GSM MSC/VLR Unstructured Supplementary Service Data (USSD) Service

Egemen Taskin
Abstract

GSM MSC/VLR Unstructured Supplementary Service Data (USSD) Service

Egemen Taskin

Although the Unstructured Supplementary Service Data (USSD) service is one of the most used GSM services such as checking the balance of prepaid cards, it is not well known by subscribers and it is so often confused with a popular GSM service, Short Message Service (SMS). While SMS is based on the characteristics of storing and forwarding data, USSD is session based and real-time.

Technically, USSD service allows the Mobile Station (MS) user and a Public Land Mobile Network (PLMN) operator defined application to communicate in a way which is transparent to the MS and to intermediate network entities.[6]

This master thesis specifies all the phases in the development of USSD service in Mobile Switching Center (MSC)/Visitor Location Register (VLR) and its integration with an USSD Gateway. Meanwhile, it shows how a message passing functional programming language (Erlang) fits to develop a highly concurrent, available, fault-tolerant and distributable telecommunication system.
Acknowledgements

There are some special people who have helped me in the preparation of this thesis.

Firstly, I would like to thank Martin Kjellin, Product Manager at Mobile Arts AB, for supervising my thesis. I would like to thank Mobile Arts AB for providing an invaluable sponsorship.

Also, I would like to thank Ping Wu, Associate Professor at Department of Engineering Sciences / Uppsala University, for reviewing my thesis.

Finally, I would like to thank my big hearted aunt Hatice Taskin for being my angel investor and my family for supporting me with an invaluable trust and patience.
# Contents

## 1 Introduction
- 1.1 Motivation ......................... 13
- 1.2 Purpose & Tasks .................. 14
- 1.3 Methodology ...................... 14
- 1.4 Related Works ................... 16
- 1.5 Overview of the thesis ........... 16

## 2 Background
- 2.1 Basics of GSM ................... 18
  - 2.1.1 Introduction to GSM ........ 18
  - 2.1.2 GSM Networks ............... 18
    - 2.1.2.1 GSM Architecture ........ 19
    - 2.1.2.2 Geographical Network Structure 19
      - 2.1.2.2.1 Cell .................. 19
      - 2.1.2.2.2 Location Area .......... 20
      - 2.1.2.2.3 MSC Area ............. 20
      - 2.1.2.2.4 PLMN Area ............ 21
    - 2.1.2.3 Components ............... 21
      - 2.1.2.3.1 Switching System .... 21
        - 2.1.2.3.1.1 MSC .............. 22
        - 2.1.2.3.1.2 HLR .............. 22
        - 2.1.2.3.1.3 VLR .............. 23
        - 2.1.2.3.1.4 EIR .............. 23
        - 2.1.2.3.1.5 G-MSC ........... 23
      - 2.1.2.3.2 Base Station System .. 23
        - 2.1.2.3.2.1 BTS .............. 23
        - 2.1.2.3.2.2 BSC .............. 23
        - 2.1.2.3.2.3 TRAU ............. 23
  - 2.2 Erlang .......................... 24
    - 2.2.1 General consideration ....... 24
    - 2.2.2 OTP(Open Telecom Platform) .. 24
2.2.2.1 Supervisor ................................................. 25
2.2.2.2 gen_server ................................................ 25
2.2.2.3 gen_fsm .................................................... 25
2.2.3 Mnesia .......................................................... 26
2.3 USSD ............................................................. 26
    2.3.1 Introduction ............................................... 26
    2.3.1.1 Why USSD? ............................................. 26
    2.3.1.2 Areas of Use ........................................... 27
    2.3.1.3 USSD vs. SMS ........................................ 27
    2.3.2 Architecture & Communication ......................... 28
    2.3.2.1 MS initiated ......................................... 29
    2.3.2.2 NW initiated ......................................... 32
3 Requirements .................................................. 37
    3.1 Functional Requirements ................................... 37
        3.1.1 Mobile Initiated USSD ................................. 37
        3.1.2 Network Initiated USSD .............................. 38
    3.2 Non-functional Requirements .............................. 39
4 Design ......................................................... 40
    4.1 System Overview ........................................... 40
    4.2 Network Design ............................................ 40
    4.3 Software Design .......................................... 42
        4.3.1 Available Architecture ............................... 42
            4.3.1.1 Components ..................................... 42
                4.3.1.1.1 TCP server ................................ 42
                4.3.1.1.2 MUS ......................................... 43
                4.3.1.1.3 HLR .......................................... 43
                4.3.1.1.4 VLR .......................................... 43
                4.3.1.1.5 Application Platform ......................... 43
                4.3.1.1.6 Media Gateway Host .......................... 43
            4.3.1.2 Communication ..................................... 44
            4.3.2 Architecture with USSD support .................. 44
        4.4 Decomposition Description ............................... 45
            4.4.1 USSD_MSC Component ................................ 45
                4.4.1.1 Overview ....................................... 45
                4.4.1.2 Processes ....................................... 45
            4.4.2 USSD_VLR Component ................................ 47
                4.4.2.1 Overview ....................................... 47
                4.4.2.2 Processes ....................................... 48
        4.5 Communication Design ................................... 49
4.5.1 MUS Controller - USSD_MSC

4.5.2 Indirect Communication with USSDGW

4.5.2.1 USSD_MSC to/from USSD_VLR

USSD_VLR to/from USSD_HLR

USSD_HLR to/from USSDGW

4.5.3 Direct Communication with USSDGW

4.5.4 Messages

4.5.4.1 BSSAP

4.5.4.1.1 BSSMAP

4.5.4.1.2 DTAP

4.5.4.1.2.1 Supplementary services

4.5.4.1.2.2 The ones already implemented

4.5.4.2 MAP

4.5.4.2.1 MAP Overview

4.5.4.2.2 Common MAP services

4.5.4.2.3 USSD MAP services

4.5.4.2.4 MAP paging and search services

4.6 Design Rationale

5 Implementation & Test

5.1 Plan

5.1.1 Initial Implementation & Test Plan

5.1.2 Problems Regarding the Initial Plan

5.1.3 The Solutions

5.2 Process

6 Results

6.1 Technical

6.2 Software Development Method

6.3 Company

6.4 Research

6.5 Personal

7 Conclusions

8 Future Work
Abbreviations

(=) Corresponding field carries the same field in incoming request

3GPP The 3rd Generation Partnership Project
BS Base Station
BSC Base Station Controller
BSS Base Station Subsystem
BSSAP Base Station System Application Part
BSSMAP Base Station System Management Application sub-Part
BTS Base Transceiver Station
C Conditional
DETS Disk-Based Erlang Term Storage
DTAP Direct Transfer Application Part
ETS Erlang Term Storage
G-MSC Gateway Mobile service Switching Centre
GSM Global System for Mobile Communication
HLR Home Location Register
HPLMN Home Public Land Mobile Network
IMSI International Mobile Subscriber Identity
M Mandatory
MAP Mobile Application Part
MGC  Media Gateway Controller
MG Host, M-MGW  Media Gateway Host, Mobile Media Gateway
MMI  Man-Machine Interface
MS  Mobile Station
MSC  Mobile service Switching Centre
MSISDN  Mobile Subscriber Integrated Services Digital Network
MUS  Mobile User Service
MUS Con  Mobile User Service Controller
O  Optional
PLMN  Public Land Mobile Network
SS7  Signalling System 7
SCCP  Signalling Connection Control Part
SIM  Subscriber Identity Module
SSN  Subsystem Number
TCAP  Transaction Capabilities Application Part
TMSI  Temporary Mobile Subscriber Identity
USSD  Unstructured Supplementary Service Data
USSD-GW  USSD Gateway
VLR  Visitor Location Register
VPLMN  Visiting Public Land Mobile Network
WAP  Wireless Application Protocol
Chapter 1

Introduction

1.1 Motivation

As Global System for Mobile Communications (GSM) companies have been increasing by each day, the rivalry of making more and more profit among the companies all over the world urges them to be creative and innovative in terms of the services provided by the network and related equipment. Voice Call, Short Message Service (SMS), Unstructured Supplementary Service Data (USSD), Voice Mail, Wireless Application Protocol (WAP) etc. were the services started to be used respectively. One of the services stated above is knowingly or unknowingly used by virtually every mobile owner by entering a code starting with asterisk (∗) and hash (#) on a mobile phone, such as for the purpose of checking balance in prepaid cell card or weather forecast. The service stated above is called as Unstructured Supplementary Service Data (USSD), which is technically a soft real time communication protocol which enables mobile stations to send/receive text messages to/from application in GSM network entities. To mention receiving/sending text message might cause a confusion with SMS. To eliminate the confusion, it can be said that the difference between USSD and SMS is very similar to the difference between chat and email.

Mobil Arts implemented an USSD Gateway which is able to handle big number of USSD requests and sold it to several cell companies. However, the fact that the company did not have a complete test environment of GSM network and did not have any experience in access network side. Therefore, they could not test it or show a demo of it by using real Mobile Station and real Base Station. The company was worried about that there might be some critical bugs or problems when the customer is operating the system and also wants
to show a demo of the system to possible customers. As a result, the company decided to know what kind of solution can be produced to overcome this problem. The solution to this problem is the main motivation of this project.

Also, that the project will be revealed as open source motivates the author because of that the end product of this project might serve software development community and the academics all around the world in terms of learning the development of GSM networks by reading the code base, adding new features and testing without making big efforts.

1.2 Purpose & Tasks

The purpose of this thesis is to be able to deal with large code bases, to design & implement and test an available, fault-tolerant and distributable telecommunication system, to cope with integration problems and to investigate do’s and don’ts while developing such a system.

Tasks of the thesis stated below:

- Analyze the code base of ”Basic Call Server” inherited from the course Project CS run at the department of Information Technology in Uppsala University in 2011
- Prepare Software Requirement Specification Report according to IEEE standards
- Prepare Software Design Description Report according to IEEE standards
- Implement highly available, fault tolerant and distributable USSD support module of MSC/VLR in Erlang according to 3GPP standards and integrate it with the company’s USSD Gateway
- Prepare unit, module, integration and acceptance test
- Collaborate with the thesis project on the support of USSD in HLR

1.3 Methodology

This thesis work was planned according to standard software engineering phases. That is, initial investigation, requirement specification, design, im-
plementation, test phases were followed in the form of V model, which is an extended version of so called waterfall model. While waterfall model follows the each development phase strictly, V model gives opportunity the verification of each phase by mapping the corresponding tests as shown below. Thus, when requirement specifications are identified, acceptance test can be acquired.

Figure 1.1: an example of V-Model[26]

In analysis part, related works, a lot of 3GPP documents about USSD, available code base were intensively read during first weeks but they were
often revisited along each development phase. All other phases are examined as separated chapters.

1.4 Related Works

Research and patenting results\cite{23, 18, 14, 27} mostly focused on scheduling, routing of USSD message, location services based on USSD. In Industry side, these focused on the services provided by USSD, such as banking applications, chat applications or development of the common gateway to handle USSD request as the USSD Gateway which the company provided for this thesis. The main reason behind results is because of the fact that 3GPP and ETSI propose how USSD should be implemented. Therefore, it is preferred to explore innovative services or ways of increasing quality of services.

1.5 Overview of the thesis

Chapter 1: Introduction
In this chapter, motivation, purpose & tasks, methodology of the thesis and related works are introduced.

Chapter 2: Background
In this chapter, GSM concepts, Erlang, USSD are introduced.

Chapter 3: Requirements
In this chapter, requirements specifications are defined in detail.

Chapter 4: Design
In this chapter, design decisions and the reasons behind these decisions are defined in detail.

Chapter 5: Implementation & Test
In this chapter, how modules are implemented and tested is explained.

Chapter 6: Results
In this chapter, results are presented in different contexts.

Chapter 7: Conclusions
In this chapter, the project is concluded under the light of results.

Chapter 8: Future Work
In this chapter, how the prototype of thesis project can be expanded more is explained.
Chapter 2

Background

2.1 Basics of GSM

2.1.1 Introduction to GSM

The history shows us that most of the innovations come up with the defense industry or military requirements. If the technology becomes very common or not need to keep it secret, it is time to reveal for commercializing. This tradition continued with cordless phones, too. It is claimed that the first mobile phone call was made in US in the year 1946.[16] Then, several countries invested in this technology. Involvements and deployments of several countries resulted in different technologies which are not compatible with each other. These studies are generally evaluated as first generation(1g) analog cellular networks.

The acronym GSM standing for Groupe Spéciale Mobile was stated by a committe Européenne des Postes et Télécommunications (CEPT), the European standardization organization which chose the analog technology for communication. Later, CEPT evolved to a new organization named as the European Telecommunications Standard Institute (ETSI). ETSI succeeded to change the national and incompatible analog networks with international digital network standards. The new standard was anymore called Global System For Mobile Communications(GSM).[20]

2.1.2 GSM Networks

GSM networks are big probably the biggest wireless networks in the world. Despite its popularity, it is not that much known. To understand the basic working principles of GSM Networks, GSM Architecture and GSM Ge-
2.1.2.1 GSM Architecture

Any network needs to have a structure to route requests/responses to specific gateways and then corresponding entity. The main challenge in GSM networks is that MS is mobile. Therefore, the architecture should support to monitor the location of MS as it moves through. How this architecture handles such a property is realized in subsequent section.

2.1.2.2 Geographical Network Structure

2.1.2.2.1 Cell The geographical area to be supported with radio service is partitioned into cells. A variety of frequency ranges are assigned to each cell which is the coverage area of an BS. Same frequency ranges are not used by neighbor cells in order to prevent an interference.
2.1.2.2 Location Area  Location area consists of a group of cells. In the network which a MS is currently attached to, the MS’s location is identified by a location area code and is stored in VLR. If the MSC moves to another cell which is located in different location area, the network must be informed to update current location area. If the MS moves to any cell which is located in same location area, no changes is are required. When there is a call to the MS, all cells in the location area in which MS is attached are paged.

2.1.2.3 MSC Area  A MSC is responsible for several location areas. In order to route a call to corresponding MSC, the MS’s MSC area code is stored in HLR and monitored.
2.1.2.2.4 PLMN Area  
Public Land Mobile Network Area represents all the radio coverage area of cells owned by an GSM network operator, such as Turkcell, Telia, Vodafone. In a country, there can be a number of PLMNs. While an MS attached to the provider’s network is assumed to be in Home PLMN (HPLMN), an MS attached to the network which a different network operator is in charge of is assumed to be in Visited PLMN (VPLMN).

2.1.2.3 Components

A typical GSM network consists of Base Station System and Switching System.

2.1.2.3.1 Switching System
SS is mainly responsible for handling calls and other type of services such as USSD, SMS and subscriber related operations such as storing subscriber information. It mainly consists of the entities stated below.

2.1.2.3.1.1 MSC stands for Mobile Services Switching Center which is mainly responsible for

- route, setup, control, handover calls into the current network or the outside network.
- updating subscriber’s current location in VLR & HLR
- checking IMEI
- security issues like authentication, encryption, allocation of TMSI
- accounting and charging
- handling the services(SMS,USSD etc.) other than voice calls

2.1.2.3.1.2 HLR stands for Home Location Register, which is a large database keeping record of mostly permanent(such as, IMSI, MSISDN, subscriber name, surname and address etc.) and confidential information of subscribers(such as, encryption/decryption keys are used to establish secure path between MS and HLR, thus this key is never revealed out of SIM card and HLR) which are independent of geographical area. Also, each subscriber is assigned to a HLR acting as reference point to current user’s location(VLR). Thus, some non-confidential information is replicated to VLR and this result in reducing the load on HLR.
2.1.2.3.1.3 **VLR** stands for Visitor Location Register which is a dynamic subscriber database which is specific to geographical network area and it is generally included in MSC. As stated in HLR, VLR’s main function is to reduce the load on HLR.

2.1.2.3.1.4 **EIR** stands for Equipment Identity Register in which the Network operator keeps the record of IMEI which is a unique number assigned to each mobile phone in EIR. The reason of this is the fact that mobile phones may run with any valid SIM card can cause some problem for stolen phones. In case of stolen, Network operators may block the mobile phone.

2.1.2.3.1.5 **G-MSC** represents Gateway MSC which is the MSC having the interface to external networks. The MSCs which do not have gateway feature forward calls into external networks through G-MSC. Also, another task of G-MSC in case of mobile terminated call is that G-MSC checks HLR in order to identify the MSC which called MS is currently attached.

2.1.2.3.2 **Base Station System**

![Figure 2.5: Typical Base Station System][19, p. 28]

BSS is an middle-man in order to coordinate connection between MS and MSC. It mainly consists of the entities stated below.

2.1.2.3.2.1 **BTS** stands for Base Transceiver Station which is a hardware providing physical connection of mobile station to network.

2.1.2.3.2.2 **BSC** stands for Base Station Controller which is mainly capable of managing several BTSs. It handles radio related messages instead of handling in MSC. Thus, the load on MSC is reduced.

2.1.2.3.2.3 **TRAU** stands for Transcoding Rate and Adaptation Unit which is responsible of de/compressing voice data between mobile station and TRAU. However, it is not used in data communications.
2.2 Erlang

2.2.1 General consideration

Erlang is a functional programming language which was mainly created for telecommunication applications.\[12\] Also, there are other reasons\[15\] p. 1] why Erlang is used in the thesis project;

- As the number of CPU is increased in the system, the code become faster and faster without considering which code is to run on which core.
- Fault-tolerant applications can be implemented in Erlang and code can be modified without stopping whole system.
- Telecommunication system can be built by having less code.
- It is very easy to distribute an application with much less effort.
- A typical Erlang application may have thousands of concurrent processes but the programmer does not need to consider mutexs and locks.

2.2.2 OTP(Open Telecom Platform)

There are many features in Erlang like linking, monitoring processes, time-outs for processes, register them with a name etc. which are given in somehow low level. Also, by the time it was realized that concurrent, distributed and fault-tolerant applications have similarities. Therefore, an experienced Erlang programmer easily realizes that a collection of Erlang libraries is required. All these are collected under the umbrella of OTP(Open Teleom Platform).

A main concept in OTP is the supervision tree. It models the structure of available processes in the system and consists of workers and supervisors. This structure consists of similar modules carrying common properties. Behaviors are the formalization of the most used solutions for the most known problems.\[12\] The standard Erlang/OTP behaviors are: supervisor, gen_server, gen_fsm, which are to be used in the present project and discussed here.
2.2.2.1 Supervisor

In classic programming languages, when a process or an application crash, there are not generally many options to recover the system and most basic thing to do is just wait for the programmer to restart it again. This is very unwanted issue if the ultimate aim is to have available and fault-tolerant systems. Erlang processes might die out of the programmer’s control or intention as well. The solution that OTP brings out is to create a process or application under the supervision of another process. Thus, if something happens to the child process, the supervisor process is informed about the status. Then, the supervisor process may restart that the child process or prevent the crash to spread over other processes or might do something else. Attitude of the supervisor depends on the programmer’s strategy when something is gone wrong. Also, a supervisor may supervise another supervisor.

![Figure 2.6: A typical Erlang supervision tree](image)

2.2.2.2 gen_server

gen_server is one of the most used Erlang behaviour in the need of a worker to be in the concept of client/server.

2.2.2.3 gen_fsm

gen_fsm is one of the most used Erlang behaviour in the need of a worker to be in the concept of finite state machine.
2.2.3 Mnesia

Mnesia is a distributed key/value database management system which is generally used in telecommunication applications requiring:

- Fast real-time key/value lookup
- Complicated non real-time queries mainly for operation and maintenance
- Distributed data due to distributed applications
- High fault-tolerance
- Dynamic re-configuration
- Complex objects

2.3 USSD

2.3.1 Introduction

GSM networks were invented for the purpose of providing basic call service. However, as GSM technology has evolved, new ways to communicate have been invented. Initially, SMS was presented for text message communication among subscribers and then USSD was born in order to fulfill session-based real time data communication needs for supplementary services. Although USSD is used so often in daily life, it is not known that the service which is utilized while checking the prepaid card’s balance by entering *100# is USSD.

2.3.1.1 Why USSD?

The advantages of USSD[24, p. 3] are listed below:

**Cost Efficient** it is not much expensive to support GSM networks with USSD because it uses existing network’s protocols.

**Fast and responsive** The real-time capability of USSD enables the operator to provide fast and responsive services.

**Interactive** Session-based property of USSD provide the operator to create interactive applications, such as chat, mobile banking, WAP.

**Reduced Marketing cost** a variety of USSD applications can be created and integrated easily because the protocol is not complex.
2.3.1.2 Areas of Use

There are very wide areas of use of USSD. They can be listed under three categories as stated below:

**MS initiated**
- Information check applications, such as news, weather, horoscope, balance check
- Bank applications, such as money transfer, account check
- Mobile ticket application, such as buying train and concert tickets

**NW initiated**
- Advertisement applications, such as the operator can advertise a product on the name of a company
- Subscriber information applications, such as subscriber is informed after each call in order to indicate remaining balance

**Both of them**
- Subscription based applications, such as the subscriber may subscribe a sport event and it is informed if any status change in the event occurs
- Instant messaging applications, the subscriber may login/logout a chat application and send/receive messages

2.3.1.3 USSD vs. SMS

Although SMS and USSD are contextually different, they share some features. Given table below [24 p. 3-4] lists the common and unique features of both of them.
<table>
<thead>
<tr>
<th>Features</th>
<th>USSD</th>
<th>SMS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Out of band signalling channels</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Communication characteristic</td>
<td>Real-time, based</td>
<td>Store and Forward</td>
</tr>
<tr>
<td>Communication entities</td>
<td>between MS and an USSD handler in an network entity</td>
<td>among Mobile Originated MS, MSC, Mobile Terminated MS</td>
</tr>
<tr>
<td>Communication protocol</td>
<td>SS7</td>
<td>SS7</td>
</tr>
<tr>
<td>Payload length</td>
<td>182 characters</td>
<td>160 characters</td>
</tr>
<tr>
<td>Message storage in mobile</td>
<td>just one-time cached to see message</td>
<td>stored in either sim card or MS memory</td>
</tr>
<tr>
<td>Analogy</td>
<td>Chat</td>
<td>E-mail</td>
</tr>
<tr>
<td>Operation costs</td>
<td>less costly because of that the USSD communication happens between MS and an handler</td>
<td>much costly because of that SMS messages might need to be passed through different networks</td>
</tr>
</tbody>
</table>

### 2.3.2 Architecture & Communication

USSD is a soft real time data communication protocol between MS and GSM network. While the data communication between MS and SS is fulfilled with BSSAP messages, MAP messages is used among MSC, VLR, HLR and USSD Gateway. The data communication is transparent to MS and BSS. Namely, USSD messages are not modified while passing through these entities. The communication can be established during call or out of call because the communication channel used for both is different. What to do about USSD messages is taken into consideration by mostly checking MMI strings in case of that MS initiates USSD while the messages pass through MSC, VLR, HLR and USSD Gateway. If an network or an application initiates USSD, the communication is handled by corresponding network entity or application. How USSD communication is handled is shown below.
2.3.2.1 MS initiated

MS initiates USSD communication by entering Man Machine Interface (MMI) strings, which contain service code and service information related to desired service. After connection is established between MS and core network, MMI string is encoded as binary in Process Unstructured Supplementary Service Request (PUSSR). PUSSR is decoded in MSC and decided what to do by checking service code in MMI. If MSC has any USSD application which is responsible for handling the request, service information is forwarded to corresponding application. Otherwise, the message is forwarded to VLR. Same decision and handling mechanism is applied to incoming messages into VLR, HLR. Subsequent messages might be Unstructured Supplementary Service Request (USSR) which asks MS to provide information or Unstructured Supplementary Service Notify (USSN) which notifies a message and does not require any response from MS. However, the last message must be always PUSSR in order to indicate that the communication is to release. How MS initiated USSD messages are handled is shown below in different conditions.
Figure 2.8: Mobile Initiated-Single operation

If the network is unable to process the request received from the MS, it shall clear the transaction by sending a RELEASE COMPLETE message containing a return error component. Error values are specified in 3GPP TS 24.080. [facilityNotSupported, unexpectedDataValue]

1. Network only supports protocol version 1 of unstructured supplementary service data operations
2. Mobile station only supports protocol version 1 of unstructured supplementary service data operations

Figure 2.8: Mobile Initiated-Single operation
Figure 2.9: Mobile Initiated-Multiple Operations
2.3.2.2 NW initiated

Any network entity is able to start a USSD communication with an MS. Network initiated communication is started manually by an application under the control of network operator or a kind of event catcher. Before communication is established between MS and corresponding network entity, MS is paged. If got any response from paging, the connection is established. The application might send/receive USSR or USSN messages. However, the last message that the application send is a release message in order to indicate that the communication is released. Network initiated requests by any network entities are just forwarded to the next entity, in which a special examination is not applied. How NW initiated USSD messages are handled is shown below in different conditions.
Figure 2.10: Network Initiated-Single Operation
Figure 2.11: Network Initiated-Multiple Operations
Figure 2.12: Network Initiated Notification-Single Operation
Figure 2.13: Network Initiated Notification-Multiple Operations
Chapter 3

Requirements

Requirements are critically important to specify a road map towards verification and validation of the project. In this chapter, both functional requirements and non-functional requirements are stated with the help of Mobile Arts and 3GPP documents.

3.1 Functional Requirements

In this phase, several meetings with Mobile Arts were handled in order to specify requirements. These specified requirements were confirmed by the company. Complete description of the behaviour of system to be developed is shown below.

3.1.1 Mobile Initiated USSD

An MS will be able to start USSD operation at any time by sending process USSD request. All USSD requests and responses will contain USSD string, an alphabet indicator and language indicator.

Handling at MSC

- MSC decides whether it forwards the message to VLR or USSD Gateway, or it evaluates in local according to destination IMSI or MSISDN or USSD service code.

- MSC will set up MAP transaction to HLR or USSD Gateway when obtained USSD operation having HPLMN service code and forwards the operation to HLR.
• Operation will be forwarded to the application in MSC if HPLMN service code is not included

• MSC will release the other if the transaction between MS and MSC or MSC and VLR is released

• MSC not supporting the alphabet specified in USSD operation will set up MAP transaction to VLR and forward the operation as it is

Handling at VLR

• VLR will set up MAP transaction to HLR or USSD Gateway when obtained USSD operation having HPLMN service code and the user is not in the HPLMN, and forwards the operation to HLR.

• Operation will be forwarded to the application in VLR if HPLMN service code is not included or the user is in the HPLMN

• VLR will release the other if the transaction between MSC and VLR or VLR and HLR/USSD Gateway is released

• VLR not supporting the alphabet specified in USSD operation will set up MAP transaction to HLR or USSD Gateway and forward the operation as it is.

3.1.2 Network Initiated USSD

Any network entity (MSC, VLR) will be able to start USSD operation at any time by sending either USSD request (waiting for response from MS) or USSD notification (not requiring any response from MS). All USSD requests, notifications and responses will contain USSD string, an alphabet indicator and language indicator.

Invoking from VLR

• An Application in VLR starting any USSD operation will be able to set up MAP transaction to MSC in which MS is available at the moment and forwards the operation to MSC

• VLR will be able to release transaction according to response (such as, due to user clearing) from MSC or before a response (such as, time out in application)

• VLR will use the same transaction until operations are completed, if application in VLR wants to send further operations
• VLR will initially release the first transaction if it wants to create a new transaction

Invoking from MSC

• An application in MSC starting any USSD operation will be able to set up BSSAP transaction to MS in which MS is available at the moment and forwards the operation to MSC
• MSC will be able to release transaction according to response (such as, due to user clearing) from MS or before a response (such as, time out in application)
• MSC will use the same transaction until operations are completed, if application in MSC wants to send further operations
• VLR will initially release the first transaction if application in MSC wants to create a new transaction

Cross-phase compatibility

• Network initiated USSD will be rejected or will release the connection if MS or any entity which operations pass through are phase 1

3.2 Non-functional Requirements

• The system should be easily maintainable in case of requiring flexible forwarding criteria changes without stopping whole system
• The system should be open to become distributable to handle thousands of USSD requests
• The system should get faster when multi-core CPUs are used
• More than two concurrent USSD dialogs should be supported
• The system should get up automatically in case of that any crash occurs during runtime
• Any local crash should not disseminate to other parts of the system. That is, crashes should be isolated.
Chapter 4
Design

4.1 System Overview

USSD service is based on available architecture designed by Project CS-2011 course students in the framework of Basic Call Service project aiming to fulfill basic communication among mobile stations attached to the system. As seen clearly in the figure shown below, there are some additions and modifications on the available architecture to support USSD service. Other than these, components are used as it is. In order that USSD service is to be added to the available system, encoding/decoding of L3/DTAP messages in MUS Core and MUS, also L3/DTAP to/from real MAP message conversion (rather than Erlang constructed message format created according to initiative of designers of the available architecture) in the communication between MSC and VLR is fulfilled. At the same time, MUS takes proper action in order to manage USSD specific workers. USSD specific workers in MSC, VLR and HLR are designed for both forwarding incoming/outgoing messages and local applications in each entity. Most importantly, USSD Gateway is integrated successfully into the available architecture.

4.2 Network Design

The IP network which actually carries SS7 messages are designed below.
Figure 4.1: SS7 Network
4.3 Software Design

4.3.1 Available Architecture

4.3.1.1 Components

In the figure stated above, it is clearly seen that components having blue color construct the infrastructure of this thesis project. Here are components and some knowledge about them:

4.3.1.1.1 TCP server

TCP server is a gen_tcp maintaining MSC’s connection with BSC. In this
way, An interface messages are obtained from BSC and forwarded to MUS as binary.

4.3.1.1.2 MUS
Mobile User Service is the entry and exit point of all messages from/to BSC. In MUS, MUS Core, which is a gen_server, decodes the incoming binary packets from TCP Server to Erlang based message. Then, a new worker of MUS Controller, which is a gen_gsm, is created and its process id is stored with Source Local Reference, Destination Local Reference, IMSI and TMSI as session key which is to be accessed for the associated session or MUS Core forwards messages to previously created MUS Controller worker by checking session key. Service logic is decided by MUS Controller. That, it checks the message and decided which service to forward the message. For example, if the message is associated with Mobile Originated Call, it forwards message to Mobile Originated worker or the message associated with Mobile Terminated Call are forwarded to Mobile Originated Worker. The most important feature of MUS controller is that it can be easily extended in case of requiring a new service, such as SMS, USSD.

4.3.1.1.3 HLR
Current HLR is a simple Erlang module which contains hard-coded subscriber information as white list which maps IMSI with MSISDN number.

4.3.1.1.4 VLR
VLR is a component consisting of DETS table in the backend and a gen_server in the front-end. In DETS table, while IMSI and TMSI are stored as key identifier, location update related and call related information constructs the rest of key. All the information in DETS is stored in temporary manner. For example, if MS is turned off, associated information in DETS table are cleaned up.

4.3.1.1.5 Application Platform
It is a specific platform which is mainly designed for call related applications, such as Voice Mail System. However, it also contains some module which seizing the virtual switches, storing session, routes ISUP messages to outside network and control media gateway hosts.

4.3.1.1.6 Media Gateway Host
It provides the service controlling RTP communication among subscribers.
4.3.1.2 Communication

While all the communication between BSS and SS are real SS7 communication over TCP/IP, the communication in SS is Erlang messages of SS7.

4.3.2 Architecture with USSD support

Architecture with USSD support consists of previous architecture and loosely coupled USSD components clearly seen on the figure:

**USSD_MSC component**

**USSD_VLR component**
4.4 Decomposition Description

4.4.1 USSD_MSC Component

4.4.1.1 Overview

Figure 4.4: USSD_MSC component

4.4.1.2 Processes

Supervisors

**USSD_MSC_sup** This process is a static supervisor which is responsible for starting, stopping and monitoring USSD_MSC process, ms_init_workers_sup process and NW_init_workers_sup process, and keep them alive against any expected or unexpected issues. Its restart strategy is one_for_one and restart mode is permanent. That is, if a child process(USSD_MSC, ms_init_workers_up or NW_init_workers_sup) terminates, only terminated process is restarted. Thus, errors are
isolated from whole system and localized, and they are prevented to disseminate to other processes. However, if USSD_MSC_sup gets a terminate message, that disseminates all processes in manner of from top to bottom and naturally every process getting terminate message terminates itself. Thus, garbage-collection is fulfilled.

**ms_init_workers_sup** This process is a dynamic supervisor which is responsible for starting, stopping and monitoring processes for MS initiated USSD dialogues in MSC, and keep them alive against any expected or unexpected issues. Its restart strategy is simple_one_for_one and restart mode is temporary. That is, it will not start any child processes at the beginning but processes to be supervised are added dynamically by calling a function in it. If a child process(SS_coordinator_MSC or ms_init_USSD_MSC) terminates, terminated process is not restarted and garbage collected successfully. Thus, errors are isolated from whole system and localized, and they are prevented to disseminate to other sibling processes running for different dialogues.

**NW_init_workers_sup** This process is a dynamic supervisor which is responsible for starting, stopping and monitoring processes for network initiated USSD dialogues in MSC, and keep them alive against any expected or unexpected issues. This supervisor has the same characteristics as ms_init_workers_sup.

**Workers**

**USSD_MSC**

This process is core process which is responsible of

1. Keep list of worker processes
2. Coordinate messages and route them according to rules in routing tables.
3. Handle local/remote application request/response
4. Coordinate tcap connections

**SS_coordinator_MSC** This co-ordinator process in the MSC handles a CM connection request with CM service type supplementary service and call the macro called process_access_request_MSC, and create the process called ms_init_USSD_MSC and forwards process_unstructured_SS_request DTAP message to it. Then, it relays every message for MS initiated dialogue.

[9, p. 617]
ms_init_USSD_MSC  This process is used in order to handle Process USSD request message, then subsequent USSD Request or USSD Notify messages coming in MSC. According to some routing table definitions in USSD_MSC, it can route messages to USSD-GW or VLR or USSD application in MSC.  
[9, p. 652]

NW_init_USSR_MSC  This process is used in order to handle USSD request or USSD Notify messages coming in MSC. According to some routing table definitions in USSD_MSC, it can route messages to USSD-GW or VLR or USSD application in MSC.  
[9, p. 667]

4.4.2 USSD_VLR Component

4.4.2.1 Overview

Figure 4.5: USSD_VLR component
4.4.2.2 Processes

Supervisors

**USSD_VLR_sup** This process is a static supervisor which is responsible for starting, stopping and monitoring USSD_VLR process, ms_init_workers_sup process and NW_init_workers_sup process, and keep them alive against any expected or unexpected issues. This supervisor has the same characteristics as USSD_MSC_sup.

**ms_init_workers_sup** This process is a dynamic supervisor which is responsible for starting, stopping and monitoring processes for MS initiated USSD dialogues in VLR, and keep them alive against any expected or unexpected issues. This supervisor has the same characteristics as ms_init_workers_sup in USSD_MSC component.

**NW_init_workers_sup** This process is a dynamic supervisor which is responsible for starting, stopping and monitoring processes for network initiated USSD dialogues in VLR, and keep them alive against any expected or unexpected issues. This supervisor has the same characteristics as ms_init_workers_sup.

Workers

**USSD_VLR**
This is the main coordinator process having the same characteristics as USSD_MSC. For explanation, go to [4.1.2](#).

**ms_init_USSD_VLR** This process is used in order to handle Process USSD request message forwarded from MSC, then subsequent USSD Request or USSD Notify messages coming in VLR. According to some routing table definitions in USSD_MSC, it can route messages to either HLR or USSD application in VLR.
[9] p. 655

**NW_init_USSD_VLR** This process is used in order to handle USSD request or USSD Notify messages coming in VLR. According to some routing table definitions in USSR_VLR, it can route messages to either HLR or USSD application in VLR.
[9] p. 671
4.5 Communication Design

4.5.1 MUS Controller - USSD_MSC

BSSAP messages defined in 4.5.4.1 are exchanged between these entities according to the SDL flow for USSD.

4.5.2 Indirect Communication with USSDGW

4.5.2.1 USSD_MSC to/from USSD_VLR
USSD_VLR to/from USSD_HLR
USSD_HLR to/from USSDGW

MAP messages defined in 4.5.4.2 are exchanged between these entities according to the SDL flow for USSD.

4.5.3 Direct Communication with USSDGW

4.5.3.1 USSD_MSC to/from USSDGW

MAP messages defined in 4.5.4.2 are exchanged between these entities according to the SDL flow for USSD.

4.5.4 Messages

4.5.4.1 BSSAP

BSSAP messages are structured in this format;

```javascript
{
  'request',
  [BSSAPMessageType],
  [SessionInfo(inproplist)],
  [Parms(inproplist)]
}

{
  'response',
  [BSSAPMessageType],
  [SessionInfo(inproplist)],
  [Parms(inproplist)]
}
```
4.5.4.1.1 **BSSMAP** messages are inherited from Project CS. That is, they are already defined.

PAGING,
COMPLETE L3 INFORMATION

- CM SERVICE REQUEST
- PAGING RESPONSE

4.5.4.1.2 **DTAP**

4.5.4.1.2.1 Supplementary services
REGISTER
FACILITY
RELEASE COMPLETE

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Request</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transaction Id</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>Message Type</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>Component Type</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>Operation Code</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>USSD Data Coding Scheme</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>USSD String</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>Problem code</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>Error code</td>
<td>C</td>
<td>C</td>
</tr>
</tbody>
</table>

4.5.4.1.2.2 The ones already implemented
IDENTITY REQUEST
IDENTITY RESPONSE
TMSI REALLOCATION COMMAND
TMSI REALLOCATION COMPLETE
TMSI REALLOCATION FAILURE
RELEASE
RELEASE COMPLETE
CM SERVICE ACCEPT
CM SERVICE REJECT
CM SERVICE ABORT
4.5.4.2 MAP

4.5.4.2.1 MAP Overview
MAP messages are structured in this format;
{
  'MAP',
  [MAPMessageType],
  [request|indication|response|confirm],
  [Parms(inproplist)]
}

4.5.4.2.2 Common MAP services

MAP_OPEN

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Request</th>
<th>Indication</th>
<th>Response</th>
<th>Confirm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application context name</td>
<td>M</td>
<td>M(=)</td>
<td>U</td>
<td>C(=)</td>
</tr>
<tr>
<td>Destination address</td>
<td>M</td>
<td>M(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Destination reference</td>
<td>U</td>
<td>C(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Originating address</td>
<td>U</td>
<td>O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Originating reference</td>
<td>U</td>
<td>C(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specific information</td>
<td>U</td>
<td>C(=)</td>
<td>U</td>
<td>C(=)</td>
</tr>
<tr>
<td>Responding address</td>
<td>U</td>
<td>C(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td>M</td>
<td>M(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Refuse-reason</td>
<td>C</td>
<td>C(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Provider error</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

MAP_CLOSE

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Request</th>
<th>Indication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release method</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>Specific information</td>
<td>U</td>
<td>C(=)</td>
</tr>
</tbody>
</table>

MAP_DELIMITER

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Request</th>
<th>Indication</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

MAP_U_ABORT

51
<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Request</th>
<th>Indication</th>
<th>Response</th>
<th>Confirm</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Reason</td>
<td>M</td>
<td>M(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Diagnostic Information</td>
<td>U</td>
<td>C(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Specific information</td>
<td>U</td>
<td>C(=)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**MAP_P_ABORT**

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Indication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Provider Reason</td>
<td>M</td>
</tr>
<tr>
<td>Source</td>
<td>M</td>
</tr>
</tbody>
</table>

**MAP_NOTICE**

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Indication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem diagnostic</td>
<td>M</td>
</tr>
</tbody>
</table>

4.5.4.2.3 **USSD MAP services**  

**MAP_PROCESS_UNSTRUCTURED_SS_REQUEST**

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Request</th>
<th>Indication</th>
<th>Response</th>
<th>Confirm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke Id</td>
<td>M</td>
<td>M(=)</td>
<td>M(=)</td>
<td>M(=)</td>
</tr>
<tr>
<td>USSD Data Coding Scheme</td>
<td>M</td>
<td>M(=)</td>
<td>C</td>
<td>C(=)</td>
</tr>
<tr>
<td>USSD String</td>
<td>M</td>
<td>M(=)</td>
<td>C</td>
<td>C(=)</td>
</tr>
<tr>
<td>MSISDN</td>
<td>C</td>
<td>C(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User error</td>
<td></td>
<td></td>
<td>C</td>
<td>C(=)</td>
</tr>
<tr>
<td>Provider error</td>
<td></td>
<td></td>
<td></td>
<td>O</td>
</tr>
</tbody>
</table>

**MAP_UNSTRUCTURED_SS_REQUEST**

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Request</th>
<th>Indication</th>
<th>Response</th>
<th>Confirm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke Id</td>
<td>M</td>
<td>M(=)</td>
<td>M(=)</td>
<td>M(=)</td>
</tr>
<tr>
<td>USSD Data Coding Scheme</td>
<td>M</td>
<td>M(=)</td>
<td>C</td>
<td>C(=)</td>
</tr>
<tr>
<td>USSD String</td>
<td>M</td>
<td>M(=)</td>
<td>C</td>
<td>C(=)</td>
</tr>
<tr>
<td>Alerting PAttern</td>
<td>C</td>
<td>C(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User error</td>
<td></td>
<td></td>
<td>C</td>
<td>C(=)</td>
</tr>
<tr>
<td>Provider error</td>
<td></td>
<td></td>
<td></td>
<td>O</td>
</tr>
</tbody>
</table>

**MAP_UNSTRUCTURED_SS_NOTIFY**
### 4.5.4.2.4 MAP paging and search services

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Request</th>
<th>Indication</th>
<th>Response</th>
<th>Confirm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke Id</td>
<td>M</td>
<td>M(=)</td>
<td>M(=)</td>
<td>M(=)</td>
</tr>
<tr>
<td>IMSI</td>
<td>M</td>
<td>M(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stored Location Area Id</td>
<td>M</td>
<td>M(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TMSI</td>
<td>U</td>
<td>C(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>User error</td>
<td>C</td>
<td>C(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Provider error</td>
<td>O</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### MAP_SEARCH_FOR_MS

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Request</th>
<th>Indication</th>
<th>Response</th>
<th>Confirm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke Id</td>
<td>M</td>
<td>M(=)</td>
<td>M(=)</td>
<td>M(=)</td>
</tr>
<tr>
<td>IMSI</td>
<td>M</td>
<td>M(=)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Current Location Area Id</td>
<td></td>
<td></td>
<td>C</td>
<td>C(=)</td>
</tr>
<tr>
<td>User error</td>
<td></td>
<td></td>
<td>C</td>
<td>C(=)</td>
</tr>
<tr>
<td>Provider error</td>
<td></td>
<td></td>
<td></td>
<td>O</td>
</tr>
</tbody>
</table>

### 4.6 Design Rationale

The main idea behind the architecture of USSD system is that each component is free and loosely coupled to each other. Therefore, it works more like in the manner of plug and play. Each component has 4 special properties:

**Open to become scalable**:

Components may work in different nodes with very minor additions in the system. For example, we can obtain it by adding a load balancer to the system and replicable global database.

**Loose coupling**:

USSD system is divided to work in components which are compatible with any third party component.
Concurrency and parallelism:
These are main mechanisms used all over the architecture because architecture is designed to processes work concurrently in form of specified function on the same machine. Also, the fact that the system is appropriate to work on different machines provides parallelism feature. Thus, we obtain a system having increased application throughput, high responsiveness for input/output and more appropriate program structure.

Carefully designed supervision tree:
This is technically known as "Let it crash". Erlang supervisors will be used in tree structure for this purpose. If any process fails, it needs to inform related process about failure or it needs to restart the system appropriately or crash silently. Thus, a high available system is obtained.

Timeout values:
The values are defined in each process, thus it can crash itself or inform other processes. But if it does not get expected message within specified timeout value. Thus, automatic garbage collection mechanism is obtained.
Chapter 5

Implementation & Test

5.1 Plan

5.1.1 Initial Implementation & Test Plan

As implementation strategy, down to top approach were considered. That is, the most basic module in order that the USSD system works in the specified functional requirements were to be implemented one by one and then the whole system was to be obtained. Because of that a similar project was conducted by another student, it was decided to collaborate. Here is the initial implementation plan accepted by the company:

- During week 1-2, the programmer of USSD/MSC implements BSSAP encoder/decoder between BSC and MSC and test a module in order to try to send/receive USSD and related messages to/from BSC while the programmer of USSD/HLR writes encoder/decoder for MAP and test module in order to try to send/receive USSD and related messages to/from USSDGW.

- In week 3-4, the programmer of USSD/MSC implements USSD_MSC and USSD_VLR components while the programmer of USSD/HLR writes USSD_HLR component.

- In week 5-6, test cases are identified and test modules are accordingly written, both module is integrated.

5.1.2 Problems Regarding the Initial Plan

When the implementation was started to progress, several problems affected the software development time and the software development methodology.
Here are the problems:

- Writing BSSAP encoder/decoder for USSD is much more difficult and complex than it looks because of that 3GPP documents had some typos and lacked describing package structure.

- Adapting MUS Core and MUS Controller to USSD messages took more time because of that no documentation is available and the codes is very long to follow.

- When BSSAP encoder/decoder, USSD_MSC and USSD_VLR applications were done, SS7 network had not been set up, no MAP encoder/decoder had not been implemented and installing USSD Gateway was a big problem.

- Finding out the correct application helping USSD_VLR to connect SS7 network

5.1.3 The Solutions

As a solution, the author decided to progress the project by taking these decision:

- It was decided that mock modules should be written in order to test encoder/decoder until the last version of encoder/decoder is obtained instead of writing whole the encoder/decoder directly and test. Thus, it got much more easy to construct other dependent applications. Namely, software development methodology is slightly changed to V-Model.

- The author had to set up SS7 network, implement MAP encoder/decoder.

- The author found out the correct application for SS7 connection and modified it according to the requirements.

As a result, implementation time got almost doubled.

5.2 Process

Because of the nature of V-Model, implementation and test were carried out together. Acceptance test, System test, integration test and module test with mocking were applied but unit test was not considered because of time limitation.
Chapter 6

Results

Results can be given in five parts stating in terms of technical results of the thesis project and the result in the perspective of software development method, benefits of the company, research and the author.

6.1 Technical

The tests results showed that the system is able to handle multiple mobile station at the same time in soft real time context and run very well and robust not only in good scenarios but also it worked very well in virtually every bad scenario. That is, requirements are fulfilled completely.

However, because of the fact that USSD support in HLR which is another thesis work conducted on parallel with thesis has not been completed yet, ussd dialog between HLR and MSC/VLR has not been tested. It is expected to run with HLR as it run with USSD gateway because the dialog is just the same.

Also, IMSI/MSISDN transformation needs in order to send TCAP messages to USSD Gateway. For now, It is not possible to connect to Mobile Arts’ HLR because of that setup & configuration of Mobile Arts’ HLR depending on the other thesis has not been done yet. Therefore, the basic HLR without SS7 support in Basic Call Service had to be used.

It is noticed that Mobile Arts’ USSD Gateway did not support network initiated USSD dialog in application level. That is, it is not possible to define an application for network initiated dialog as done for mobile initiated dialog. Therefore, network initiated dialog from USSD Gateway run on USSD
That the system helps any programmer to write and run a local application in short time without restarting the system was confirmed by colleagues.

Elastic forwarding feature of the system proved that system might be reconfigured without restarting the system.

6.2 Software Development Method

As stated before, this thesis is planned to coordinate with another thesis. While the topic of this thesis is USSD support in VLR, the other one is USSD support in HLR. Because of that both thesis shares lots of stuff, it is decided that MAP dialog will be implemented by the thesis worker of USSD support in HLR and the dialog on A interface will be implemented by the thesis worker of USSD support in HLR. Thus, more time were going to be allocated for testing. All plans were done according to that but it failed and development time was extended. The result is that time for unit test could not be allocated and properly understood that one plan might fail so secondary plan should be done to cover the possible or expected problems. To sum up, plans with the possible failure should be considered.

Expected that setup SS7 network and MA’s USSD gateway are not a big deal and plans were done according to this assumption. However, to do all these took more time than implementation. The result is that extra development time overhead occurred. That is, dependencies or the environment that the system run on affect development time critically. Therefore, before planning initial investigation time should be allocated for healthy plans.

6.3 Company

That the thesis project implementation is used in the real system’s test shows that it is professional and useful for the company.

6.4 Research

This implementation is just an extension to Basic Call Server and Basic Call server provided the author to fulfill the research without developing lots
of basic stuffs. Therefore, this implementation may help other people in Universities in case of adding more features and get results in short time.

6.5 Personal

The thesis work was extremely challenging and required lot of reading of modules, 3GPP documents and SS7 network. Therefore, it was extremely great opportunity for professional development of the author in Telecommunication.
Chapter 7

Conclusions

In this thesis project, lots of 3GPP documents, patents, books, and the code base were read in order to understand USSD and available GSM infrastructure. However, books offered very limited information about USSD. Standard software engineering processes were applied. According to validation and verification phase results, the developed product fulfills virtually all requirements stated before and also the most important goal which is to reach an available, fault-tolerant and distributable telecommunication system was fulfilled. Although tests heavily depended on MS and BTS, most critical tests were able to be applied. Thanks to Erlang’s freely provided features, these goals were obtained easily, within a reasonable time and at a reasonable performance. The thesis project was successfully integrated to given code base. Thus, the system became such a composite product that it can handle both call and USSD requests and also it supports any modular additions. Thesis project was not integrated with USSD module in HLR because of that HLR implementation was someone else’s responsibility and not finished yet.

According to the perspective of obtained results which state both how most goals are reached well and how other goals were not reached as expected, these conclusions are drawn:

- Erlang is very appropriate for developing telecommunication applications.
- Testing a system which is very tightly dependent on hardware is difficult.
- Predicting exact amount of time for the development phases of a system which is very new to a programmer is hard.
• V-model is appropriate for the kind of projects having requirements which are fixed and do not subject to change.

• This thesis project is very important step to become a telecommunication specialist.

• The product is professional because of that it passed acceptance test.
Chapter 8

Future Work

Although learning curve of the implementation is pretty difficult because of that most part of the implementation lacks comments, the system is very sufficient to add more features. For example, adding SMS support is so easy.

As stated before, no unit testing time was allocated because of the problems stated in the parts of Technical and Software Development Method in Results. Therefore, the system might be tested more in order to discover any bugs.

Actually, USSD can be used by other services. For example, when a phone call gets over, an USSD message can be sent to the subscriber about the available balance after the call. Also, it can be used together with voice mail server in order to notify the subscriber about a new voice mail.

Instead of configuring the system by using some function calls, this can be done through a web interface.

For more realistic scenarios, charging and accounting feature might be added to system.

USSD gateway lacks the feature of creating network initiated programs. Therefore, this feature can be added to USSD Gateway.

The system is very open to become scalable because of the nature of Erlang. Therefore, it might handle dialogs as distributed by making some minor modifications.

Security issues were not considered in the implementation. Therefore, some
security mechanisms can be added to the system.
Bibliography


65


