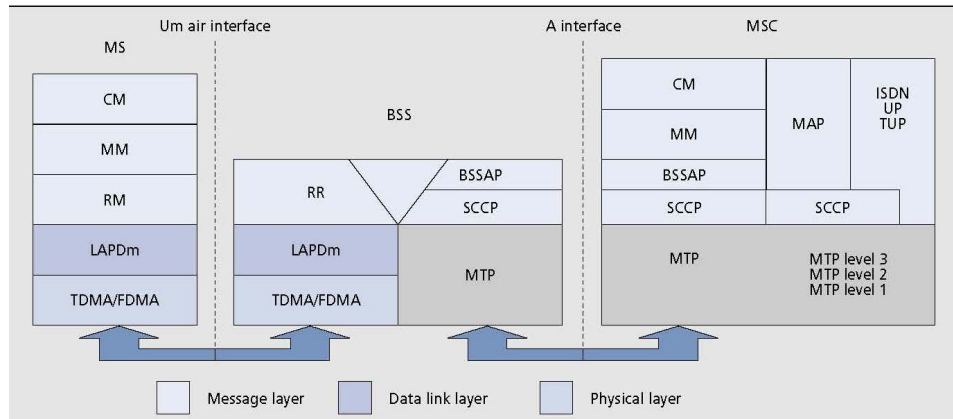


Figure F.2: The GSM Protocol Architecture *source: [57]*

**The Physical Layer** This layer on the bottom consists of the radio link on the radio channel which are multiplexed on an aggregate of the Transportation Demand Management (TDM) slots in the GSM network[57].

**Data link Layer** This employs a modified version of the Link Access Protocol for the D channel(LAPD) as used in ISDN consequently it is called the Link access protocol on the Dm channel (LAP-Dm) an it operates on the Um Air interface. But across the A interface, the Message Transfer Part (MTP), Layer 2 of Signaling System number 7 is used [57].

**Message Layer** The third layer of the GSM signaling protocol is divided into three sublayers

- Radio Resource management (RR)
- Mobility Management (MM)
- Connection Management (CM)

For the three Sub layers of the Message Layer. The Connection Management (CM)sublayer has the role of managing all the call-related supplementary services, Short Message Service and call-independent supplementary services support. The Mobile Management (MM)sublayer provides functions to create, maintain, and drop a connection between the MS and the MSC, over which an instance of the CM sublayer can exchange information with its peer. Other roles performed by this sublayer is are location updating, IMSI management, and Temporary Mobile Subscriber Identity (TMSI) identification, authentication, and reallocation. The Radio Resource (RR) sublayer establishes the physical connection connection over the radio link to transmit call-related signaling

information such as the establishment of the signaling and traffic channel between the MS and the BSS.

## F.2 General SMS Network Architecture

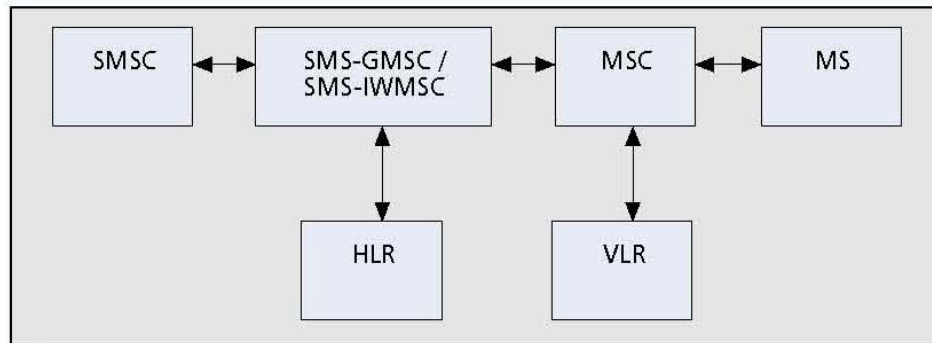


Figure F.3: General SMS Network Architecture *source: [57]*

As illustrated in Figure F.3 the general architecture of an SMS Network comprise of the following. It has the SMSC, the SMS-GMSC / SMS-IWMSC, the MSC, the MC, the HLR, The VLR. The role and functionality whose which is described in F.1. In addition to the general roles as in F.1 given a roaming environment the visited public limited mobile network (PLMN) reroutes the SMS to an appropriate SMS-IWMSC. The SMSC identifies each short message uniquely by adding a time stamp in the SMS-DELIVER TP-SCTS (TP-Service-Centre-Time-Stamp) field. Accuracy i.e to the second is provided on arrival to the SMSC which assures different time stamps for SMSs that it receives within a second separation from each other. The Mobile Station (MS) has to be able to receive/submit a short message Transport Protocol Data Unit (TPDU), and then return a delivery report upon successful reception. The MS notifies the network when it has memory capacity available to receive one or more messages given that it had rejected a short message because its memory capacity was exceeded [45].

## F.3 SMS Protocol Stack

The SMS protocol stack is composed of four layers:

- The Application Layer
- The Transfer Layer
- The Relay Layer
- The Link Layer

Generally in the development of SMS-based applications, the transfer layer (SM-TL), see Figure F.4, is the layer in which the applications are directly based it then services the Application Layer (SM-AL) [45]. The SM-TL in operation would conduct the PDU exchanges, the type of which are as listed in Table F.1.

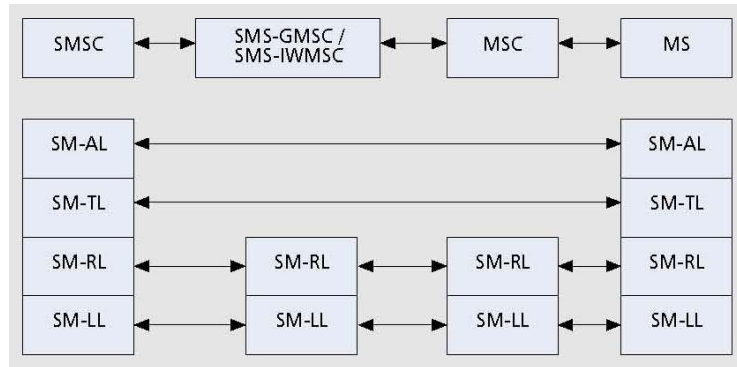


Figure F.4: The SMS Protocol Architecture (*source:[57]*)

**Table F.1** Transport Protocol Data Unit Types

TPDU Type	Function
SMS-Deliver	Conveying a short message from the SMSC to the MS
SMS-Deliver-Report	Conveying a failure cause
SMS-Submit	Conveying a short message from the MS to the SMSC
SMS-Submit-Report	Conveying a failure cause
SMS-Status-Report	Conveying a status report from the SMSC to the MS
SMS-Command	Conveying a command from the MS to the SMSC

### Application Layer

An implementation of the SM-AL would be handled in the context of applications handling the SMS transaction like message sending, receiving and content interpretation [45].

### Transfer Layer

Within the transfer layer an SMS is regarded as a sequence of octets, an entity having exactly eight bits, with information pertaining to message length, message originator or recipient, date of reception [45].

### Relay Layer

Across the network, the Short Message Relay Layer (SM-RL), facilitates the transport of the message which can be stored at various stages depending on

availability of the elements of the subsequent stages within the communication channel. The HLR interrogation, message reception and switching are also done at the SM-RL apart from SMS InterWorking MSC (SMS-IW MSC) which is the reception of messages from a mobile network and submission to the serving SMSC [45].

#### **Link Layer**

The link layer allows the transmission of the message at the physical level. For this purpose, the message is protected to cope with low-level channel errors. The link layer is also known as the SM-LL for Short-Message-Link-Layer. The stack of transport protocol layers for SMS is shown in Figure 3.6. For transport purposes, an application maps the message content and associated delivery instructions onto a Transfer Protocol Data Unit (TPDU) at the transfer layer. ATPDU is composed of various parameters indicating the type of the message, specifying whether or not a status report is requested, containing the text part of the message [57].



# Appendix G

**Statistics on Internet and Mobile Phone usage in Africa (source: [36])**

	Population	Main telephone lines		Mobile subscribers		Internet users	
	000s	000s	per 100 inhabitants	000s	per 100 inhabitants	000s	per 100 inhabitants
Algeria	33'354	2841.3	8.52	20'998.0	62.95	2'460.0	7.38
Egypt	75'437	10807.7	14.33	18'001.1	23.86	6'000.0	7.95
Libya	5'968	483.0	8.09	3'927.6	65.81	232.0	3.96
Morocco	30'735	1266.1	4.12	16'004.7	52.07	6'100.0	19.85
Tunisia	10'210	1'268.5	12.42	7'339.1	71.88	1'294.9	12.68
<b>North Africa</b>	<b>155'704</b>	<b>16666.6</b>	<b>10.70</b>	<b>66'270.5</b>	<b>42.56</b>	<b>16'086.9</b>	<b>10.34</b>
South Africa	47'594	4729.0	9.97	39'662.0	83.33	5'100.0	10.75
<b>South Africa</b>	<b>47'594</b>	<b>4729.0</b>	<b>9.97</b>	<b>39'662.0</b>	<b>83.33</b>	<b>5'100.0</b>	<b>10.75</b>
Angola	15'802	98.2	0.62	2'264.2	14.33	85.0	0.55
Benin	8'703	77.3	0.89	1'056.0	12.13	700.0	8.04
Botswana	1'760	136.9	7.78	979.8	55.68	60.0	3.40
Burkina Faso	13'634	94.8	0.70	1'016.6	7.46	80.0	0.59
Burundi	7'834	31.1	0.41	153.0	2.03	60.0	0.77
Cameroon	16'601	100.3	0.61	2'252.5	13.80	370.0	2.23
Cape Verde	519	71.6	13.80	108.9	20.99	29.0	6.09
Central African Rep.	4'093	10.0	0.25	100.0	2.48	13.0	0.32
Chad	10'032	13.0	0.13	466.1	4.65	60.0	0.60
Comoros	819	16.9	2.12	16.1	2.01	21.0	2.56
Congo	4'117	15.9	0.40	490.0	12.25	70.0	1.70
Côte d'Ivoire	18'454	260.9	1.41	4'065.4	22.03	300.0	1.63
D.R. Congo	59'320	9.7	0.02	4'415.0	7.44	180.0	0.30
Djibouti	807	10.8	1.56	44.1	6.37	11.0	1.36
Equatorial Guinea	515	10.0	1.99	96.9	19.26	8.0	1.55
Eritrea	4'560	37.5	0.82	62.0	1.36	100.0	2.19
Ethiopia	79'289	725.1	0.91	866.7	1.09	164.0	0.21
Gabon	1'406	36.5	2.59	764.7	54.39	81.0	5.76
Gambia	1'556	52.9	3.40	404.3	25.99	58.0	3.82
Ghana	22'556	356.4	1.58	5'207.2	23.09	609.8	2.70
Guinea	9'603	26.3	0.33	189.0	2.36	50.0	0.52
Guinea-Bissau	1'634	10.2	0.76	95.0	7.10	37.0	2.26
Kenya	35'106	293.4	0.84	6'484.8	18.47	2'770.3	7.89
Lesotho	1'791	48.0	2.67	249.8	13.92	51.5	2.87
Liberia	3'356	6.9	0.21	160.0	4.87	...	...
Madagascar	19'105	129.8	0.68	1'045.9	5.47	110.0	0.58
Malawi	13'166	102.7	0.80	429.3	3.33	59.7	0.45
Mali	13'918	82.5	0.59	1'513.0	10.87	70.0	0.50
Mauritania	3'158	34.9	1.10	1'060.1	33.57	100.0	3.17
Mauritius	1'256	357.3	28.45	722.4	61.50	300.0	24.10
Mozambique	20'158	67.0	0.33	2'339.3	11.60	178.0	0.90
Namibia	2'052	138.9	6.84	495.0	24.37	80.6	3.97
Niger	14'426	24.0	0.17	323.9	2.32	40.0	0.28
Nigeria	134'375	1688.0	1.26	32'322.2	24.05	8'000.0	5.95
Rwanda	9'230	16.5	0.18	314.0	3.40	65.0	0.70
S. Tomé & Príncipe	160	7.6	4.74	18.0	11.51	29.0	18.11
Senegal	11'936	282.6	2.37	2'982.6	24.99	650.0	5.45
Seychelles	81	20.7	25.44	70.3	86.52	29.0	35.67
Sierra Leone	5'678	24.0	0.49	113.2	2.21	10.0	0.19
Somalia	8'496	100.0	1.22	500.0	6.08	94.0	1.11
Sudan	36'993	636.9	1.72	4'683.1	12.66	3'500.0	9.46
Swaziland	1'029	44.0	4.27	250.0	24.29	41.6	4.02
Tanzania	39'025	157.3	0.40	5'767.0	14.78	384.3	1.00
Togo	6'306	82.1	1.30	708.0	11.23	320.0	5.07
Uganda	29'856	108.1	0.36	2'008.8	6.73	750.0	2.51
Zambia	11'861	93.4	0.79	1'663.0	14.02	500.0	4.22
Zimbabwe	13'085	331.7	2.54	832.5	6.36	1'220.0	9.32
<b>Sub-Saharan</b>	<b>719'220</b>	<b>7080.4</b>	<b>0.99</b>	<b>92'220.0</b>	<b>12.90</b>	<b>22'499.7</b>	<b>3.16</b>
<b>Africa</b>	<b>922'510</b>	<b>28475.9</b>	<b>3.10</b>	<b>198'153.0</b>	<b>21.58</b>	<b>43'686.7</b>	<b>4.77</b>

# Index

- GPS, 18
- Dead reckoning, 18
- Geographic Information, 1
- Global System for Mobile  
communication, 9
- JavaScript Object Notation, 33
- Representational State Transfer  
(REST) Interface, 33
- Service-Oriented Architecture, 39
- Short Message Service, 1
- Unified Modeling Language, 39