IT 10 042
Examensarbete 30 hp
September 2010

IP Multimedia Subsystem (IMS) Test Environment Simulator

Hao Zhang

Institutionen för informationsteknologi
Department of Information Technology

UPPSALA
UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

IP Multimedia Subsystem (IMS) Test Environment Simulator

Hao Zhang

The IP Multimedia Subsystem (IMS) is the key element in the 3G mobile network architecture. IMS makes it possible to provide subscribers with ubiquitous cellular access to all the Internet provided services. This thesis report fully describes the project which aims on investigation and implementation of a test environment simulating key functional entities within the IMS core network. The achieved test environment simulator is intended to be used to facilitate development and test of IMS based systems without requiring access to a live IMS network.

This report starts with an overview of IMS concepts and system requirements. Then it will give thorough description on system design and implementation. Several major communication protocols in IMS core network, such as SIP, RTP and Diameter, are implemented. The main IMS network elements, CSCF and HSS, are simulated. In addition, a handset simulator that is capable of depositing and retrieving voice mail is also implemented. Tests are conducted between completed IMS Test Environment Simulator and external IMS Voice Mail System by performing signaling and media communication in between. Finally, the report discusses potential future work based on the accomplished system prototype and summarizes achievements as well as challenges of the project.

Handledare: Martin Kjellin
Ämnesgranskare: Sven-Olof Nyström
Examinator: Anders Jansson
IT 10 042
Sponsor: Mobile Arts AB

Tryckt av: Reprocentralen ITC

"Computers are incredibly fast, accurate and stupid. Human beings are incredibly slow, inaccurate and brilliant. Together they are powerful beyond imagination."

Albert Einstein

Contents

List of Figures

List of Tables

Abbreviations

3G	**T**hird **G**eneration mobile telephony technology
3GPP	**T**hird **G**eneration **P**atnership **P**roject
AAA	**A**uthentication **A**uthorization **A**ccounting
ADSL	**A**symmetric **D**igital **S**ubscriber **L**ine
AS	**A**pplication **S**erver
AUC	**AU**thentication **C**entre
AVP	**A**ttribute **V**alue **P**air
BGCF	**B**reakout **G**ateway **C**ontrol **F**unction
CGI	**C**ommon **G**ateway **I**nterface
CSCF	**C**all **S**ession **C**ontrol **F**unction
DBMS	**D**ata**B**ase **M**anagement **S**ystem
DNS	**D**omain **N**ame **S**ystem
DTMF	**D**ual-**T**one **M**ulti-**F**requency signaling
ENUM	telephon**E** **NU**mber **M**apping
FSM	**F**inite **S**tate **M**achine
GPRS	**G**eneral **P**acket **R**adio **S**ervice
GSM	**G**lobal **S**ystem for **M**obile communication
GUI	**G**raphic **U**ser **I**nterface
HLR	**H**ome **L**ocation **R**egister
HSS	**H**ome **S**ubscriber **S**erver
HTTP	**H**yper**T**ext **T**ransfer **P**rotocol
I-CSCF	**I**nterrogating **C**all **S**ession **C**ontrol **F**unction
IETF	**I**nternet **E**ngineering **T**ask **F**orce
IMS	**I**P **M**ultimedia **S**ubsystem
IM-SSF	**I**P **M**ultimedia **S**ervice **S**witching **F**unction

IP	**I**nternet **P**rotocol
ITU	**I**nternational **T**elecommunication **U**nion
MEGACO	**ME**dia **GA**teway **CO**ntrol
MGCF	**M**edia **G**ateway **C**ontroller **F**unction
MGW	**M**edia **G**ate**W**ay
MRF	**M**edia **R**esource **F**unction
MRFC	**M**edia **R**esource **F**unction **C**ontroller
MRFP	**M**edia **R**esource **F**unction **P**rocessor
NNI	**N**etwork to **N**etwork **I**nterface
OAM	**O**peration **A**dministration and **M**aintenance
OSA-SCS	**O**pen **S**ervice **A**cess-**S**ervice **C**apability **S**erver
OTP	**O**pen **T**elecom **P**latform
P-CSCF	**P**roxy **C**all **S**ession **C**ontrol **F**unction
PDA	**P**ersonal **D**igital **A**ssistant
PSTN	**P**ublic **S**witched **T**elephone **N**etwork
QoS	**Q**uality **o**f **S**ervice
RADIUS	**R**emote **A**uthentication **D**ial **I**n **U**ser **S**ervice
RTP	**R**eal-time **T**ransport **P**rotocol
RTCP	**RT**P **C**ontrol **P**rotocol
S-CSCF	**S**erving **C**all **S**ession **C**ontrol **F**unction
SCTP	**S**tream **C**ontrol **T**ransmission **P**rotocol
SDP	**S**ession **D**escription **P**rotocol
SGW	**S**ignaling **G**ate**W**ay
SIP	**S**ession **I**nitiation **P**rotocol
SLF	**S**ubscriber **L**ocation **F**unction
SVN	**S**ub**V**ersio**N**
TCP	**T**ransmission **C**ontrol **P**rotocol
TLS	**T**ransport **L**ayer **S**ecurity
UAC	**U**ser **A**gent **C**lient
UAS	**U**ser **A**gent **S**erver
UDP	**U**ser **D**atagram **P**rotocol
UE	**U**ser **E**quipment
UMTS	**U**niversal **M**obile **T**elecommunications **S**ystem

UPSF	**U**ser **P**rofile **S**erver **F**unction
VMS	**V**oice **M**ail **S**ystem
VoIP	**V**oice **o**ver **IP**
WLAN	**W**ireless **L**ocal **A**rea **N**etwork

Dedicated to my beloved wife Xiaoyi Liu and lovely son TaoTao

Chapter 1

Introduction

1.1 Project Overview

This thesis is initiated by Mobile Arts AB with the purpose of investigating and developing a testing environment for IMS (IP Multimedia Subsystem) [1, 2] based systems. Mobile Arts AB is a Swedish telecommunication software company which provides innovative solutions to mobile operators in the areas of messaging, location and advertisement in GSM and UMTS networks.

The goals of this thesis project have been to:

1. Develop a test environment for IP Multimedia Subsystem (IMS) based systems that support Session Initiation Protocol (SIP) [3, 4], Real-time Transport Protocol (RTP) [5] and Diameter protocol [6];
2. Provide a test environment for a concurrent thesis with respect to an IMS Voice Mail System (VMS) [7].

Nowadays, Third Generation (3G) networks are getting more and more popular. They aim to merge two of the most successful paradigms in communications: cellular networks and the Internet. The IP Multimedia Subsystem (IMS) [8] is the key element in the 3G architecture that makes it possible to provide ubiquitous cellular access to all the services that the Internet provides. Thus IMS has become quite promising in providing subscribers with substantial multimedia services.

As a telecommunication service solution provider company, Mobile Arts AB has great interest to implement IMS based application servers to meet mobile operators future needs. This is the motivation and reason that Mobile Arts AB initiates this thesis

project. However, it is absolutely not cost-effective to pay a lot to access a live IMS core network just in order to test its IMS based servers functionality. Thus, an IMS test environment simulator is really necessary. Hopefully the prototype achieved in this thesis project can bring benefits and inspirations

The project is implemented based on Erlang/OTP (http://www.erlang.org/) with an x86 PC running Ubuntu Linux distribution. It has been done mainly at Mobile Arts AB premises in Stockholm, Sweden. This report is written using the LaTeX typesetting software.

I would like to thank my supervisor Martin Kjellin at Mobile Arts AB who guided me all the time during the project, my examiner Sven-Olof Nystrom at the Computer Science Department of Uppsala University who gave a lot of constructive comments and suggestions to my thesis report, as well as all engineers at Mobile Arts AB who helped me a lot during my thesis work.

1.2 Project Background: IP Multimedia Subsystem (IMS)

In this section a brief introduction to the IP Multimedia Subsystem (IMS) [1, 2] is given, as an internationally standardized network architecture describing how telecommunication operators can provide multimedia services to subscribers.

1.2.1 IMS Requirements

Third Generation (3G) networks aim to merge two of the most successful paradigm in communications: cellular networks and the Internet [9]. The IP Multimedia Subsystem is the key element in the 3G architecture that makes it possible to provide ubiquitous cellular access to all the services that the Internet provides. This is the future vision of IMS.

Since the birth of the telephone, the telecommunication industry has used circuit-switched technology. However, the current trend is to substitute it with more efficient packet-switched technology. 3G networks have a packet-switched domain. The packet-switched domain provides IP access to the Internet, so that a user can take advantage of all the services that service providers on the Internet offer, such as voice mail, conferencing service and VoIP.

So why do we need the IMS, if all the power of the Internet is already available for 3G users through the packet-switched domain? The answer is threefold: QoS (Quality of Service), charging, and integration of different services.

The main issue with the packet-switched domain to provide real-time multimedia services is that it provides a best-effort service without QoS; that is, the network offers no guarantees about the amount of bandwidth a user gets for a particular connection or about the delay the packets experience. So, one of the reasons for creating the IMS was to provide the QoS required for enjoying, rather than suffering, real-time multimedia sessions. The IMS takes care of synchronizing session establishment with QoS provision so that users have a predictable experience. In summary, the three main reasons to create the IMS are:

1. To provide reasonable QoS (Quality of Service) which means to guarantee the amount of bandwidth a user gets for a particular connection and the delay of packets in order to offer enjoyable real-time multimedia sessions experience to end-users.

2. To be able to charge multimedia sessions appropriately. The IMS provides information about the service being invoked by the user, and with this information the operator decides whether to use a flat rate for the service, apply traditional time-based charging, apply QoS-based, or perform any new type of charging.

3. To provide integrated services to users. The IMS defines the standard interfaces to be used by service developers. This way, operators can take advantage of a powerful multi-vendor service creation industry, avoiding sticking to a single vendor to obtain new services. Furthermore, the aim of the IMS is not only to provide new services but to provide all the services, current and future, that the Internet provides. To achieve these goals the IMS uses Internet technologies and Internet protocols. Moreover, the interfaces for service developers we mentioned above are also based on Internet protocols.

As all above, the IMS is created to provide all Internet services with a reasonable QoS at an acceptable price to users. It truly merges the Internet with the cellular world by using cellular technologies to provide ubiquitous access and Internet technologies to provide appealing services.

1.2.2 Overview of IMS Architecture

I have drawn Figure 1.1 to depict an overview of the IMS architecture [10, 11]. The figure shows most of the signaling interfaces in the IMS, typically referred to by a two or three letter code. On the left side of Figure 1.1 we can see the IMS mobile terminal, typically referred to as the User Equipment (UE). The IMS terminal attaches to a packet network, such as the GPRS network, through a radio link. The IMS also supports other

types of devices and accesses. PDAs (Personal Digital Assistants) and computers are examples of devices that can connect to the IMS. Examples of alternative accesses are WLAN or ADSL.

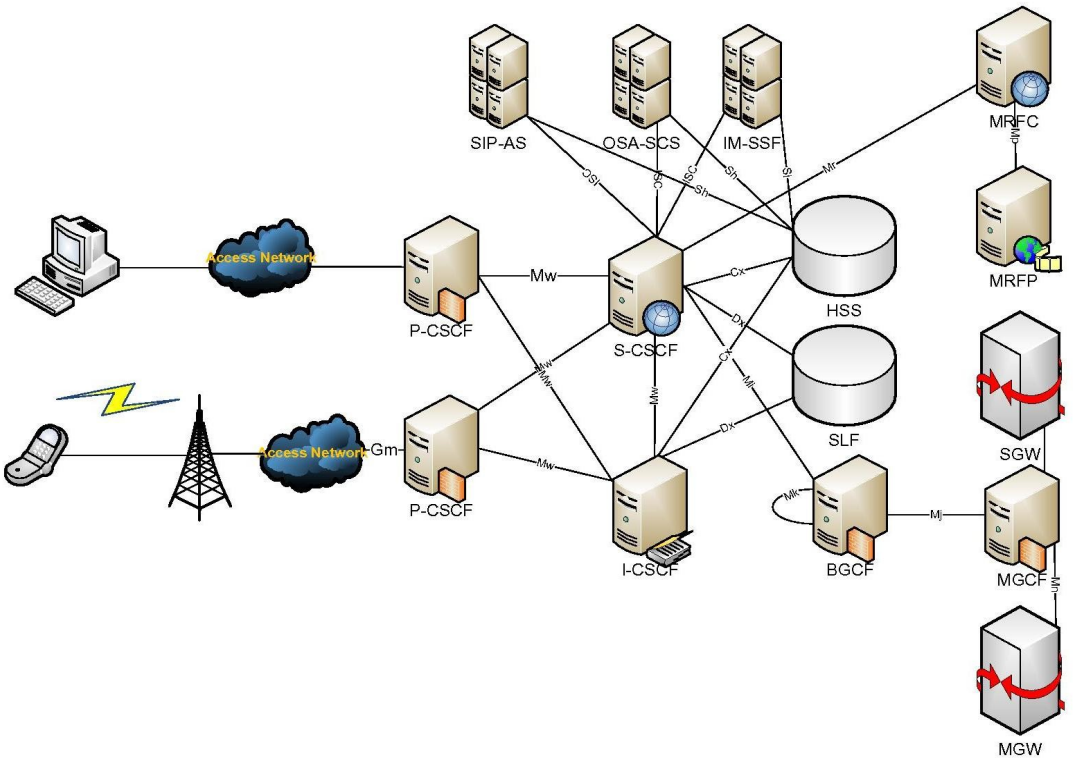

FIGURE 1.1: IMS Architecture Overview

The remainder of Figure 1.1 shows the nodes included in the so-called IP Multimedia Core Network Subsystem. These nodes are:

1. One or more SIP servers (Session Initiation Protocol servers), collectively known as CSCFs (Call Session Control Functions). The Call Session Control Function (CSCF) establishes, monitors, supports and releases multimedia sessions and manages the user's service interactions. It can play three different roles: Serving-, Proxy- or Interrogating- Call Session Control Function (S-, P- and I-CSCF). The S-CSCF is the proxy server controlling the communication session. It invokes the Applications Servers related to the requested services. It is always located in the home network. The P-CSCF is the IMS contact point for the SIP user agents. The I-CSCF provides a gateway to other domains. It is used essentially for topology hiding or if several S-CSCF are located in the same domain.

2. One or more user databases, called HSSs (Home Subscriber Servers) and SLFs (Subscriber Location Functions). The Home Subscriber Server (HSS) is a secure

database storing user profile information. It can be accessed by the S-CSCF using Diameter protocol. It is to be noted that the HSS can be seen as an evolution of the former Home Location Register (HLR). In case several HSSs are used in a domain, a Subscriber Location Function (SLF) is required. The SLF is a simple database indicating in which HSS a user profile is located.

3. One or more ASs (Application Servers), namely SIP-AS (SIP Application Server), OSA-SCS (Open Service Access-Service Capability Server) and IM-SSF (IP Multimedia Service Switching Function).

4. One or more MRFs (Media Resource Functions), each one further divided into MRFC (Media Resource Function Controllers) and MRFP (Media Resource Function Processors). The MRFC is used for controlling a MRFP that essentially provides trans-coding and content adaptation functionalities.

5. One or more BGCFs (Breakout Gateway Control Functions). The BGCF "selects the network in which PSTN breakout is to occur and – within the network where the breakout is to occur – selects the MGCF". This means that it is used for interworking with the Circuit Switched domain.

6. One or more PSTN gateways (Public Switched Telephone Network gateways), each one decomposed into an SGW (Signaling Gateway), an MGCF (Media Gateway Controller Function) and an MGW (Media Gateway). The MGCF is, as its name indicates, used to control a MGW.

1.2.3 Overview of Protocols used in the IMS

1. Session Control Protocol

   The protocols that control the calls play a key role in any telephony system. SIP (Session Initiation Protocol) is chosen as the session control protocol for the IMS. SIP is a protocol to establish and manage multimedia sessions over IP networks. SIP makes it easy to create new services. Since SIP is based on HTTP, SIP service developers can use all the service frameworks developed for HTTP, such as CGI (Common Gateway Interface) and Java servlets.

2. The AAA protocol

   In addition to the session control protocol there are a number of other protocols that play important roles in the IMS. Diameter is chosen to be the AAA (Authentication, Authorization, and Accounting) protocol in the IMS. Diameter is an evolution of RADIUS (Remote Authentication Dial In User Service), which is a protocol that is widely used on the Internet to perform AAA.

3. Other Protocols

   In addition to SIP and Diameter there are other protocols that are used in the IMS.

   H.248 and its packages are used by signaling nodes to control nodes in the media plane (e.g. a media gateway controller controlling a media gateway). H.248 was jointly developed by ITU-T and IETF and is also referred to as the MEGACO (MEdia GAteway COntrol) protocol.

   RTP (Real-Time Transport Protocol) and RTCP (RTP Control Protocol) are used to transport real-time media, such as video and audio.

1.3 Chapter Structure

The structure of chapters in this report is described as the followings:

- **Chapter 1 – Introduction:** It gives an overview of the project including a brief introduction to background knowledge about IMS concepts and presents general layout of the whole report.
- **Chapter 2 – Project Description:** This chapter describes the project in details regarding project purpose, requirements and method.
- **Chapter 3 – Technical Solution:** This chapter focuses on system design and implementation.
- **Chapter 4 – Evaluation:** This chapter will give verifications on completed system prototype. It goes through system test, integration as well as test the target system – IMS Voice Mail System (VMS).
- **Chapter 5 – Future Work:** Future work on the current prototype will be put in this chapter.
- **Chapter 6 – Conclusion:** The last chapter puts emphasis on summarizing achievement of the implemented system prototype together with the challenges the author faced during the project.
- **Glossary:** Important terminologies covered in this thesis will be listed out in the glossary section.
- **Appendix:** Appendix A will clearly show SIP protocol signaling flow and message contents achieved in the system prototype. Appendix B is a table that lists out all defined Diameter commands some of which are used in the system prototype.

- **References:** All the references relevant to this report will be presented in the references section.

Chapter 2

Problem Description

2.1 Project: IMS Test Environment Simulator

2.1.1 Goals and Motivation

The thesis project focuses on investigating and implementing a test environment which simulates different functional entities in the IMS core network. It should be used for testing IMS based systems which support protocols such as SIP, RTP and Diameter.

The main purpose is to facilitate development of IMS products without accessing to a live IMS network. The major target system is an IMS Voice Mail System so that all necessary functionalities used by it should be in the implementation scope of this thesis project. The mentioned IMS Voice Mail System is another thesis completed by two other students in Mobile Arts AB. In addition, a handset simulator that is capable of depositing and retrieving voice mails should also be implemented.

Thus the goals of this thesis could be summarized as:

1. To develop a test environment for IP Multimedia Subsystem (IMS) based systems which supports protocols:
   - Session Initiation Protocol (SIP)
   - Real-time Transport Protocol (RTP)
   - Diameter protocol
2. To provide a test environment targeted to a third-party thesis project regarding an IMS Voice Mail System (VMS)

The following is out of scope for the Thesis:

1. Operation, Administration and Maintenance (OAM) support
2. Charging support
3. Product documentation
4. System testing with respect to stability, redundancy, OAM, charging

2.1.2 Requirements and Management

The general tasks to be fulfilled in the thesis project contain:

1. Design of a simulated IMS network architecture
2. Implementation of the entities in IMS which are necessary for testing a Voice Mail System (VMS)
3. Implementation of a script-able IMS client
4. Implementation of the relevant protocols

Generally speaking, IMS Test Environment Simulator system should include but not limited to the following functionalities:

1. Originating call
2. Terminating call
3. Subscriber register with authentication
4. Subscriber re-register
5. Subscriber de-register
6. Instant Message transfer
7. Test environment re-configuration
8. Test call generator
9. Test object response validating tool
10. Test object characteristics measurements

The system shall consist of the following separated nodes (functional entities in IMS network):

1. IMS-Client: Talks via SIP protocol, subscriber attachment/detachment.
2. CSCF: Proxy via SIP protocol to communicate with VMS and IMS-Client.
3. HSS: Stores subscriber and service related data, contains triggers for subscriber registration notification, talks via Sh [12, 13] and Cx [14, 15] interfaces with VMS and CSCF over Diameter protocol.
4. ENUM/DNS [16]: Supports ENUM and DNS queries. (optional)

Although the thesis project is a single-person project, I tried to make the development procedure as formal and organized as possible. Therefore I used the well-known project management method: Waterfall model. I followed the path: requirement specification – system design – system implementation – system integration – system verification. The necessary documentation for each stage is kept. Besides, SVN is used to maintain all the source code.

2.1.3 Development Method

2.1.3.1 Erlang/OTP and process structure

Due to high performance (i.e. concurrent, distributed, scalable, fault-tolerant, robust etc) consideration as well as telecommunication software demanding soft real-time, Erlang/OTP (http://www.erlang.org/) is chosen as the development language and library for the thesis project. Erlang is a concurrent functional programming language and runtime system. The main strength of Erlang is its high concurrency ability. Erlang has a small but powerful set of primitives to be used to create processes and make processes communicate between each other. Erlang needs a run-time environment since it is by default an interpreted language. Open Telecom Platform (OTP) is an Erlang library developed by Ericsson AB that defines a large portion of Erlang behavior, since much of Erlang development is based on this library. This IMS Test Environment Simulator system is developed under the Open Telecom Platform.

A basic design principle in Erlang/OTP is the **supervision tree**. This is a process structuring model based on the idea of **workers** and **supervisors**. In Figure 2.1, square boxes represent supervisors and circles represent workers.

1. Workers are processes which perform computations, that is, they do the actual work.

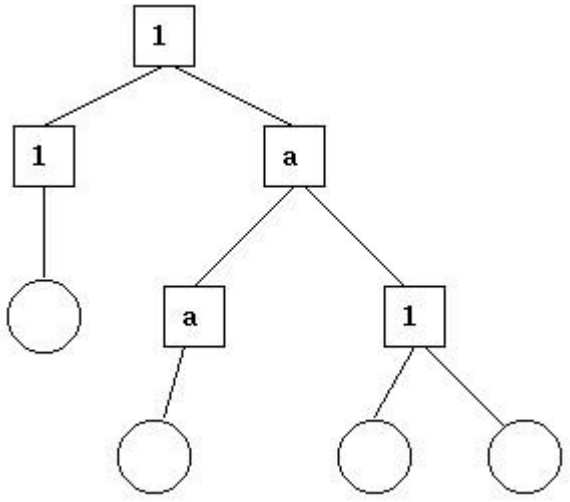

FIGURE 2.1: Supervision Tree

2. Supervisors are processes which monitor the behavior of workers. A supervisor can restart a worker if something goes wrong.

3. The supervision tree is a hierarchical arrangement of code into supervisors and workers, making it possible to design and program fault-tolerant software.

One highlight of Erlang is its high performance achieved by parallel computing. Since Erlang processes are so light-weight, we do not hesitate to separate functionality into processes, even if those processes are show-lived. Thus, things that go wrong will cause a process to terminate abnormally, and it is up to the linked process to handle that error. If the linked process cannot handle a condition, it should also terminate abnormally allowing a higher-level process to handle the problem. This leads to the Erlang concept, supervision tree, which is mentioned in the section above. The supervisor process will make sure the monitored work processes are alive, for example by restarting them if they die abnormally. This concept provides very good scalability and robustness to the application. The application can have high performance by creating many worker processes to do computation in parallel as well as be fault-tolerant with supervisor processes monitoring all the worker processes or even other supervisor processes.

Another advantage of building a supervision tree is that if a supervisor process is terminated by a higher-level supervisor, then it will terminate its worker processes before terminating itself, which leads to a graceful shutdown automatically. I have applied these concepts to my applications involved in this thesis project.

2.1.3.2 Development tool and environment

As the thesis requires close cooperation with system design at Mobile Arts AB, the thesis project was therefore done mainly at Mobile Arts AB premises in Stockholm. The workstation being used is x86 PC with Linux Ubuntu distribution and Windows XP. For version handling, a Subversion server is set up and all code is committed and kept updated. Emacs and Make are chosen as the development tools for the project. All code shall be documented with edoc (a code documentation tool in Erlang/OTP) when everything is done in the end.

Chapter 3

Technical Solution

In Chapter 2, a general overview of IMS Test Environment Simulator system has been given. Next step is to clarify how to implement the system. In this chapter, I will give the technical solution divided into two main parts: (1) System Design and (2) System Implementation. This is the essential chapter of the whole paper which presents thorough internal design details of different components in the system as well as the way used to achieve them.

3.1 System Design

In this section, I will present the overview of system architecture for IMS Test Environment Simulator at first. Then the system will be broken down into three main parts: (1) IMS-Client (handset simulator), (2) CSCF (Call Session Control Function) and (3) HSS (Home Subscriber Server). They will be described one by one thoroughly in the following sub-sections. The ending part of this section is a detailed description about all essential protocols that are involved in the system.

3.1.1 System Architecture Overview

Since the scope of this thesis project is defined to set up a test environment for testing the IMS Voice Mail System (IMS VMS), it is not necessary to simulate all functional entities or nodes as well as all protocol interfaces in real IMS core network. Instead, some essential functional nodes and protocol interfaces relevant to IMS VMS system testing are selected to simulate and implement. Thus, the thesis has been reduced to a reasonable level of complexity.

Based on this goal, IMS Test Environment Simulator is mainly composed of the following five parts: (Please refer to Figure 3.1.)

1. **IMS-Client:** talk via SIP + subscriber attach/detach
2. **CSCF(Call Session Control Function):** proxy via SIP + communicate with VMS and IMS-Client
3. **HSS(Home Subscriber Server):** store subscriber data and state + triggers for notification etc + talk via Sh/Diameter and Cx/Diameter with VMS and CSCF
4. **VMS(external):** Voice Mail System application which is the target system
5. **ENUM/DNS(optional):** not implemented, only hard coded to get tests to work

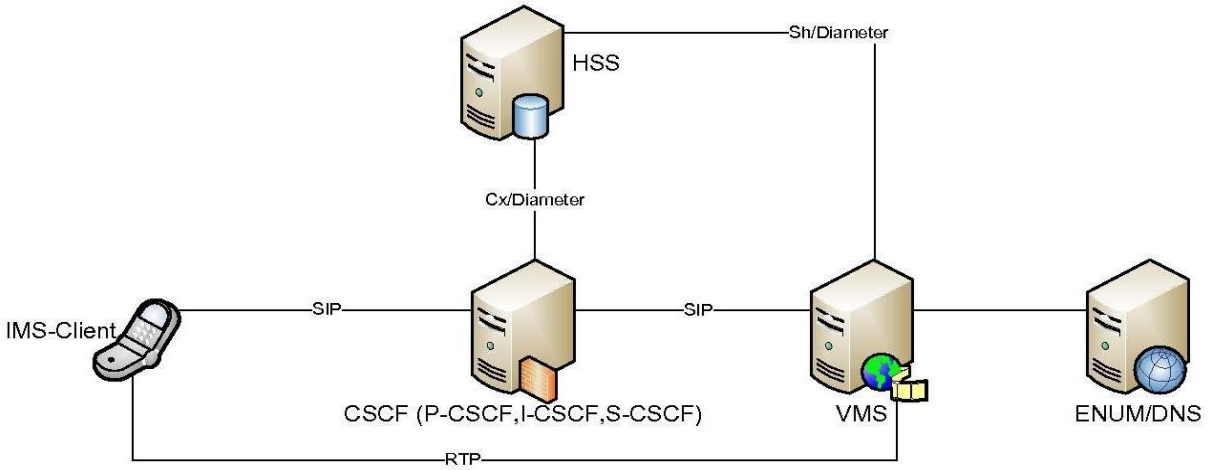

FIGURE 3.1: IMS Test Environment Simulator System Architecture

Among them, the actual CSCF according to different functionalities are divided into three categories, namely, S-CSCF (Serving-CSCF), P-CSCF (Proxy-CSCF) and the I-CSCF (Interrogating-CSCF), see Figure 1.1. However in IMS Test Environment Simulator system, in order to simplify the implementation, I decided to combine these three CSCFs into a general CSCF which contains the same functionality. It can be simplified in this way because:

In the live IMS network, since the number of mobile users is tremendous, it needs more than one S-CSCF to serve all the users. So when a user accesses IMS network and requires connect to S-CSCF, P-CSCF needs to look for a particular S-CSCF for the user. P-CSCF will inquire I-CSCF to get the address of the particular S-CSCF. Then P-CSCF can locate that S-CSCF and forward user's request to it to begin following message interactions between user and the S-CSCF.

However, in this thesis, the mobile user terminal is also simulated by program in which it can set S-CSCF address where to send the request. So it does not need to query I-CSCF to locate S-CSCF. Besides, in this IMS Test Environment Simulator system, it does not require multiple S-CSCFs to handle requests. All requests can be handled by single S-CSCF and only SIP messages need to be routed between P-CSCF and S-CSCF.

Therefore, my design idea here is that under the premise of not affecting any test functionality, the three kinds of CSCFs (P-CSCF, I-CSCF and S-CSCF) can be replaced by a general CSCF which can achieve the same functionality. The main test task is to set up SIP/RTP session between IMS-Client and IMS VMS system. So it is not necessary to separately implement all three kinds of CSCFs but one general CSCF is enough for test purpose.

IMS Test Environment Simulator system supports SIP, RTP, and Diameter protocols which are major communication protocols in the IMS network. These protocols are used for communication between various functional entities within IMS Test Environment Simulator system as well as communication with external test object IMS VMS system.

As it is shown in Figure 3.2 according to my design mentioned in previous section, the design scheme of the whole system can be mainly divided into three separate applications:

1. IMS-Client: it simulates subscribers handset. Its SIP handler and RTP handler processes are capable of sending and receiving SIP request/response and RTP package.

2. CSCF: it simulates Call Session Control Function which is a key element in IMS core network. It can proxy SIP request/response by its SIP listener and worker processes. It also includes Cx Handler which is an interface to communicate with HSS over Diameter protocol.

3. HSS: it simulates Home Subscriber Server which is the major database server in IMS core network. It consists of Cx Listener/Worker and Sh Listener/Worker processes. They can access the underlying database storing all subscriber related information. HSS communicates with external CSCF via Cx interface and external Application Server (in this thesis, it is IMS VMS) via Sh interface over Diameter protocol.

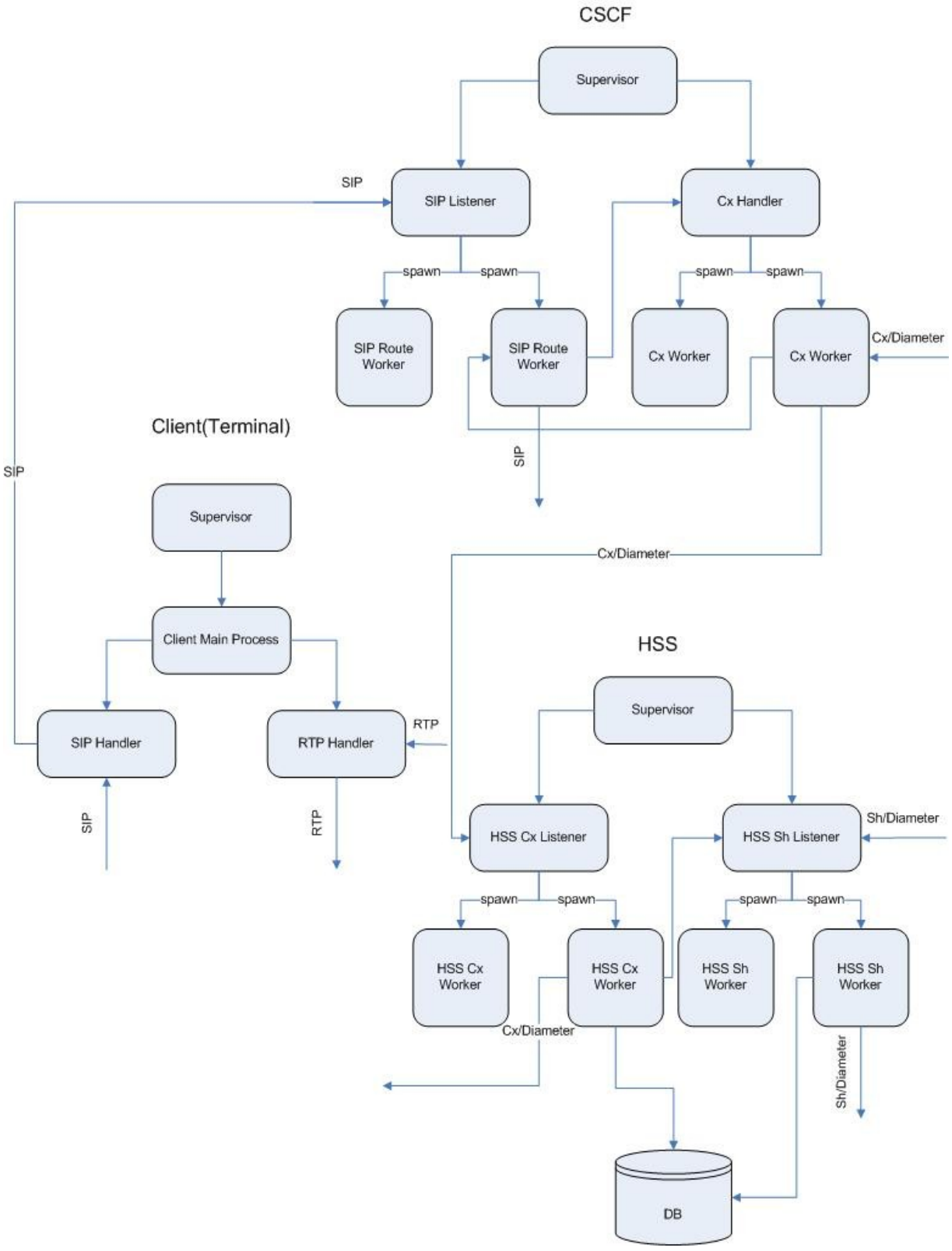

FIGURE 3.2: Design Scheme for IMS Test Environment Simulator (high-level process structure)

3.1.2 System Environment Functionality

Besides the requirement to simulate key elements within IMS core network such as IMS-Client, CSCF and HSS described in last section, the following test environment specific functionalities have to be developed/in-sourced as well:

1. Test environment re-configuration
2. Test call generator
3. Test object response validating tool
4. Test object characteristics measurements

3.1.2.1 Test environment re-configuration

1. The test environment shall provide interfaces to let the user reset all the configuration parameters and restart the whole system.
2. The test environment shall provide a simple GUI based on Web GUI libraries from Mobile Arts AB to let the user do all the re-configuration setting work. (Optional)

3.1.2.2 Test call generator

1. The Test environment shall be able to generate test calls to the targeted test object (focus on VMS in this Thesis) for test purposes.
2. The test environment shall be able to set the number of test calls to be generated.
3. The test environment shall be able to concurrently generate massive test calls.
4. The test environment shall be able to generate a number of test calls continuously during a period of time specified by system configuration settings.
5. The Test environment shall provide a simple GUI based on Web GUI libraries from Mobile Arts AB to let the user do the entire test calls generating related work. (Optional)

3.1.2.3 Test object response validating tool

1. The Test environment shall be able to provide the tool to valid all the responses sent back from the test object (focus on VMS in this Thesis) in the aspects as the followings:

   (a) SIP and/or RTP Packet loss rate

   (b) SIP and/or RTP Packet content accuracy

2. The essential use case to be validated shall be:

   (a) B is detached

   (b) A calls B and deposits a voice mail to B

   (c) B is attached and gets notification of having a new voice mail

   (d) B retrieves the voice mail from A

3. The validating tool shall be able to display validation results to the user on the Test environment interface and simultaneously log the results into files for further reference.

4. The test environment shall provide a simple GUI of the validating tool based on Web GUI libraries from Mobile Arts AB to the user. (Optional)

3.1.2.4 Test object characteristics measurements

1. The Test environment shall be able to measure the characteristics of the test object (focus on VMS in this Thesis) in the aspects as the followings:

   (a) Minimal response time of test object (focus on VMS in this Thesis)

   (b) Average response time of test object (focus on VMS in this Thesis)

   (c) Maximal response time of test object (focus on VMS in this Thesis)

   (d) Minimal setting up time of SIP and/or RTP session

   (e) Average setting up time of SIP and/or RTP session

   (f) Maximal setting up time of SIP and/or RTP session

   (g) Time (of response and/or setting up session) variation to the increment of test calls

   (h) The maximum number of test calls which the test object (focus on VMS in this Thesis) can handle

2. The essential use case to be measured shall be:

   (a) B is detached

   (b) A calls B and deposits a voice mail to B

   (c) B is attached and gets notification of having a new voice mail

   (d) B retrieves the voice mail from A

3. The test environment shall be able to display measurements results to the user and simultaneously log the results into files for further reference.

4. The Test environment shall provide a simple GUI based on Web GUI libraries from Mobile Arts AB to let the user do all the test object characteristics measurements related work. (Optional)

3.1.3 Traffic Scenario

This simulator is targeted on testing VMS (Voice Mail System). So its main traffic scenarios will be as the following in accord to my design:

3.1.3.1 Scenario 1

Subscriber A attaches i.e. turns on the mobile phone/device to connect into IMS network. Please refer to Figure 3.3.

Flow (1) - (2): If VMS has new voice mails for Subscriber A, it will register notification of Subscriber A's availability from HSS while Subscriber A is detached.

Flow (3) - (12): As soon as Subscriber A attaches and becomes available, HSS will send notification to VMS about Subscriber A's availability.

Flow (13) - (16): VMS will send SIP MESSAGE [17, 18] directly to Subscriber A about his/her voice mail box status or other related information.

3.1.3.2 Scenario 2

Subscriber A deposits a voice mail for Subscriber B in the VMS. Please refer to Figure 3.4.

Flow (1) - (10): Subscriber A wants to leave a voice mail for Subscriber B. So firstly Subscriber A needs to set up SIP session with VMS.

Flow (11) - (18): After SIP session between Subscriber A and VMS has been set up, an RTP session will be initiated in between. Then Subscriber A starts to send RTP package containing voice digital data towards VMS. VMS will store these data into its disk array for future retrieval.

Flow (19) - (22): As soon as Subscriber A finishes transferring voice mail, a SIP BYE message will be sent to VMS and SIP session will be close after receiving 200 OK message from VMS.

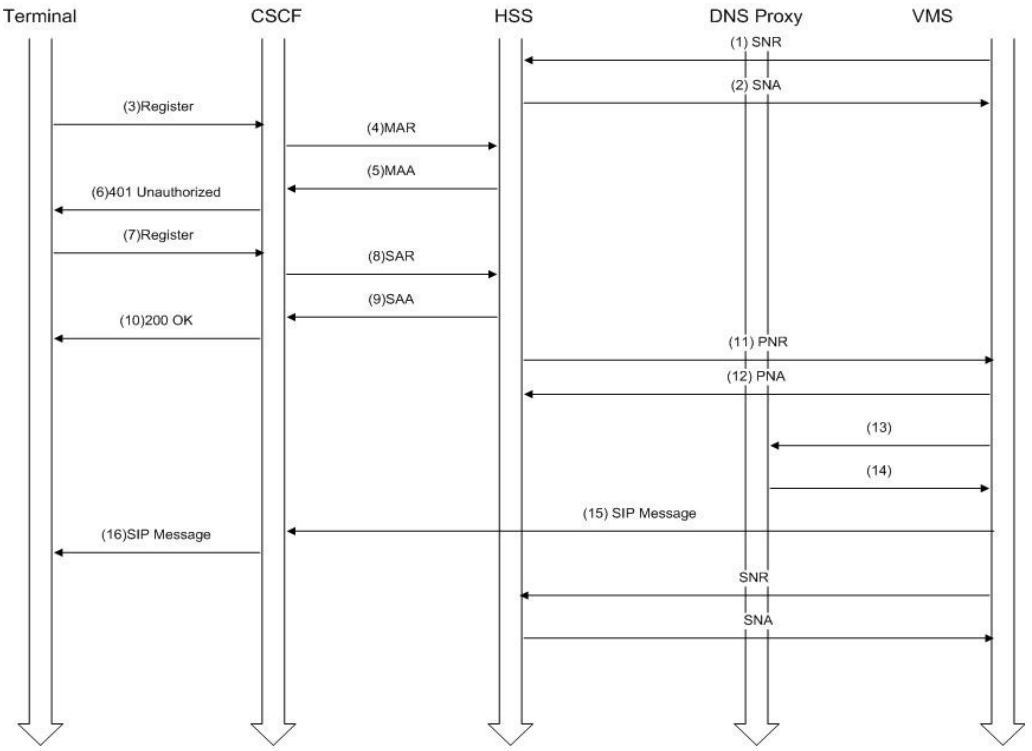

FIGURE 3.3: Subscriber attach

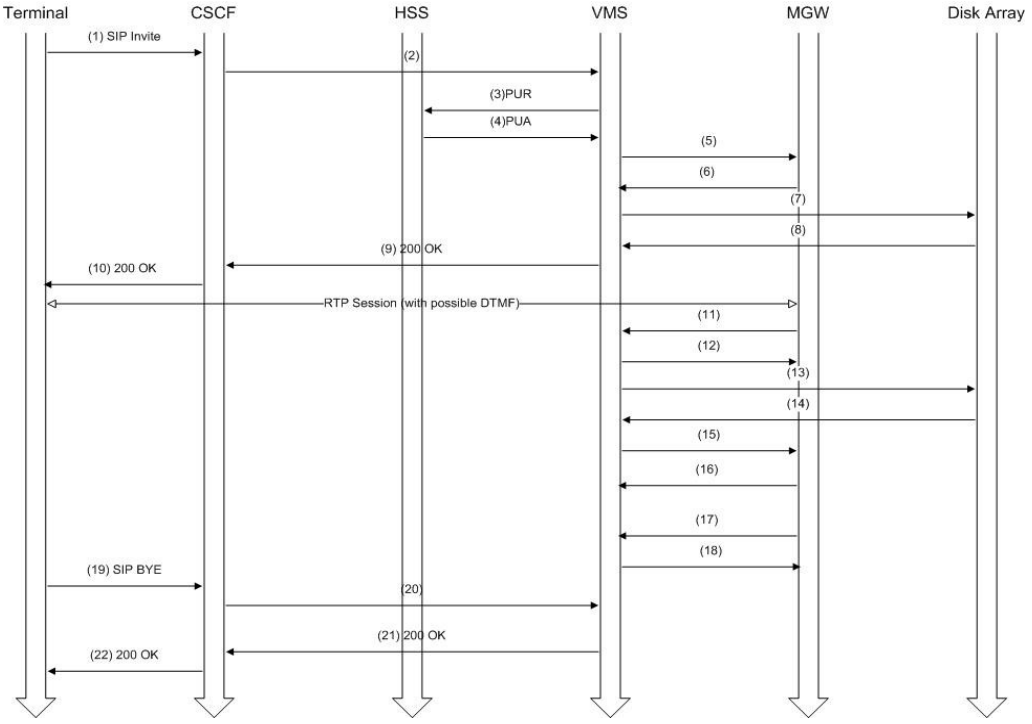

FIGURE 3.4: Subscriber deposit voice mail into VMS

3.1.3.3 Scenario 3

Subscriber B retrieves the voice mail left by Subscriber A from the VMS. Please refer to Figure 3.5.

Flow (1) - (8): When Subscriber B attaches into IMS network, he/she will get notified by VMS about his/her voice mail box status. (refer to Scenario 1) Suppose Subscriber B has a new voice mail left by Subscriber A in his/her voice mail box on VMS. So Subscriber B starts to initiate SIP session with VMS.

Flow (9) - (16): After SIP session between Subscriber B and VMS has been set up, an RTP session will be initiated in between. Then Subscriber B starts to receive RTP package containing voice digital data from VMS.

Flow (17) - (22): As soon as Subscriber B finishes receiving the voice mail, a SIP BYE message will be sent to VMS and SIP session will be close after receiving 200 OK message from VMS.

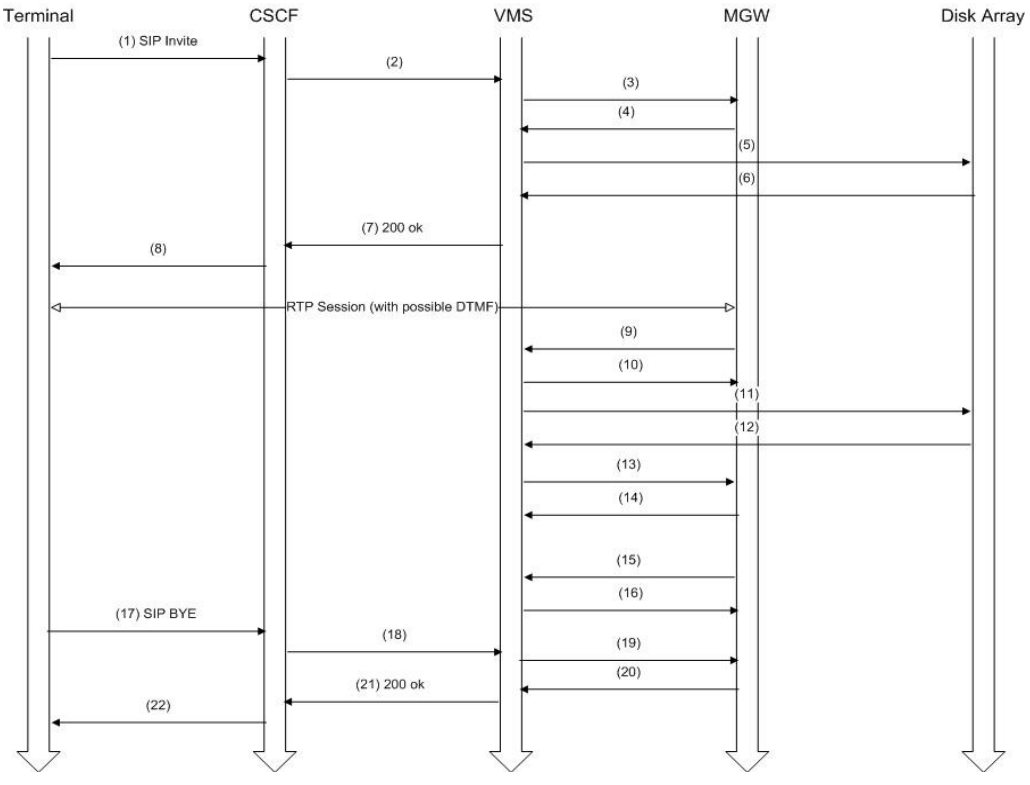

FIGURE 3.5: Subscriber retrieve voice mail from VMS

3.1.4 Database Design

The Home Subscriber Server (HSS) [19] is the master user database server that supports the IMS network entities that actually handle calls and sessions. Technically, the HSS is an evolution of the HLR (Home Location Register), which is a GSM node. The HSS contains all the user-related subscription data required to handle multimedia sessions. These data include, among other items, location information, security information (including both authentication and authorization information), user profile information (including the services that the user is subscribed to), and the S-CSCF (Serving-CSCF) allocated to the user.

Some new information is needed for IMS service in addition to the existing user database in the HLR (Home Location Register) environment. Thus, the user profile database design of the IMS subscriber is an important part of building the HSS system for efficient database control and system performance. In this thesis, a design and implementation of HSS user profile database is given. Figure 3.6 shows the design schema of the database. It consists of a total of seven relations. Each relation is connected with others through one-to-one or one-to-N instance mapping by primary keys.

Mnesia is chosen to implement the database. Mnesia is a distributed Database Management System (DBMS), appropriate for telecommunications applications and other Erlang applications which require continuous operation and exhibit soft real-time properties. Mnesia itself is written in Erlang so that it is very suitable to use when all the rest parts of this IMS Test Environment Simulator are implemented using Erlang. The distributed and scalable ability of Mnesia ensures HSS to be stable even if subscribers are growing quickly. Its soft real-time access speed provides high performance to HSS handling a great amount of requests from external application servers simultaneously.

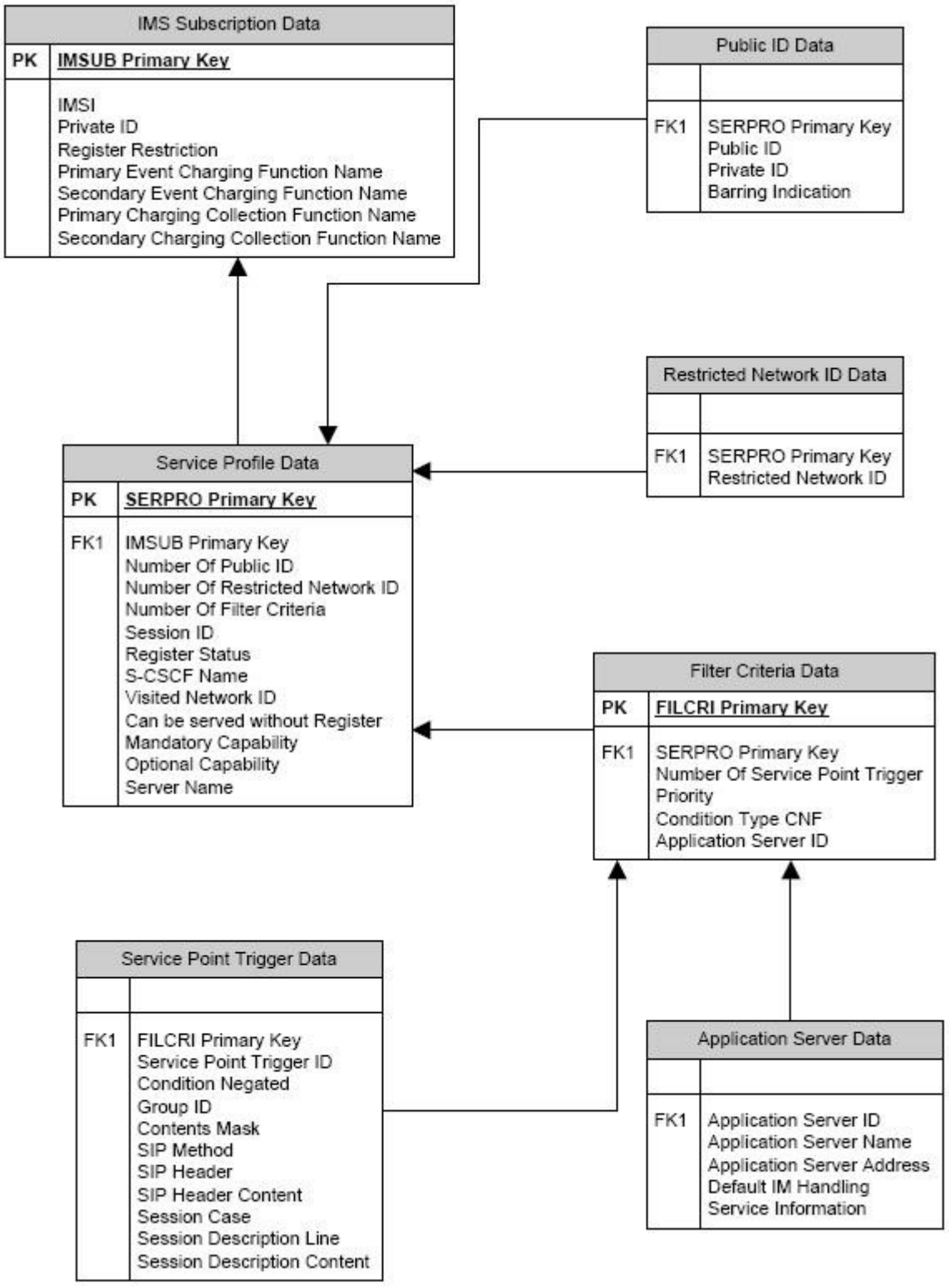

FIGURE 3.6: HSS User Profile Database Schema

3.2 System Implementation

3.2.1 System Components

As mentioned in above sections, the simulator is composed of three major components:

1. IMS-Client
2. Call Session Control Function (CSCF)
3. Home Subscriber Server (HSS) Now let us dive into each component application to have a clear and complete picture about its internal design principle.

3.2.1.1 IMS-Client

1. Definition

   IMS-Client is a program that simulates a subscriber handset's simple functionalities mainly on supporting SIP and RTP protocols. It can communicate with CSCF for registration and authentication. It is also able to set up SIP call session with IMS VMS and transfer audio data via RTP connection in between.

2. Functionality

   The basic functional requirements of IMS-Client could be summarized as the following:

   (a) It shall implement SIP stack and Network-to-Network-Interface (NNI) in the IMS-Client in accord with RFC 3261.

   (b) The IMS-Client shall be able to communicate with CSCF via SIP.

   (c) The IMS-Client shall be able to originate test calls to send to the CSCF.

   (d) The IMS-Client shall be able to terminate test calls sent from the CSCF.

   (e) The IMS-Client shall be able to register with authentication to the CSCF.

   (f) The IMS-Client shall be able to re-register to the CSCF.

   (g) The IMS-Client shall be able to de-register from the CSCF.

   (h) The IMS-Client shall be able to handle Instant Message (SIP MESSAGE) sent from the CSCF.

   (i) It shall implement RTP stack in the IMS-Client in accord with RFC 3550.

(j) The IMS-Client shall be able to communicate with VMS (outside) via RTP. Thus, the essential task is to implement SIP and RTP stack for IMS-Client since it relies on SIP to communicate with CSCF for different operations and RTP to transfer media data to/from IMS VMS.

3. Design and Implementation

According to the functionality of IMS-Client given above, I have drawn a FSM (Finite State Machine) design diagram of its internal logic which is shown as Figure 3.7. It shows the internal logic of IMS-Client. After the program starts, IMS-Client will first go to "Init State" which represents subscriber turning on mobile device.

Then IMS-Client will send out SIP REGISTER request towards external CSCF and enters "Registering State" waiting for reply. CSCF will send back SIP 401 Unauthorized response. IMS-Client gets the response and send out SIP REGISTER request with authentication data to CSCF and entering "Waiting State". CSCF will response with SIP 200 OK response. It indicates IMS-Client has successfully registered in CSCF and be able to access the IMS network. This process simulates in real world a subscriber's mobile device attaching into the mobile network and allowing making calls or other operations.

After IMS-Client succeeds registering and enters "Success State", the VMS will get notified about this subscriber availability in the network. VMS will send an SIP MESSAGE to IMS-Client about its voice mail box status. IMS-Client program will enter "Terminated State" which means the registration and notification process has finished. The program will be idle until next operation invoked by the user.

Next operation should be to initiate SIP session towards VMS and transfer audio data via RTP connection in between. So the IMS-Client re-enters "Init State" this time but send out SIP INVITE request instead of SIP REGISTER towards CSCF then enters "Active State". CSCF will send back SIP 100 Trying response at once. CSCF will route the SIP INVITE request to VMS and get SIP 180 Ringing response. CSCF send the SIP 180 Ringing response back to IMS-Client making it enter "Trying State" to wait further response. VMS can either response with SIP error messages or 200 OK. IMS-Client will enter "Error State" and restart if receiving error message or continue to "Confirm State" when getting SIP 200 OK response. Until this step, IMS-Client has gone through whole process of setting up a SIP session with VMS.

Assuming that IMS-Client succeeds setting up the SIP session, it finally enters "Calling State". IMS-Client will try to connect VMS and start sending or receiving audio data (depending on depositing voice mail or retrieving voice mail in VMS)

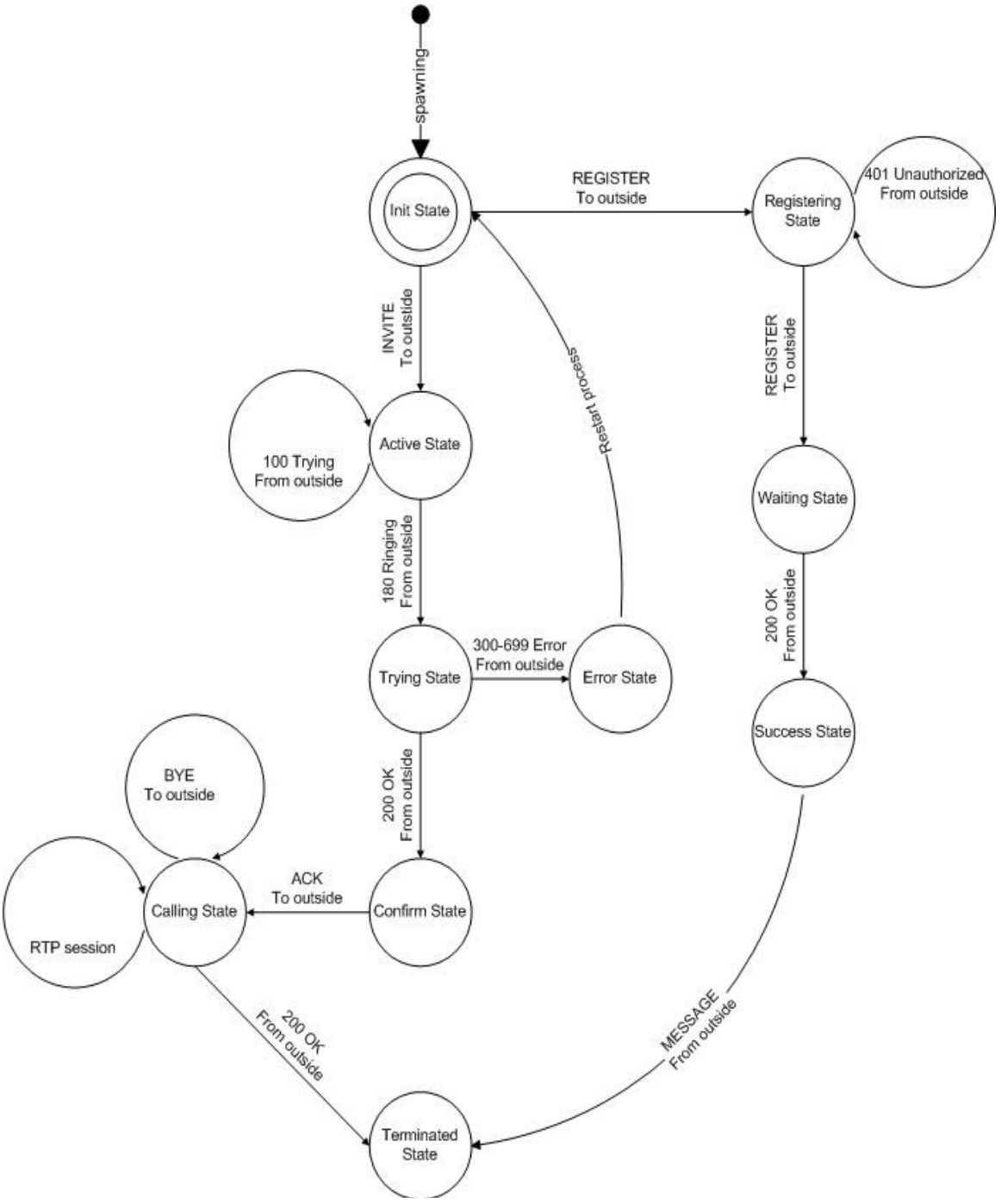

FIGURE 3.7: IMS-Client FSM Design Diagram

using RTP protocol. After data transfer is finished, IMS-Client will close RTP connection and send out SIP BYE request. As soon as it receives SIP 200 OK response from VMS, it will enter "Terminated State". The whole SIP call session is completed at this moment.

According to the design, IMS-Client is implemented as an Erlang application. Its process supervision tree diagram is shown as Figure 3.8.

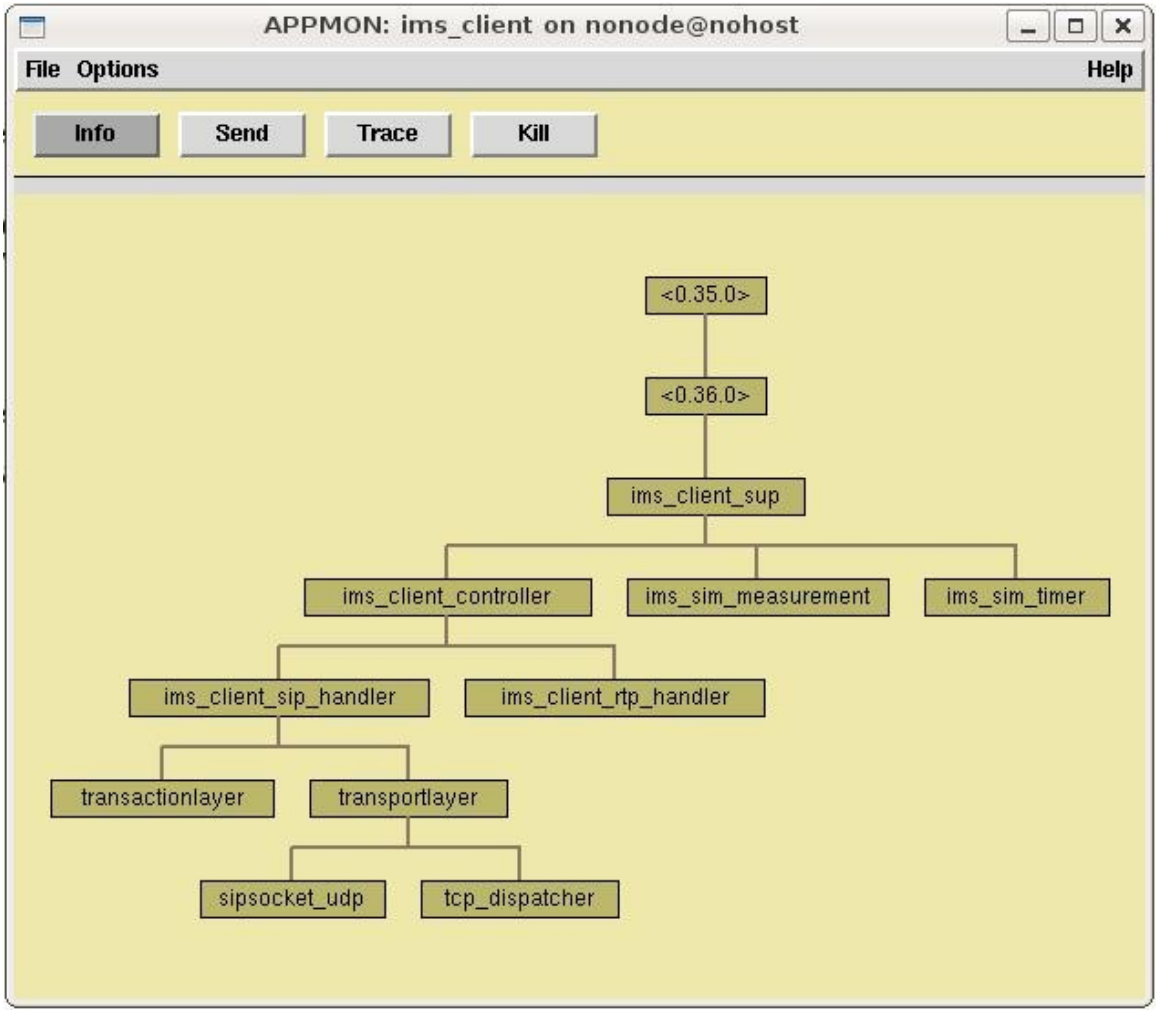

FIGURE 3.8: IMS-Client Process Supervision Tree Diagram

3.2.1.2 CSCF

1. Definition

   Call Session Control Function (CSCF) is a central component to signaling and control within the IMS network. CSCF is responsible for all signaling via Session Initiation Protocol (SIP) and acts as a SIP server or proxy which processes SIP (Session Initiation Protocol) signaling packets in the IMS core network. The CSCF

consists of three specific kinds: (1) Proxy-CSCF (P-CSCF) (2) Serving-CSCF (S-CSCF) (3) Interrogating-CSCF (I-CSCF). But in this thesis scope, a general CSCF with basic features is implemented. It can handle IMS-Client registration and routing SIP messages between IMS-Client and VMS.

2. Functionality

   The essential functional requirements of CSCF are listed out as following:

   (a) It shall implement SIP stack and Network-to-Network-Interface (NNI) in the CSCF in accord with RFC 3261.

   (b) The CSCF shall be able to communicate with IMS-Client and VMS via SIP.

   (c) The CSCF shall be able to proxy SIP messages between IMS-Client and VMS.

   (d) The CSCF shall be able to handle IMS-Client's registration with authentication via SIP.

   (e) The CSCF shall be able to handle IMS-Client's re-registration via SIP.

   (f) The CSCF shall be able to handle IMS-Client's de-registration via SIP.

   (g) The CSCF shall be able to proxy Instant Message (SIP MESSAGE) from VMS to Client.

   (h) It shall implement Diameter stack in the CSCF.

   (i) It shall implement Diameter Cx interface in the CSCF in accord with TS 29.228.

   (j) The CSCF shall be able to communicate with HSS via Diameter Cx interface in order to retrieve subscriber profile data and state from HSS via Diameter Cx interface.

   (k) The CSCF shall be able to authenticate the IMS-Client's registration request with subscriber profile data retrieved from HSS.

   Thus, the essential task is to implement SIP stack, Diameter stack and Cx interface for CSCF since it should be able to proxy SIP messages between IMS-Client and VMS as well as communicate with HSS via Diameter Cx interface to fulfill IMS-Client's registration and authentication requests.

3. Design and implementation

   I have drawn the FSM (Finite State Machine) design diagram of CSCF internal logic which is shown as Figure 3.9. It is in accord to its functional requirements mentioned in last section.

   After the program starts, CSCF will first go to "Init State" waiting for SIP requests from IMS-Client. As soon as CSCF receives SIP REGISTER request from IMS-Client, it enters "Authenticating State" and sends SIP 401 Unauthorized response

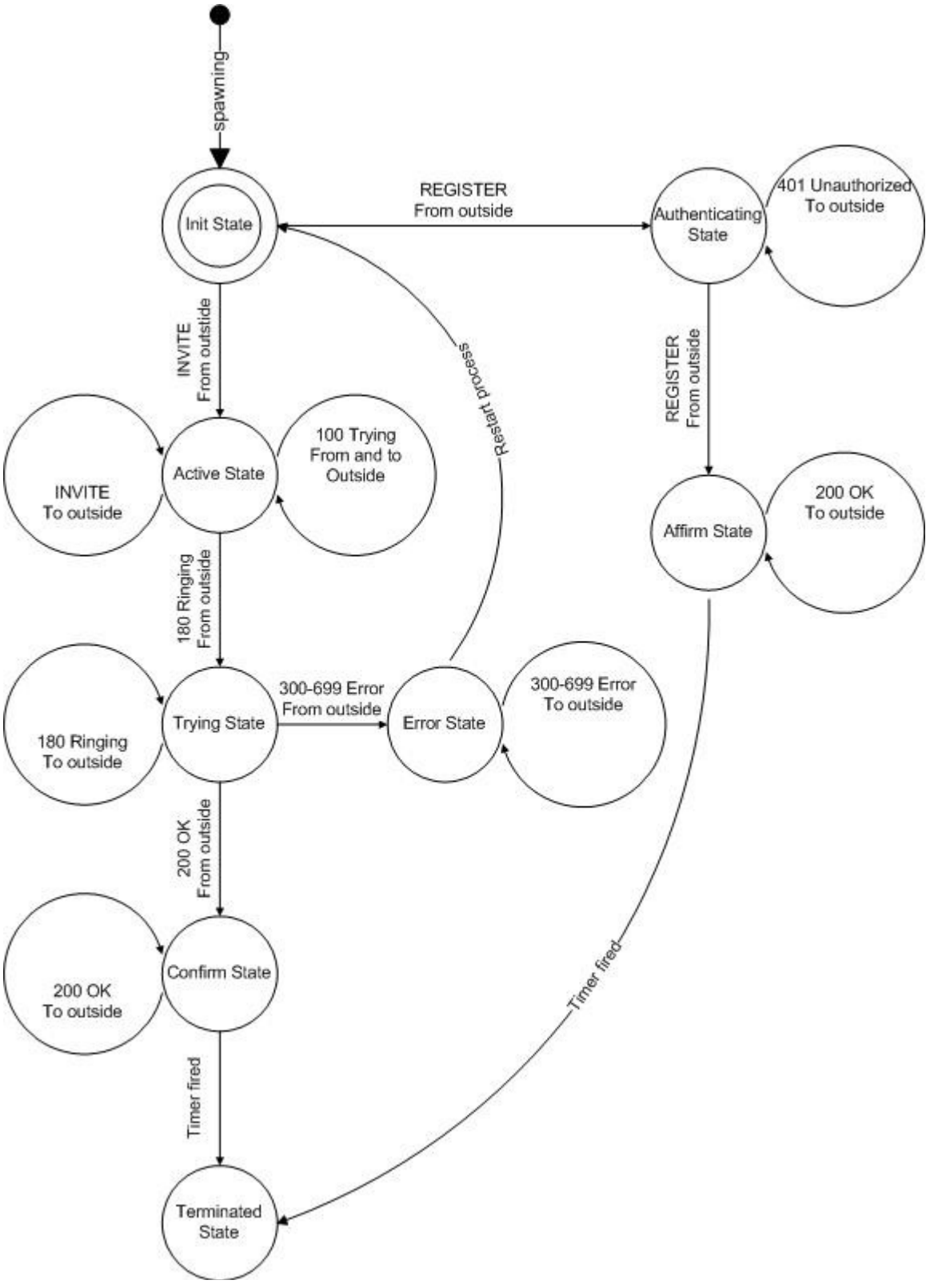

FIGURE 3.9: CSCF FSM Design Diagram

to IMS-Client. Then CSCF will wait until IMS-Client sends back SIP REGISTER request with authentication data. CSCF enters "Affirm State" and send SIP 200 OK response to IMS-Client after authentication. This indicates that IMS-Client has successfully registered in CSCF and be able to access the IMS network.

Another logic path will be SIP session initiation attempt from IMS-Client. CSCF will be a SIP proxy server between IMS-Client and VMS. In this case, IMS-Client starts to send SIP INVITE request to CSCF. CSCF will enter "Active State" and route the SIP INVITE request to VMS. It will get SIP 100 Trying response from VMS and route it back to IMS-Client. As soon as CSCF receives SIP 180 Ringing response from VMS, it will enter "Trying State" and send SIP 180 Ringing response back to IMS-Client. Then VMS can either response with SIP error messages or 200 OK. CSCF will enter "Error State" and route the SIP error message to IMS-Client if it receives error from VMS. Or CSCF will continue to "Confirm State" when getting SIP 200 OK response from VMS and route it back to IMS-Client. Until this step, the SIP session between IMS-Client and VMS is set up. CSCF process will enter "Terminated State" and program goes into idle mode.

According to the design, CSCF is implemented as an Erlang application. Its process supervision tree diagram is shown as Figure 3.10.

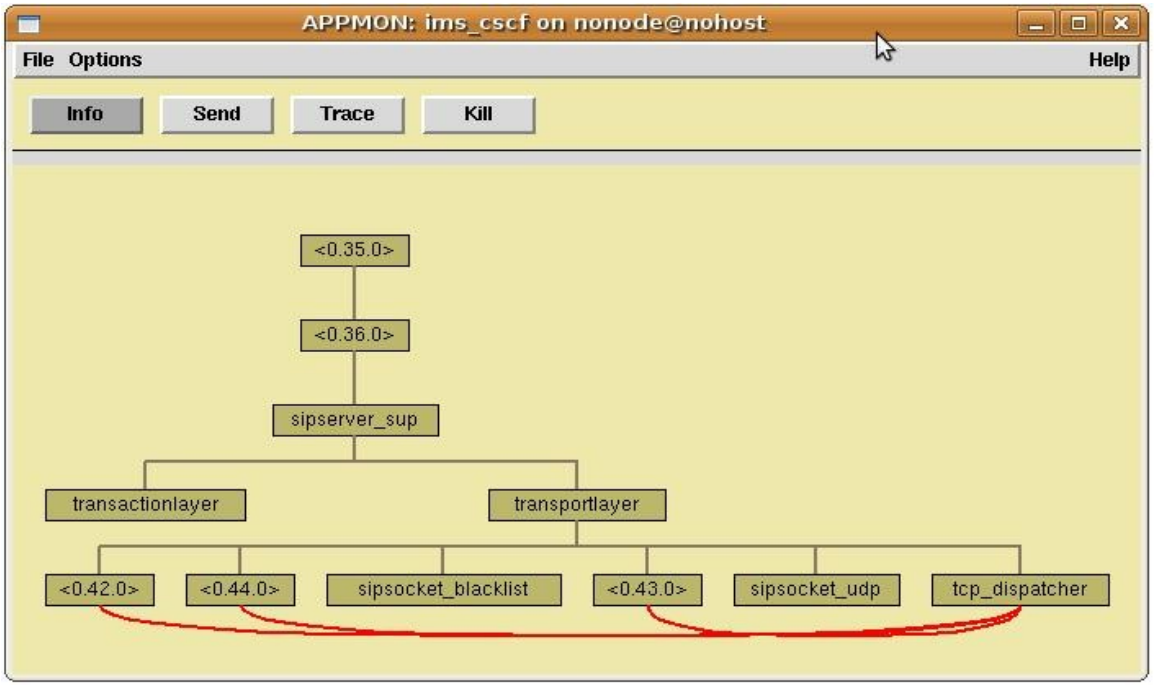

FIGURE 3.10: CSCF Process Supervision Tree Diagram

3.2.1.3 HSS (Cx and Sh interfaces)

1. Definition

   Home Subscriber Server is one of the main components in IMS core network. It is a central database for subscriber information. Cx [14, 15] is a Diameter interface between HSS and CSCF while Sh [12, 13] is a Diameter interface between HSS and AS (application server, in this thesis it is VMS). In this thesis scope, a server application as well as a subscriber database is developed to simulate HSS with it basic behaviors.

2. Functionality

   The essential functional requirements of HSS are listed out as following:

   (a) The HSS shall be able to store all subscriber and service related data.

   (b) The HSS shall have triggers for subscriber registration notification to the VMS.

   (c) It shall implement Sh/Diameter in the HSS.

   (d) The HSS shall be able to communicate with VMS via Sh/Diameter.

   (e) It shall implement Cx/Diameter in the HSS.

   (f) The HSS shall be able to communicate with CSCF via Cx/Diameter.

   (g) The HSS shall be able to handle Diameter requests from CSCF and/or VMS to update the subscriber and/or service data stored in the database.

3. Design and Implementation

   The HSS subscriber database design has been described in detail in section 3.1.4 which will not be repeated here. The focus is on HSS server and Diameter Cx and Sh interfaces.

   As specified in the system requirements, HSS should support Cx and Sh interfaces. Cx is between CSCF and HSS while Sh is between VMS and HSS. The protocol applied is Diameter. Thus HSS server consists of two logic separated parts: Cx handler and Sh handler. Both handlers listen on specified ports to handle Diameter messages sent from CSCF and VMS. They can access underlying HSS database to retrieve or add user profile data according to Diameter requests. For CSCF, it mainly requests downloading user authentication data through Cx interface. For VMS, it requires to subscribe certain users' state in HSS and get notified when changes are performed on that user. So Sh handler will maintain a table to keep track of these notification subscriptions from VMS so that it can send notifications to VMS via Diameter message as soon as the user data is changed in the database.

Both Cx and Sh handler is able to handle simultaneous connections because each session is handled by a new process.

HSS server is implemented as an Erlang application. Its corresponding process supervision tree diagram is shown in Figure 3.11. Only Cx and Sh interfaces are supported in current prototype of simulated HSS. However, it is relative easy to add extra modules to support more interfaces defined in HSS standards in future when new requirements arrive.

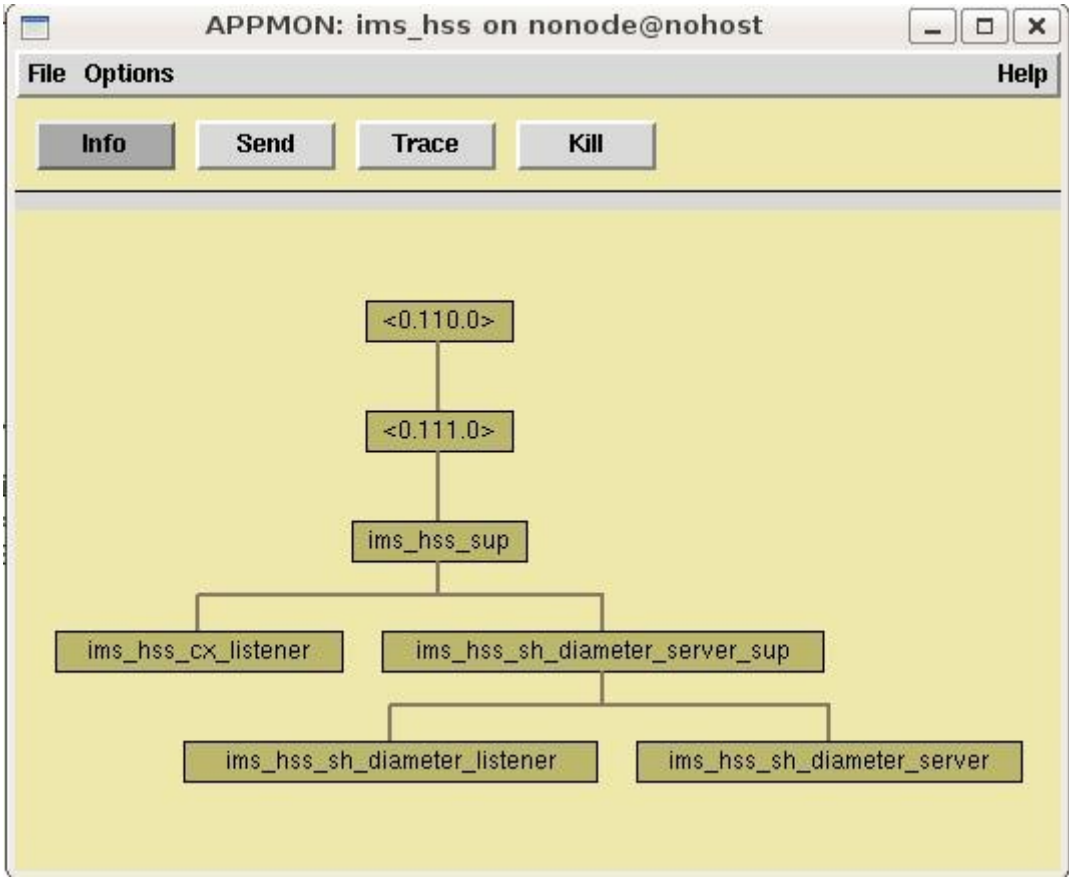

FIGURE 3.11: HSS Process Supervision Tree Diagram

3.2.2 Protocol Interfaces

In IMS core network, there exists quite a few of essential protocols such as SIP (Session Initiation Protocol), Diameter, RTP (Real-time Transport Protocol) and etc. These protocols shall be implemented partially/fully in IMS Test Environment Simulator in order to fulfill the demand of testing IMS based system. Since the main target is to test an IMS VMS, the protocols to be implemented will be narrowed down to the following ones.

3.2.2.1 SIP

1. **What is SIP?**

   The Session Initiation Protocol (SIP) [3, 20] is an application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants. It can be used to create two-party, multi-party, or multi-cast sessions that include Internet telephone calls, multimedia distribution, and multimedia conferences. Media can be added to (and removed from) an existing session. SIP is designed to be independent of the underlying transport layer; it can run on TCP, UDP, or SCTP. It was originally designed by Henning Schulzrinne (Columbia University) and Mark Handley (UCL) starting in 1996. The latest version of the specification is RFC 3261 from the IETF SIP Working Group. In November 2000, SIP was accepted as a 3GPP signaling protocol and permanent element of the IMS architecture. It is widely used as a signaling protocol for Voice over IP, along with H.323 and others.

   SIP employs design elements similar to HTTP-like request/response transaction model. Each transaction consists of a client request that invokes a particular method or function on the server and at least one response [21]. SIP reuses most of the header fields, encoding rules and status codes of HTTP, providing a readable text-based format. SIP uses the Session Description Protocol (SDP) [4, 22] to exchange the session content.

   SIP works in concert with several other protocols and is only involved in the signaling portion of a communication session. SIP is a carrier for the Session Description Protocol (SDP), which describes the media content of the session, e.g. what UDP ports to use, the codec being used etc. In typical use, SIP "sessions" are simply packet streams of the Real-time Transport Protocol (RTP) [5]. RTP is the carrier for the actual voice or video content itself.

   Three key elements in SIP network are UAC, UAS and SIP Proxy Server:

   (a) UAC: User Agent Client is a logical network end-point used to create or receive SIP messages and thereby manage a SIP session, e.g. a SIP phone or an application simulates UAC behavior such as IMS-Client in this thesis. It can send SIP requests to the User Agent Server (UAS).

   (b) UAS: User Agent Server receives the SIP requests and returns a SIP response, e.g. IMS VMS in this thesis.

   (c) SIP Proxy Server: it is an intermediary entity that acts as both a server and a client for the purpose of making requests on behalf of other clients. Mostly it redirects the messages to their destinations. There are several types of proxy

servers defined. A stateful proxy is a proxy that stores server or client transaction state machines. A stateless Proxy only forwards request/response, and do not store any transaction state e.g. CSCF in the IMS architecture.

SIP is a text-based protocol with syntax similar to that of HTTP. There are two different types of SIP messages: requests and responses. The first line of a request has a method, defining the nature of the request, and a Request-URI, indicating where the request should be sent. The first line of a response has a response code.

For SIP requests, RFC 3261 defines the following methods:

(a) REGISTER: Used by a UA to notify its current IP address and the URLs for which it would like to receive calls.

(b) INVITE: Used to establish a media session between user agents.

(c) ACK: Confirms reliable message exchanges.

(d) CANCEL: Terminates a pending request.

(e) BYE: Terminates a session between two users in a conference.

(f) OPTIONS: Requests information about the capabilities of a caller, without setting up a call.

The SIP response types defined in RFC 3261 fall in one of the following categories:

(a) Provisional (1xx): Request received and being processed.

(b) Success (2xx): The action was successfully received, understood, and accepted.

(c) Redirection (3xx): Further action needs to be taken (typically by sender) to complete the request.

(d) Client Error (4xx): The request contains bad syntax or cannot be fulfilled at the server.

(e) Server Error (5xx): The server failed to fulfill an apparently valid request.

(f) Global Failure (6xx): The request cannot be fulfilled at any server.

We can get a vivid picture of how SIP session is set up to transfer media data between two UACs through a list of stateful or stateless SIP proxy servers from Figure 3.12. This figure is taken from outsource Tech-invite web portal (http://www.tech-invite.com/index.html).

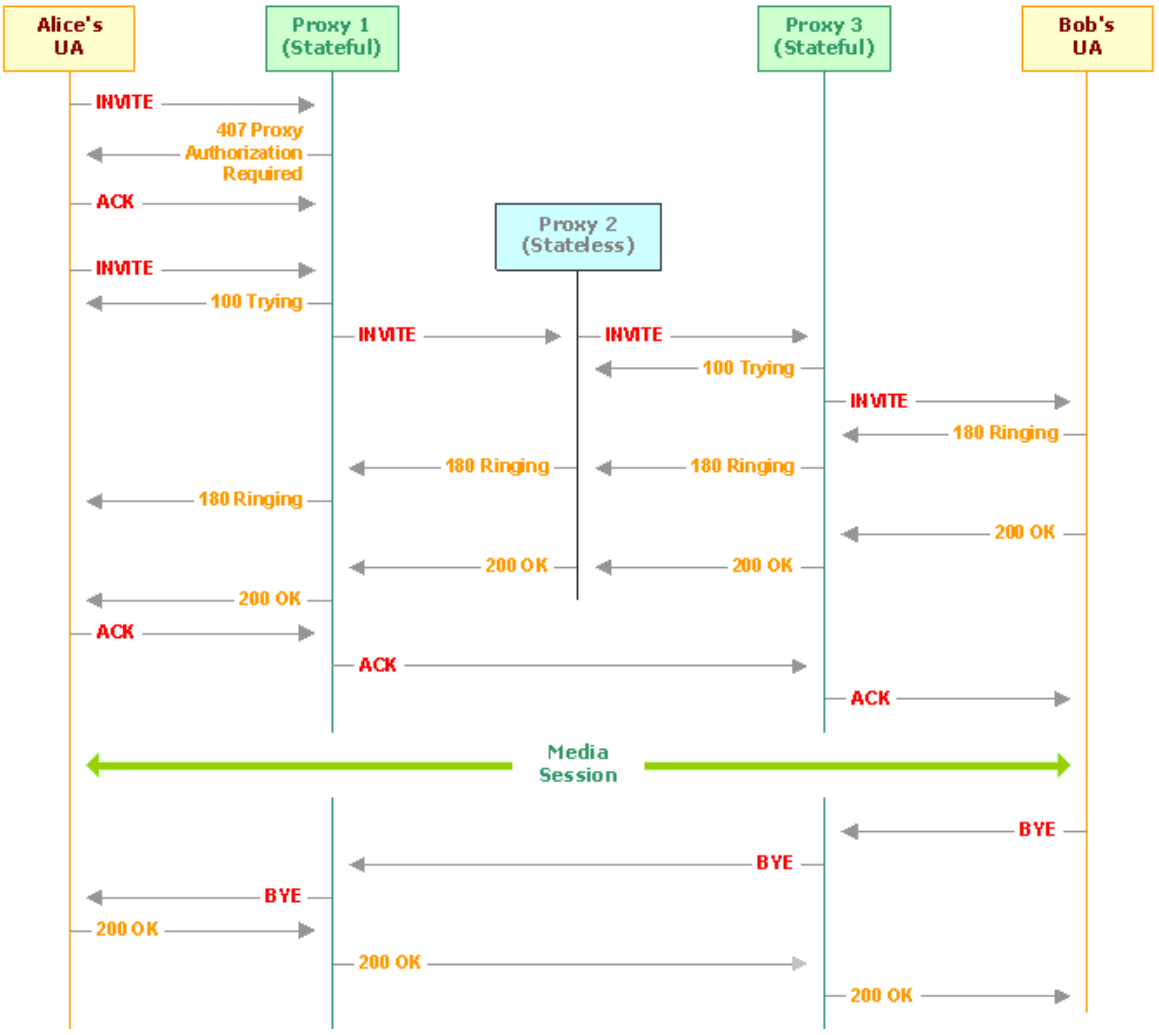

FIGURE 3.12: SIP Signaling Sequence Diagram

2. **How did I implement SIP in this project?**

   (a) SIP signaling flow path realization

   According to the requirements, the system needs two flow paths of SIP session signaling. I have drawn Flow Path 1 which is presented as Figure 3.13 while Flow Path 2 is shown as Figure 3.14.

   SIP Flow Path 1 (see Figure 3.13): In this case, IMS-Client acts as a SIP User Agent Client (UAC) and initiates SIP session by sending SIP INVITE request to CSCF which is a SIP proxy server. CSCF proxies the SIP requests from IMS-Client to VMS that acts as a SIP User Agent Server (UAS). VMS will send corresponding SIP responses to CSCF and CSCF will route these responses back to IMS-Client. After SIP session is set up between IMS-Client and VMS, the RTP media session will initiated between them. As soon as RTP session finishes, IMS-Client will send out SIP BYE request and get SIP 200 OK response. At this moment, SIP session is completed and terminated.

   SIP Flow Path 2 (see Figure 3.14): In this case, IMS-Client still acts as SIP User Agent Client (UAC). It tries to register in CSCF in order to access IMS core network. This is achieved by SIP session between IMS-Client and CSCF. After IMS-Client finishes registration, VMS will get notified about its availability so that it sends SIP Message to IMS-Client about its voice mail box status.

   In conclusion, SIP is most important signaling protocol in the IMS network. It facilitates the session setting up between subscriber, CSCF and AS (Application Server, e.g. VMS in this thesis) in order to transferring voice or video stream via RTP protocol. SIP uses the Session Description Protocol (SDP) [23] to exchange the session content. SDP data is contained in SIP message body and exchanged by session peers (e.g. IMS-Client and VMS in this thesis) to negotiate capacity or else for session establishment purpose. The SIP message body corresponding to SIP Signaling Flow Path 1 (Figure 3.13) and Flow Path 2 (Figure 3.14) are given in the Appendix A.

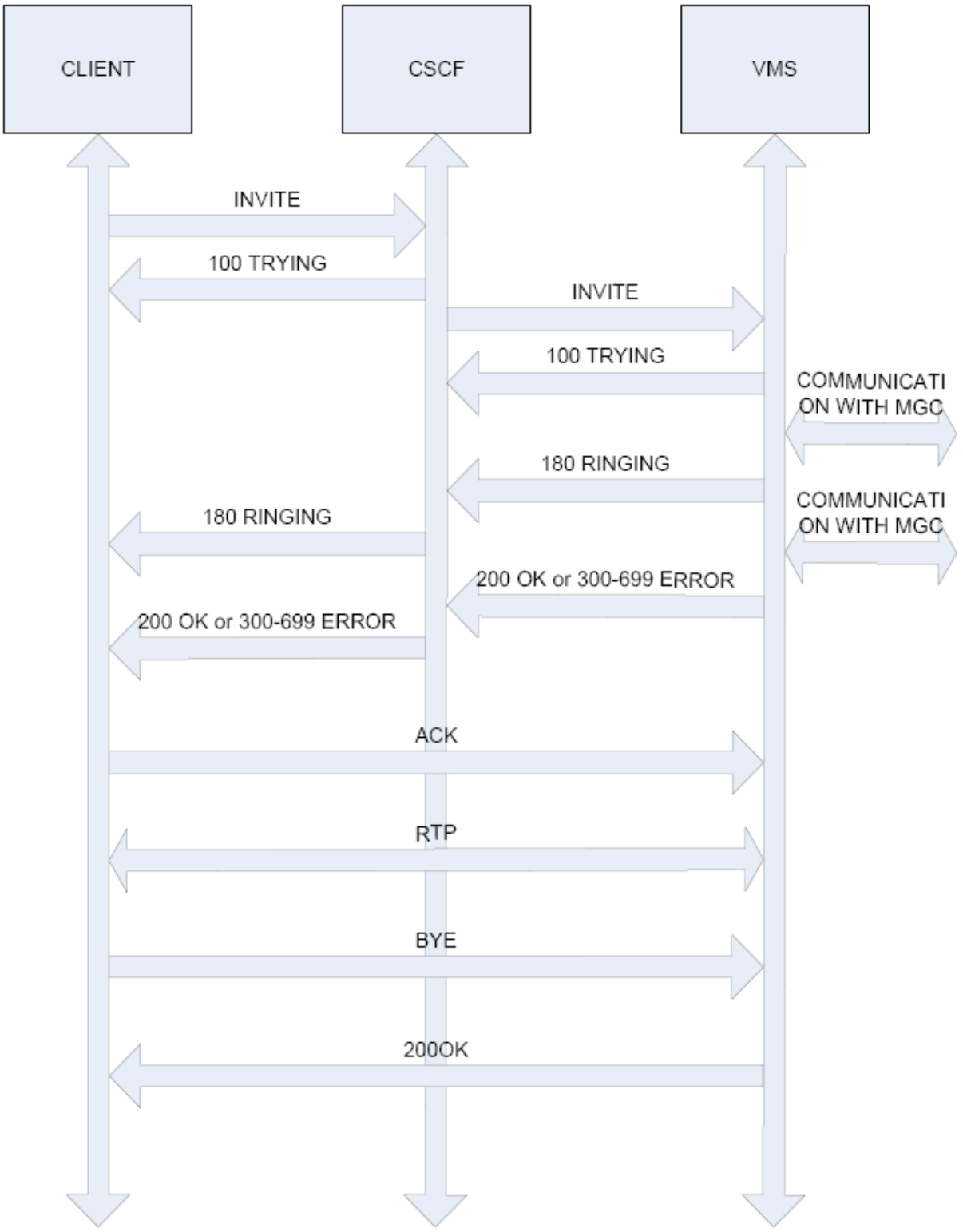

FIGURE 3.13: SIP Signaling Flow Path 1 in IMS Test Environment Simulator

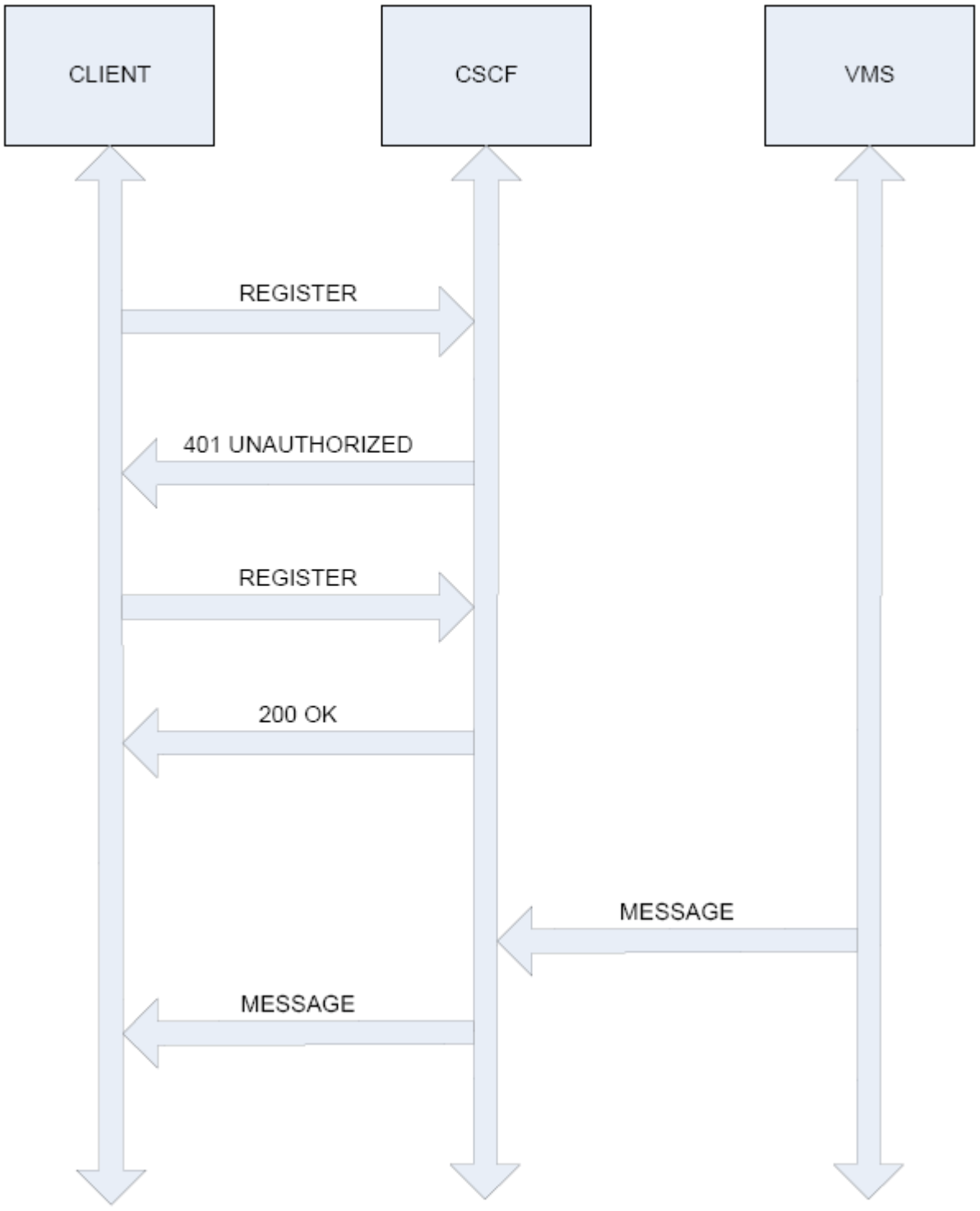

FIGURE 3.14: SIP Signaling Flow Path 2 in IMS Test Environment Simulator

(b) SIP stack implementation

The SIP signaling flow has been described in the last section. In order to enable IMS-Client and CSCF to handle SIP protocol, a SIP stack has to be implemented to meet the system requirements.

Erlang does not have standard library module to handle SIP message parsing and building. Luckily I found an open source project called YXA (http://www.stacken.kth.se/ a SIP software written in Erlang by three persons in KTH (Royal Institute of Technology) in Stockholm. I was inspired by core SIP related codes in YXA and implemented the SIP stack. The SIP stack is composed of three main layers as shown in Figure 3.15 that is taken from external link http://www.docs.hp.com/en/5992-1950/ch01s03.html.

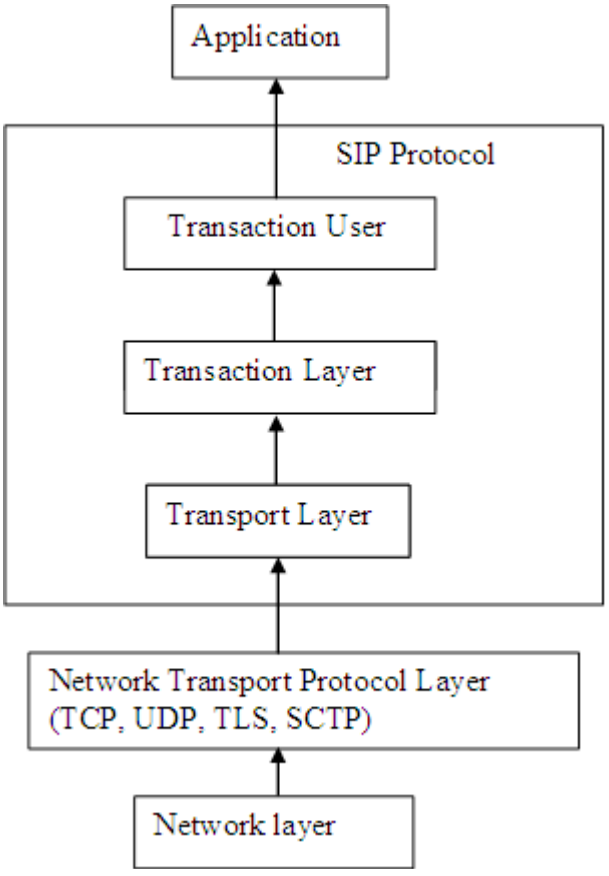

FIGURE 3.15: SIP Protocol Stack Layers

The lowest layer is the transport layer. It defines how a client sends requests and receives responses and how a server receives requests and sends responses over the network. TCP and UDP are mandatory and I did achieve them. Although TLS and SCTP are for secure transportation which is not

very important in this project, I still enabled SIP message to be able to transport via SCTP connection with a user contributed Erlang module for SCTP connection handling.

The second layer is the transaction layer. A transaction is a request sent by a client transaction (using the transport layer) to a server transaction, along with all responses to that request sent from the server transaction back to the client. The transaction layer handles application layer retransmissions and timeouts, and matches responses to requests. Any task that a User Agent Client (UAC) accomplishes takes place using a series of transactions. Stateless proxies do not contain a transaction layer. But in this project, CSCF is supposed to be a stateful SIP proxy server. So I implemented the transaction layer for both CSCF and IMS-Client (which acts as a UAC) to keep track of SIP requests/responses.

The layer above the transaction layer is called the transaction user (TU). A transaction user can be any SIP entity except a stateless proxy. A transaction user uses transactions to send a request to the peer. It creates a client transaction and sends the request, the destination IP address, port number, and transport service to which the request must be sent.

Besides the three layers, a parser for SIP syntax and encoding is also achieved. However the final implemented SIP stack does not support all SIP request and response methods defined in RFC 3261. It only supports SIP request methods such as REGISTER, INVITE, ACK, BYE and MESSAGE together with their corresponding response methods since these are necessary to this project. If further upgrade is needed, this SIP stack can be easily extended to support all defined SIP methods.

3.2.2.2 RTP

1. **What is RTP?**

   Real-time Transport Protocol (RTP) [5] provides end-to-end delivery services for data with real-time characteristics, such as interactive audio and video. Those services include payload type identification, sequence numbering, time stamping and delivery monitoring. Applications typically run RTP on top of UDP to make use of its multiplexing and checksum services; both protocols contribute parts of the transport protocol functionality.

   RTP itself does not provide any mechanism to ensure timely delivery or provide other quality-of-service (QoS) guarantees, but it is usually used in conjunction with the RTP Control Protocol (RTCP). While RTP carries the media streams (e.g.,

audio and video) or out-of-band signaling (DTMF), RTCP is used to monitor transmission statistics and quality-of-service (QoS) information. RTP does not guarantee delivery or prevent out-of-order delivery, nor does it assume that the underlying network is reliable and delivers packets in sequence. The sequence numbers included in RTP allow the receiver to reconstruct the sender's packet sequence, but sequence numbers might also be used to determine the proper location of a packet, for example in video decoding, without necessarily decoding packets in sequence. The RTP packet format is presented in Figure 3.16 below which is taken from external link.(http://www.cl.cam.ac.uk/jac22/books/mm/book/node159.html)

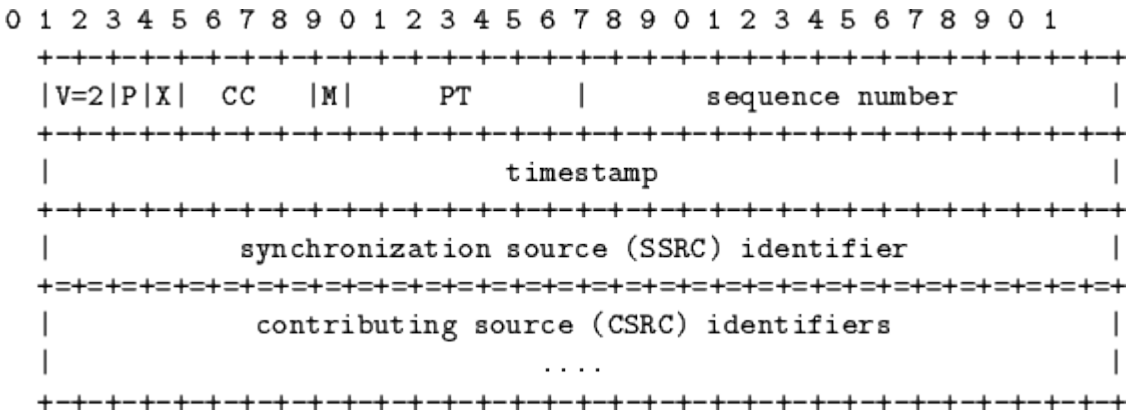

FIGURE 3.16: RTP Packet Format

2. **How did I implement RTP in this project?**

   According to RTP packet format and transportation mechanism, I implemented a single module to parse and build RTP packets and use UDP to transport them. Erlang supports UDP communication well so that it facilitates my implementation. RTP is relatively easy to implement compared to SIP mentioned in last section. It only needs to matter with transport and application layers. When the IMS-Client application receives RTP packets stream from VMS, it will parse them and reconstruct according to the sequence number in each packet to build up a complete voice mail.

   Although RTCP is used together with RTP to monitor transmission statistics and quality-of-service (QoS) information, it is not mandatory in the scope of this thesis project. It can be an extra feature in the future system upgrade.

3.2.2.3 Diameter (Cx and Sh interface)

1. **What is Diameter?**

Diameter [6] is an Authentication, Authorization and Accounting (AAA) protocol developed by the IETF. Diameter is used to provide AAA services for a range of access technologies. Instead of building the protocol from scratch, Diameter is loosely based on the Remote Authentication Dial In User Service (RADIUS), which has previously been used to provide AAA services, at least for dial-up and terminal server access environments.

The Diameter protocol is actually split into two parts: Diameter Base Protocol and Diameter Applications. The Diameter Base Protocol defines the minimum requirements for an AAA protocol. Diameter Applications can extend the base protocol by adding new commands and/or attributes. A Diameter Application is not a program but a protocol based on Diameter. Diameter uses both TCP and SCTP as transport. Diameter is a peer-to-peer protocol since any Diameter node can initiate a request. Diameter has three different types of network nodes: clients, servers and agents.

The Diameter message consists of a Diameter header, followed by a certain number of Diameter Attribute Value Pairs (AVPs). The Diameter header comprises binary data and as such is similar to an IP header or a TCP header. The format of the Diameter header is shown in Figure 3.17 which is taken from external link.(http://en.wikipedia.org/wiki/Diameter)

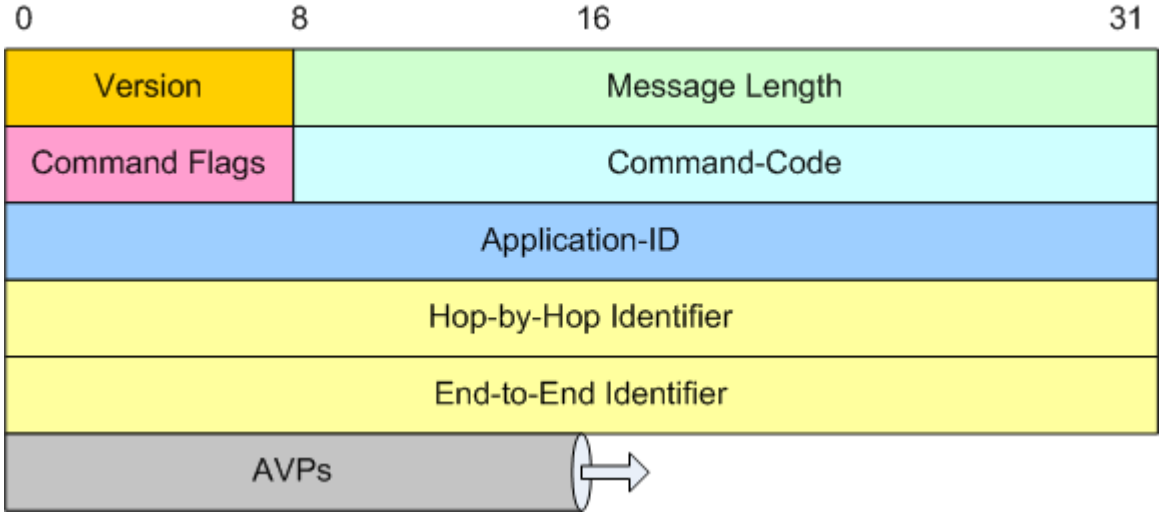

FIGURE 3.17: Diameter Header Format

The Diameter Attribute Value Pairs (AVPs) contain authentication, authorization and accounting information elements, as well as routing, security and configuration information elements that are relevant to the particular Diameter request or answer message. Each AVP contains an AVP header and some AVP-specific

data. The AVP header is shown in Figure 3.18 which is taken from external link.(http://en.wikipedia.org/wiki/Diameter)

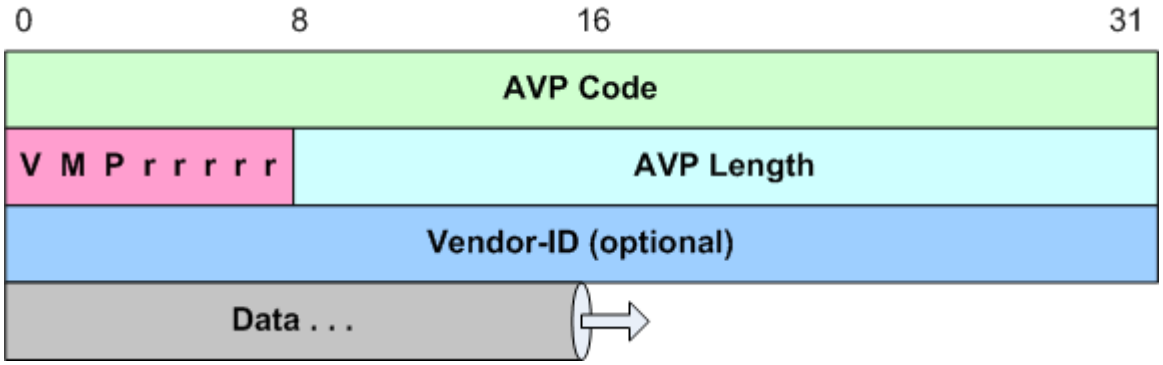

FIGURE 3.18: Diameter AVP Header Format

Diameter is selected to provide AAA service in IMS network. There are quite a few of Diameter applications used in IMS. However, the one that is closely related to this project is the Diameter Session Initiation Protocol (SIP) application [Draft-ietf-aaa-diameter-sip-app], which is used in the Cx, Dx, Sh and Dh interfaces. The interfaces Cx and Sh are used in this project and will be described in the following section.

2. **How did I implement Diameter in this project?**

   In this project scope, Diameter is the protocol used between HSS and entities such as CSCF and VMS.

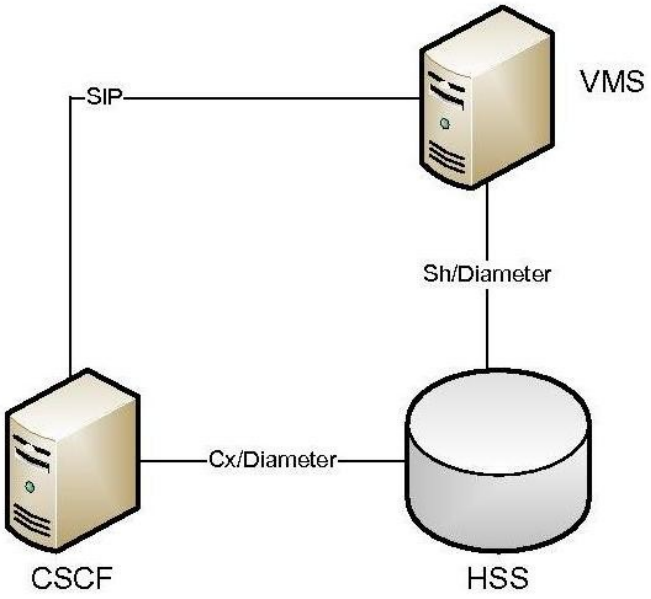

FIGURE 3.19: HSS Diameter Cx and Sh Interfaces

In IMS architecture, the Cx interface is used communicate between CSCF and HSS while the Sh interface is used to exchange information between SIP AS and HSS. Both interfaces use Diameter protocol. The VMS is acting as a SIP AS in this case. I have drawn Figure 3.19 to show a clear view of this architecture.

HSS is the master user database in IMS core network which stores subscriber profile and service data. When the user tries to access IMS network, the user has to register in CSCF first. So CSCF communicates HSS via Cx Diameter interface to download user profile data for authentication and authorization. The corresponding Diameter request/response pairs for this procedure are MAR/MAA and SAR/SAA (See Table 3.1). VMS has to be notified when the user becomes available in IMS network in order to inform his voice mail box status. Before the user accesses IMS network, VMS will subscribes notification of the user status change. Then as soon as the user finishes registration in CSCF, user status will be changed in HSS and HSS will send notification to VMS. This procedure is fulfilled by Diameter request/response pairs SNR/SNA and PNR/PNA (See Table 3.1). The whole Diameter message flow for these two cases is shown in Figure 3.20. Table 3.1 lists out the Diameter Commands [24] used in this project. The full list of Diameter Commands is given in Appendix B.

Table 3.1: Diameter Commands (partial)

Command Name	Abbreviation	Command Code
Server-Assignment-Request	SAR	301
Server-Assignment-Answer	SAA	301
Multimedia-Auth-Request	MAR	303
Multimedia-Auth-Answer	MAA	303
Subscribe-Notifications-Request	SNR	308
Subscribe-Notifications-Answer	SNA	308
Push-Notification-Request	PNR	309
Push-Notification-Answer	PNA	309

As we can see in Figure 3.20 which I have drawn, VMS sends SNR to HSS in order to subscribe the notification of the IMS-Client's availability. HSS replies to VMS with SNA. After that, the IMS-Client tries to get access to IMS network by sending SIP Register message to CSCF. Then CSCF sends MAR to HSS to request user authentication data. HSS will reply MAA containing required data. After IMS-Client gets registered in CSCF, CSCF will send SAR to HSS to change user status in HSS and get SAA response from HSS. Since VMS has already subscribed notification, HSS will then send PNR to deliver notification of IMS-Client availability to VMS. VMS confirms by sending PNA response. Then VMS could do further operation on informing IMS-Client about new voice mail arrived

or other relevant information through SIP message. This is a complete procedure of how HSS interacting with CSCF and VMS via Diameter protocol to provide subscriber authentication and availability notification service.

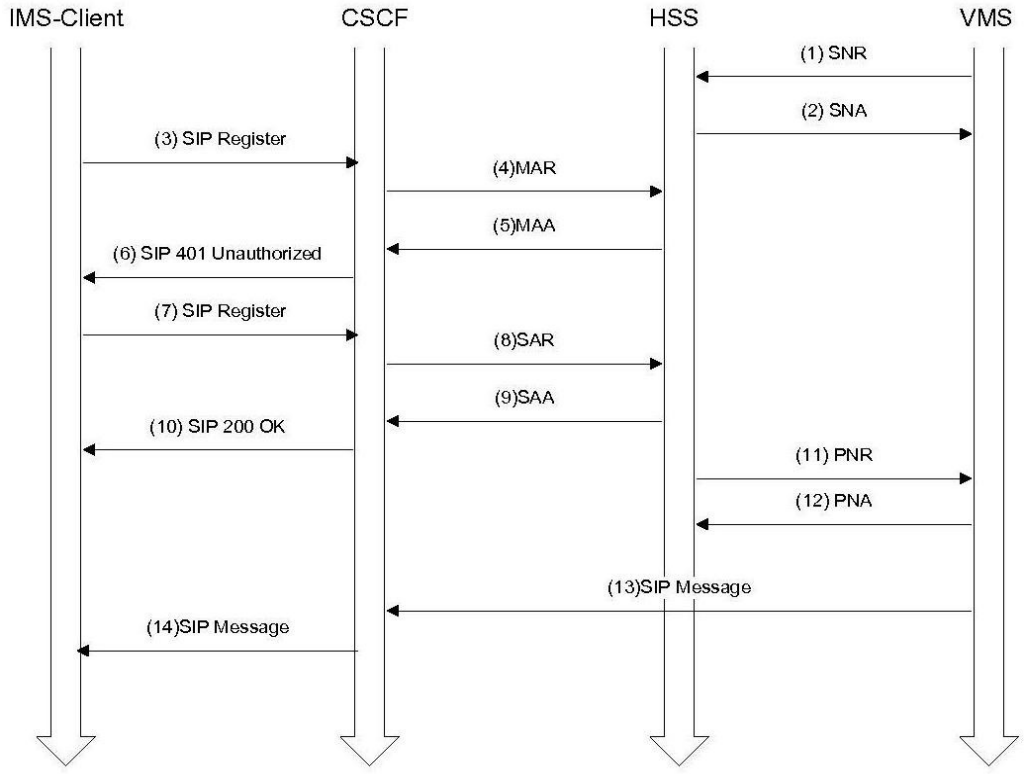

FIGURE 3.20: HSS Diameter Message Flow (SNR/SNA, PNR/PNA, MAR/MAA, SAR/SAA)

Both Cx and Sh interfaces are implemented in HSS to handle Diameter requests from CSCF and VMS. Cx interface in CSCF is also realized. CSCF is able to contact with HSS using MAR/MAA and SAR/SAA Diameter commands for user registration and authentication. The external VMS enables Sh interface so that it can communicate with HSS through Sh/Diameter. HSS is able to send notification to VMS about user status.

The implemented Diameter stack is functional to handle the 8 Diameter commands listed out in Table 3.1 which are applied in this project. It is able to parse and build Diameter message packet and transport them on top of TCP and SCTP. The stack is fully extensible to support more commands listed in Appendix B in future upgrade.

3.2.3 System Testing and Integration

Unit testing is a basic test on individual units of source code in order to make sure they are functional and fit for use. A unit is the smallest testable part of an application. Erlang is a functional programming language which is used to develop this project. So a unit in Erlang is an individual function. Each single function is well tested by feeding both expected and unexpected argument values to make sure the function can handle them properly without crashes. The unit testing is carried out as soon as the author finishes implementing a single Erlang function.

As described in system design and implementation in previous sections in this chapter, the main components such as IMS-Client, CSCF and HSS are implemented as Erlang applications. An Erlang application consists of several modules each of which contains a bunch of Erlang functions. The Erlang application is a component that can be started and stopped as a unit and which can be re-used. Thus, functional tests are conducted on these stand-alone application components of the system to ensure that they meet the essential requirements listed out in section 3.2.1 on each component. The actual used testing methods are to prepare different test cases according to each single requirement and perform them by calling relevant functions in the application. Each component application in the system operates functionally.

Although stand-alone system components such as IMS-Client, CSCF and HSS are running normally, the protocol interfaces between them need to be tested and verified as well. This is very important since the key goal of this project is to create a test environment for testing IMS application server (i.e. VMS in this project) which based on several primary protocols such as SIP, RTP and Diameter. The test environment simulator must be able to communicate with the target system VMS via these protocols. So the protocol stacks implemented in IMS-Client, CSCF and HSS are tested properly to make sure they can build correct protocol messages to send and parse the received protocol messages. When testing the protocol stacks, not only correct messages but also incorrect messages such as messages with syntax errors are provided to check if the stack is fault-tolerant and handles error case well. The result indicates that SIP, RTP and Diameter stacks implemented in the project behave in an acceptable manner and conform to the standards. However, only a certain part of all protocol messages which are necessary in this project are verified. All the rest unsupported protocol messages are skipped.

In conclusion, all system components are well tested and meet the system requirements. The next step will be to integrate them into a complete IMS test environment simulator system.

After system testing which is conducted on each separated system component (IMS-Client, CSCF and HSS), it should integrate them to build up a complete IMS test environment targeted on testing the external VMS system. Therefore, the first integration stage should be system internal integration which is clearly revealed as the left part in Figure 3.21 which I have drawn.

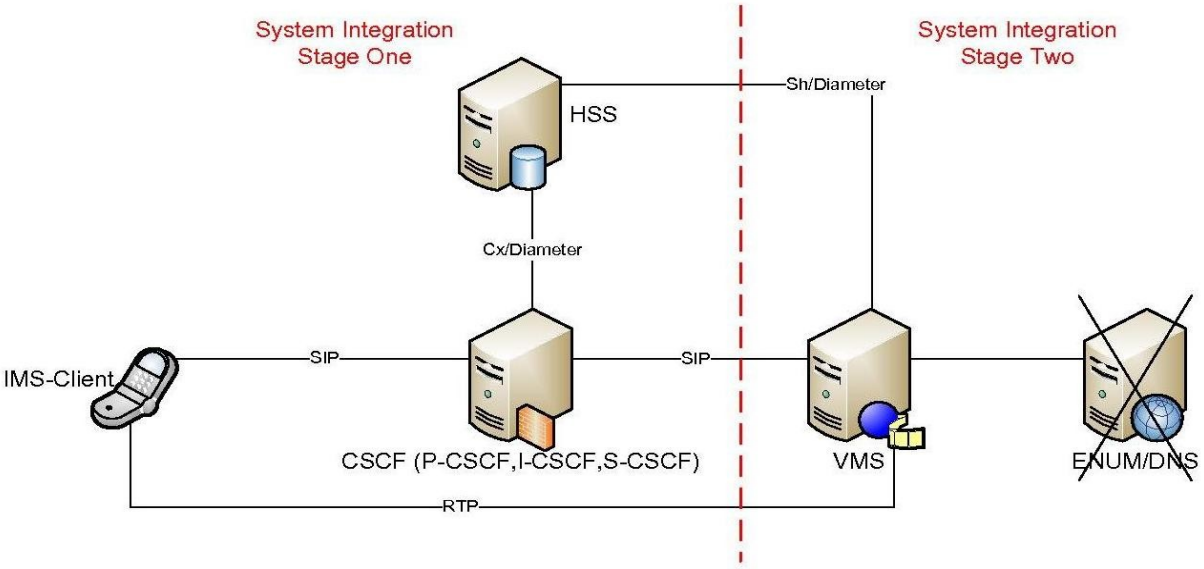

FIGURE 3.21: IMS Test Environment Simulator System Internal Integration

First stage is to integrate system components into a functional IMS test environment simulator. This includes configuring IMS-Client, CSCF and HSS on their listening ports, setting up TCP connection between IMS-Client and CSCF as well as between CSCF and HSS. After TCP connections are up and running, IMS-Client initiates SIP session towards CSCF. This is fulfilled by several SIP messages such as REGISTER, 401 UNAUTHORIZED and 200 OK exchanged between IMS-Client and CSCF. Meanwhile, CSCF will automatically communicate with HSS through Cx/Diameter interface in order to download user authentication data from HSS database. For this purpose, Diameter commands pairs MAR/MAA and SAR/SAA are used between HSS and CSCF. During this stage, not only each system components internal service logic is tested and verified but also the protocol stacks and interfaces between each other are formally tested. This system integration testing will reveal some serious bugs and even system design flaws. It helps the author to reconsider design on some component and correct the found errors before entering system integration stage two which indicates external integration with VMS system. External integration belongs to system verification aspect which will be discussed in the next chapter about evaluation.

Chapter 4

Evaluation

One of the two goals of this thesis is to provide a test environment targeted to a third-party thesis project regarding an IMS Voice Mail System (VMS). This means it is necessary to set up different functional entities in the IMS network to be able to verify SIP/RTP/Diameter signaling or media traffics with VMS system. This verification of VMS functionality with the help of the IMS test environment simulator is the proper way to evaluate the simulator itself. Thus this chapter will describe the process of integrating the IMS test environment simulator with external VMS system and carry out test cases to verify VMS basic functionality.

Evaluation of the IMS Test Environment Simulator system is mainly based on system verification by means of external integration with VMS system. Since it targets on testing an IMS Voice Mail Server (VMS) system, a complete test scenario has been performed which proves the simulator works as expected.

4.1 System Verification

At second stage in system integration, the main task is changed from internal testing and integration to verify the system by means of integrating it with external VMS system. After the test environment simulator is ready with all three nodes IMS-Client, CSCF and HSS up running and be able to handle SIP, RTP and Diameter messages, it is time to fulfill the goal of this project which is to test targeted on an IMS Voice Mail Server (VMS) system. As described in previous chapters, the basic functionalities of the VMS system include: (1) communicates with HSS to subscribe notification of user's availability via Sh/Diameter interface; (2) sets up SIP session with the user (i.e. IMS-Client) when the user tries to deposit or retrieve voice mails from the VMS via RTP connection in between.

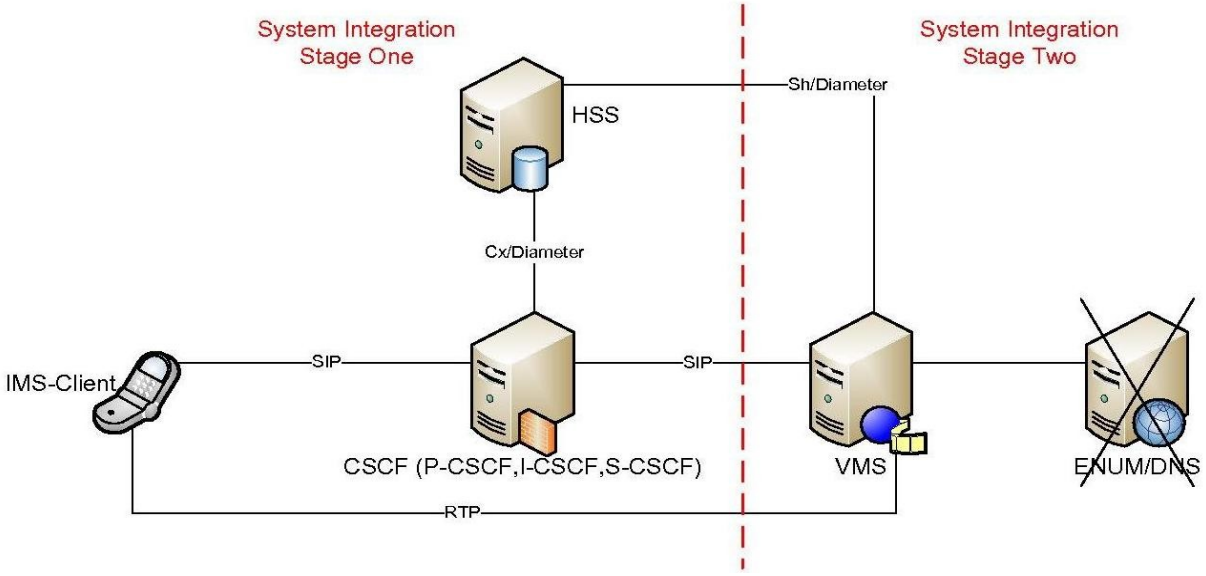

FIGURE 4.1: IMS Test Environment Simulator System External Integration

As shown in Figure 4.1, HSS and CSCF are configured to set up TCP connection between external VMS system. Then tests are performed according to the three traffic scenarios (or user cases) mentioned in chapter 3.1.3.

In first user case, external VMS will send Subscribe-Notification-Request (SNR) to HSS to register notification of the IMS-Client availability. HSS successfully saves a trigger on corresponding user data record in the database and sends Subscribe-Notification-Answer (SNA) back to VMS. This is verified by checking HSS internal trigger table as well as whether VMS gets correct SNA message. The result is positive. Next step is making IMS-Client becomes on-line. This procedure is done by setting up SIP session between IMS-Client and CSCF for registration. CSCF will communicate with HSS downloading authentication data about IMS-Client. This procedure is verified by monitoring IMS-Client, CSCF and HSS at the same time to make sure they all send and receive correct and expected messages and user state of IMS-Client is changed in HSS database ultimately. In HSS, the trigger previously set on user state fires and sends Push-Notification-Request (PNR) to VMS and gets expected Push-Notification-Answer (PNA) from VMS. This is verified by VMS getting the correct PNR with IMS-Client's user profile data. As shown in Figure 4.1, ENUM/DNS is wiped out because it is not simulated in this project due to time limitation and it is not a critical requirement for a test environment. In real IMS network, the ENUM/DNS server is for mapping subscriber's SIP address or E.164 [25] number to a physical IP address so that Application Server such VMS can request the subscriber's IP address in order to deliver SIP message directly to it. Therefore in this project, after VMS gets notified about IMS-Client becoming on-line, VMS simply delivers SIP message directly towards the IMS-Client with

its IP address pre-configured in VMS. On IMS-Client side, it successfully receives the SIP message sent from VMS.

Second and third user cases, the test procedure are similar. The only difference is that depositing a voice mail is transferring voice data from IMS-Client towards VMS while retrieving a voice mail is transferring voice data from VMS to IMS-Client. So let us only emphasis on discussing the procedure of depositing a voice mail into VMS. IMS-Client initiates a SIP INVITE message and sends it to CSCF. As soon as CSCF receives this message, it will proxy it immediately to external VMS. After a while, VMS will send back a SIP 200 OK message to CSCF if VMS operates properly on the SIP INVITE message. (It could be possible that VMS returns SIP error message other than 200 OK if the SIP INVITE message initiated by IMS-Client contains wrong data. If so, it should re-check the IMS-Client application to ensure it to send correct SIP message.) Therefore CSCF will proxy SIP 200 OK back to IMS-Client and SIP session between IMS-Client and VMS has been set up at this moment. Next step is to transfer voice mail data from IMS-Client to VMS via RTP session. The SIP messages exchanged between IMS-Client and VMS at the beginning contains SDP data which describes all necessary parameters for setting up a media session. For example, it tells IMS-Client about VMS UDP port and some other useful arguments. With these values, IMS-Client tries to connect VMS to set up a RTP session in between. After RTP session is established, IMS-Client transfer pre-stored voice mail data to VMS. As soon as all data transportation is done, IMS-Client will send SIP BYE message to CSCF and CSCF will proxy it to VMS. VMS gets it and becomes aware that all voice mail data is transferred, it will terminate the RTP session and send final SIP 200 OK message back to IMS-Client via CSCF. This will successfully end the whole SIP session. This procedure is verified by monitoring IMS-Client, CSCF and external VMS on their behaviors and sent/received SIP messages and RTP packages.

All mentioned three test scenarios above are well conducted by the author with the help of the developers of external VMS system and the final outcomes are satisfactory by means that: (1) SIP session between IMS-Client, CSCF and VMS as well as RTP session between IMS-Client and VMS can be successfully set up. The SIP stack in IMS-Client and CSCF operates normally on building and parsing SIP messages exchanged with external VMS. (2) RTP stack in IMS-Client can build up RTP package containing media data to send to VMS as well as parse RTP package from VMS and extract the media data included. (3) HSS Cx and Sh interfaces for CSCF and VMS work as expected. Diameter stack in HSS is able to build and parse Diameter Command messages correctly. HSS can manage the user profile database and keep track of triggers on the user data for notification purpose.

In conclusion, all mandatory and essential features of the IMS test environment simulator are accomplished and meet the demand of testing the external VMS system. Besides these integration tests mentioned above which carried out traffic cases defined in chapter 3.1.3, a possible additional test could be stress testing (or called performance testing). The functionality to perform this stress testing is already implemented in IMS test environment simulator. But unfortunately due to time limitation of the VMS thesis project, stress testing was not able to carry out in the end. Thus I have put this part in future work described in chapter 5.

Chapter 5

Future Work

Although the final prototype of IMS test environment simulator meets all essential requirements described in chapter 3 and succeeds in testing the external VMS system, there are still further improvements that could be made for the prototype in the future. The unimplemented features and limitations of the system prototype will be pointed out. Future work on these issues will be necessary if the system is planned to be evolved and put into practical use in the sponsored company. Generally speaking, the future work may include the following three aspects:

1. **Optional Features Add-On**

   Besides the essential requirements, there are some optional features. One is to simulate ENUM/DNS entity in the test environment. With ENUM/DNS simulation, it can support external VMS to lookup for certain subscriber's IP address in order to send direct SIP MESSAGE. It is a text message. So VMS can use it to notify the subscriber when a new voice mail coming in. In addition, internal CSCF can query ENUM/DNS about VMS location in the network. So it could proxy SIP packet to VMS when subscriber tries to set up a SIP session with VMS. In current prototype, ENUM/DNS simulation is not achieved. Instead, configure IMS-Client, CSCF and VMS to be aware of each other's IP address information. But in real IMS network, the entities must query ENUM/DNS in order to get each other's IP address or other relevant information. Thus, ENUM/DNS simulation should be added into the prototype in the future.

   Another optional feature is to create a GUI (Graphic User Interface) for the system. It facilitates operating the system. Instead of starting system nodes such as IMS-Client, CSCF and HSS separately by calling functions through Erlang shell, one can start/stop/restart them easily through GUI. In addition, more functionality

can be added into GUI such as system re-configuration, display or dump system log, enable test cases and gather test results into specific file, display diagram generated by the test results and etc. With the GUI, it will be much easier for the potential user (could be a tester or developer) to set up this IMS test environment simulator and carry out test cases towards target system. (e.g. VMS or other IMS based application server) In future upgrade, a full functional GUI will be very necessary if planning to put the system prototype into practical use.

2. **Interface and Protocol Stack Improvement**

   In the current prototype, SIP, RTP and Diameter are used to communicate between internal system components as well as with external target system VMS. IMS-Client needs to set up SIP session with VMS via CSCF as a SIP proxy. The implemented SIP stack only support a small subset of SIP methods such as: INVITE, ACK, BYE, REGISTER and MESSAGE. These are enough to set up SIP session to meet the current requirements. But for future system extension, the SIP stack should support more SIP methods defined in its protocol standard (RFC 3261) because this IMS test environment simulator is intended to test different IMS based application server not limited to VMS in future. A more mature SIP stack which can handle all standard SIP messages should be achieved in the following development. The current SIP stack is modular structured and organized so that it can be easily extended.

   The two Diameter interfaces supported in HSS are Cx and Sh. Cx/Diameter is between HSS and CSCF while Sh/Diameter is between HSS and VMS. These two interfaces are basically implemented only to meet the system requirements. So the Diameter protocol stack is not fully implemented. However, in further enhancement, the Diameter stack can be extended to support the full list of Diameter commands presented in Appendix B. This is quite important because it is very likely to add more entities into the test environment simulator and these entities will need to communicate with HSS using different Diameter commands unsupported at the moment. Cx and Sh Diameter interfaces can also be improved to be able to handle more Diameter commands which can provide CSCF and VMS with more AAA services other than current registration and notification mentioned in previous chapters.

   Furthermore, the media data between IMS-Client and VMS is carried by RTP packets. The underlying transport protocol is UDP which does not guarantee reliability, ordering, or data integrity. Thus, RTCP is compliment control protocol for RTP which intends to ensure quality-of-service (QoS) for RTP session. RTCP provides out-of-band statistics and control information for an RTP flow. It partners RTP in the delivery and packaging of multimedia data. It provides feedback on

the quality-of-service (QoS) in media distribution by periodically sending statistics information to participants in a streaming multimedia session. So in the future development, RTCP stack should be implemented and enables RTCP messages to be transmit together with RTP packets during the RTP session in order to guarantee QoS.

3. **Performance Testing on Target System**

   Although the system testing is well performed as described in Chapter 4, there are still something left untested which is performance test of external VMS system. This test is intended to test the performance of VMS by measuring its characteristics. It includes counting the response time of handling a single test call, maximal number of ongoing simultaneous call sessions, time variation to the increment of test calls and etc. Due to time limitation of both VMS project and this project, this performance testing is skipped at the moment. However, since the module to conduct this kind of test is done in IMS test environment simulator, it is possible to do further testing on VMS system performance in the future when new requirements are made.

Chapter 6

Conclusion

In this last chapter of the paper, I will at first discuss the academic challenges I had faced during the project. Then a summary on the overall achievement of the project goal addressed in Section 2.1.1 will be given.

6.1 Academic Challenges

During the research, design and development phases of the whole project, I faced a few of academic challenges which could be interesting to point out. They are:

- **Erlang language and system:**

  The thesis project uses Erlang functional programming language which is specific to concurrent and distributed programming. In addition to being comfortable using it, it is necessary for me to dig deeper into Erlang and OTP (Open Telecom Platform) which is the development environment and library for Erlang concurrent programming. This study took me a big amount of time but it was really beneficial to me. It helped me to build the system prototype according to standard design principle. The achieved prototype is compatible and easy to maintain and further develop based on it.

- **Telecom software system design and development:**

  The thesis is specific to Telecom industry with focus on IMS (IP Multimedia Subsystem) which is supposed to be the next generation Telecom mobile network system. So what I should do is to study and understand most of the IMS architecture concepts so as to be able to design and implement my IMS Test Environment

Simulator. This effort is very time-consuming at the beginning of the project research because I am totally new to this area. But it paid off as the project goes on. The final outcome as a working prototype is the best prove. For myself, I gained a lot from studying IMS concepts to build up knowledge of the telecommunication network.

- **Protocols implementation:**

  According to the requirements of the thesis project, the system should support quite a few of network protocols such as SIP, RTP and Diameter. They are the primary protocols applied in the IMS network architecture. I needed to digest and fully understand these network protocols to implement them with Erlang code. The most tricky and complex one among them goes to the SIP protocol. It took me quite a long time to implement SIP stack and make it functional.

- **Test environment simulator:**

  What and How to simulate the whole IMS environment for testing purpose is the key point of my thesis research. I had no preliminary knowledge on software testing and simulation before this thesis. So I was learning by doing which turned out to be really good for me.

- **Project management:**

  Although the thesis project was a single-person project, I was trying to make its development procedure as formal and professional as possible. So I used some of the main project management methods in software engineering area. For example, all the necessary documentations for project management were written and SVN was used to maintain all the source code etc. Project management was a little bit tedious but really important and valuable in keeping everything on the track. In addition, this makes maintenance and further development much easier.

- **Miscellaneous:**

  Forming a proper programming habit based on coding convention was another request of the thesis project. Besides that, the project required a lot of standards specifications from 3GPP and IETF organizations. So it was critical to implement the system strictly following the standards for compatibility and extension in future.

6.2 Summary

I have described all aspects of my thesis project - IMS Test Environment Simulator. A complete simulation of all entities and all interfaces with full functionality in the IMS core network demands a huge amount of work and not worthwhile when the goal is to set up a test environment for testing IMS based application server. Therefore the project scope is narrowed down to a reasonable complexity level so as to target on testing a Voice Mail System (VMS). Thus the final implemented prototype simulates only the key entities in IMS core network architecture such as CSCF and HSS as well as a IMS-Client application which acts as a subscribers handset. Real CSCF consists of three different kinds which are P-CSCF, I-CSCF and S-CSCF. In order to simply the implementation, these three are combined into one general simulated CSCF which has the same functionality. Primary network protocols applied in IMS network such as SIP, RTP and Diameter are supported in the simulator. They are used to communicate between entities inside the simulator as well as outside with target VMS system. These protocol stacks implementation gained me a lot of insights into the IMS world.

The final system integration and testing outcome reveals that the prototype has met all essential requirements and it works as expected to test the external VMS system. It is able to set up SIP session between IMS-Client, CSCF and target VMS. Media data can be transported via RTP connection between IMS-Client and VMS on depositing or retrieving voice mail. The HSS can notify VMS when IMS-Client gets connected. These help testing the VMSs basic functionalities. Furthermore, the simulator has special modules to test VMSs performance by initiating a great amount of simultaneous sessions during a specified period and record VMSs characteristics. Although it lacks time to do the performance tests on VMS, the feature is already built in the simulator.

The current prototype has functionality limitation because it is specifically built targeted on testing a VMS prototype which only supports basic services. As mentioned in last chapter about future evolution, this test environment simulator prototype can be extended a lot by supporting more protocols and interfaces, further development on the protocol stacks, simulating more entities in the IMS network and etc. The prototype is modular organized so that extension is not that hard but very time consuming.

As a telecommunication service provider company such as Mobile Arts AB that initiates this thesis project, it is of great interest to implement IMS based application servers in the future. However the cost of accessing real IMS core network only in order to test those IMS based application server is not paid off. So an IMS test environment simulator for testing purpose is really necessary. Hopefully the prototype achieved in this thesis project can bring benefits and inspirations.

Appendix A

Glossary

- **3GPP**

  The 3rd Generation Partnership Project (3GPP) is a collaboration between groups of telecommunications associations, to make a globally applicable third generation (3G) mobile phone system specification within the scope of the International Mobile Telecommunications-2000 project of the International Telecommunication Union (ITU). 3GPP specifications are based on evolved Global System for Mobile Communications (GSM) specifications. 3GPP standardization encompasses Radio, Core Network and Service architecture.

  The groups are the European Telecommunications Standards Institute, Association of Radio Industries and Businesses/Telecommunication Technology Committee (ARIB/TTC) (Japan), China Communications Standards Association, Alliance for Telecommunications Industry Solutions (North America) and Telecommunications Technology Association (South Korea). The project was established in December 1998.

- **Cx**

  A Diameter protocol interface between HSS and CSCF.

- **CSCF**

  Several roles of Session Initiation Protocol (SIP) servers or proxies, collectively called Call Session Control Function (CSCF), are used to process SIP signaling packets in the IMS.

- **Diameter**

  A computer networking protocol for AAA (Authentication, Authorization and Accounting). The Diameter Base Protocol is defined by RFC 3588, and defines the

minimum requirements for an AAA protocol. Diameter Applications can extend the base protocol, by adding new commands and/or attributes. An application is not a program, but a protocol based on Diameter. Diameter security is provided by IPSEC or TLS, both well-regarded protocols.

- **DNS**

  On the Internet, the Domain Name System (DNS) associates various sorts of information with so-called domain names; most importantly, it serves as the phone book for the Internet by translating human-readable computer host names, e.g. en.wikipedia.org, into the IP addresses, e.g. 66.230.200.100, that networking equipment needs for delivering information. It also stores other information such as the list of mail exchange servers that accept email for a given domain. In providing a worldwide keyword-based redirection service, the Domain Name System is an essential component of contemporary Internet use.

- **ENUM**

  Telephone Number Mapping, is a suite of protocols to unify the telephone numbering system E.164 with the Internet addressing system DNS by using an indirect lookup method, to obtain NAPTR records. The records are stored at a DNS database.

- **HSS**

  The Home Subscriber Server (HSS), or User Profile Server Function (UPSF), is a master user database that supports the IMS network entities that actually handle calls. It contains the subscription-related information (user profiles), performs authentication and authorization of the user, and can provide information about the users physical location. It is similar to the GSM Home Location Register (HLR) and Authentication Centre (AUC).

- **IETF**

  The Internet Engineering Task Force (IETF) develops and promotes Internet standards, cooperating closely with the W3C and ISO/IEC standard bodies; and dealing in particular with standards of the TCP/IP and Internet protocol suite. It is an open, standards organization, with no formal membership or membership requirements. All participants and leaders are volunteers, though their work is usually funded by their employers or sponsors; for instance, the current chairperson is funded by VeriSign and the U.S. governments National Security Agency.

- **IMS**

  The IP Multimedia Subsystem (IMS) is an architectural framework for delivering Internet Protocol (IP) multimedia to mobile users. It was originally designed

by the wireless standards body 3rd Generation Partnership Project (3GPP), and is part of the vision for evolving mobile networks beyond GSM. Its original formulation (3GPP R5) represented an approach to delivering Internet services over GPRS. This vision was later updated by 3GPP, 3GPP2 and TISPAN by requiring support of networks other than GPRS, such as Wireless LAN, CDMA2000 and fixed line.

- **RTP**

  The Real-time Transport Protocol (or RTP) defines a standardized packet format for delivering audio and video over the Internet. It was developed by the Audio-Video Transport Working Group of the IETF and first published in 1996 as RFC 1889 which was made obsolete in 2003 by RFC 3550. Real time transport protocol can also be used in conjunction with RSVP protocol which enhances the field of multimedia applications.

  RTP does not have a standard TCP or UDP port on which it communicates. The only standard that it obeys is that UDP communications are done via an even port and the next higher odd port is used for RTP Control Protocol (RTCP) communications. Although there are no standards assigned, RTP is generally configured to use ports 16384-32767. RTP can carry any data with real-time characteristics, such as interactive audio and video. Call setup and tear-down is usually performed by the SIP protocol. The fact that RTP uses a dynamic port range makes it difficult for it to traverse firewalls. In order to get around this problem, it is often necessary to set up a STUN server.

- **Sh**

  A Diameter protocol interface between HSS and AS (Application Server).

- **SIP**

  The Session Initiation Protocol (SIP) is an application-layer control (signaling) protocol for creating, modifying, and terminating sessions with one or more participants. It can be used to create two-party, multi-party, or multi-cast sessions that include Internet telephone calls, multimedia distribution, and multimedia conferences. (Cit. RFC 3261). SIP is designed to be independent of the underlying transport layer; it can run on TCP, UDP, or SCTP. It was originally designed by Henning Schulzrinne (Columbia University) and Mark Handley (UCL) starting in 1996. The latest version of the specification is RFC 3261 from the IETF SIP Working Group. In November 2000, SIP was accepted as a 3GPP signaling protocol and permanent element of the IMS architecture. It is widely used as a signaling protocol for Voice over IP, along with H.323 and others.

- **IMS-Client**

  An IMS Client program(a handset simulator) which is capable of depositing and retrieving voice mails, originating and terminating calls via SIP and RTP protocols for test purposes.

- **VMS**

  Voice Mail System, is a centralized system of managing telephone messages for a large group of people. It should be capable of depositing, retrieving and deleting voice mails as well as notifying the user when he has got new voice mails from other users.

Appendix B

SIP Signaling Flows and Message Contents

SIP signaling flow path 1,2 and message contents in IMS Test Environment Simulator

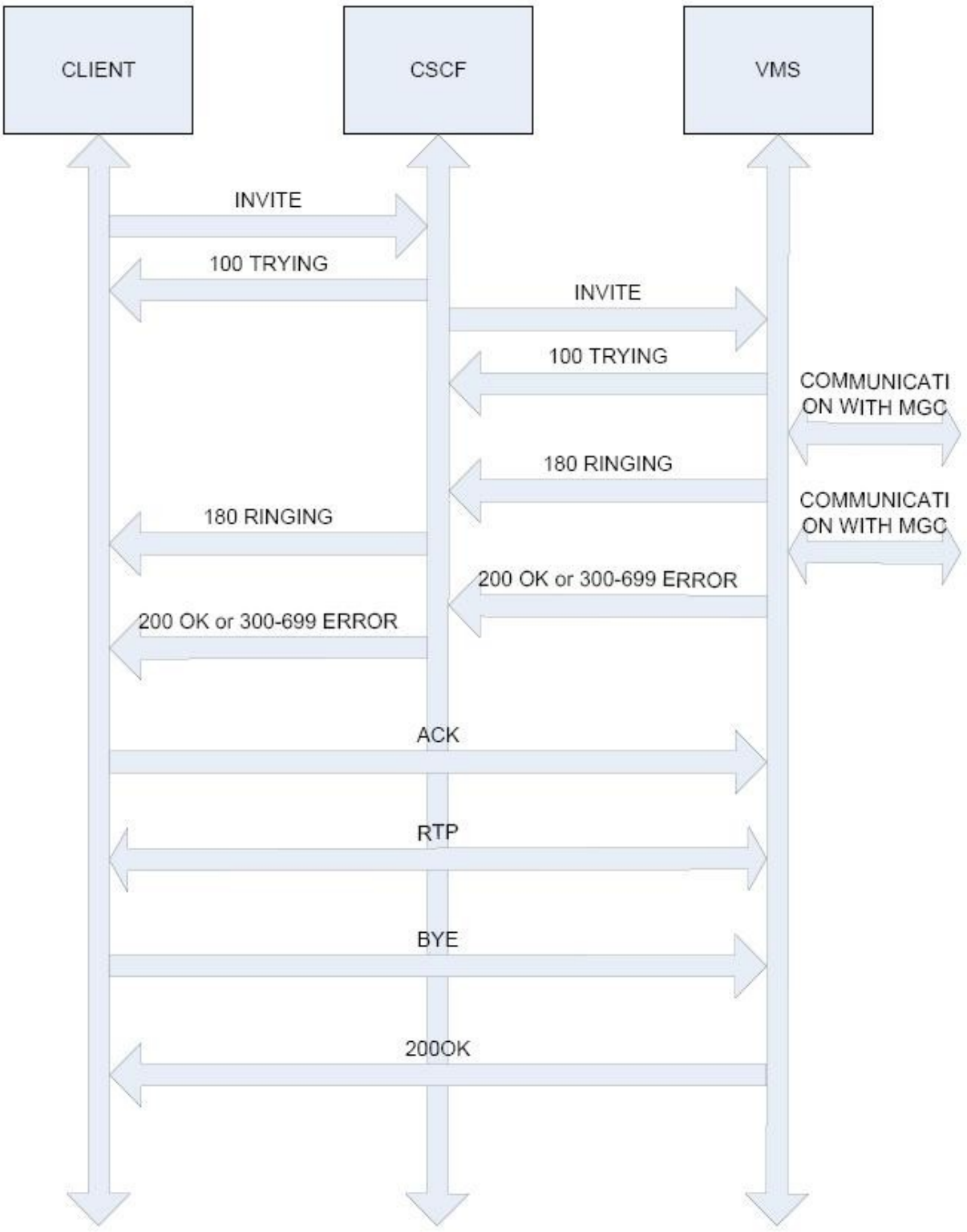

FIGURE B.1: SIP Signaling Flow Path 1 in IMS Test Environment Simulator

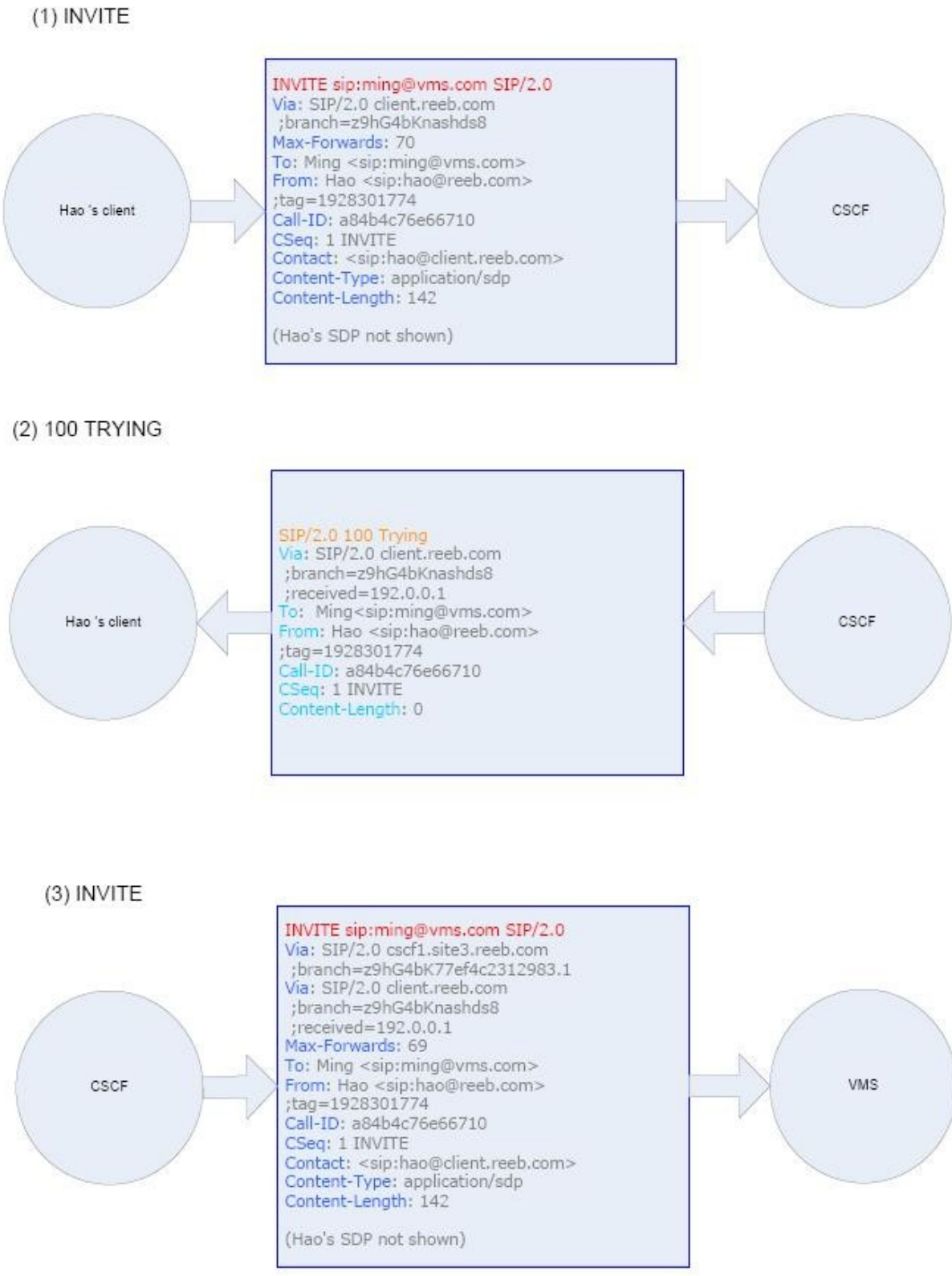

FIGURE B.2: SIP Signaling Flow Path 1 Message Contents

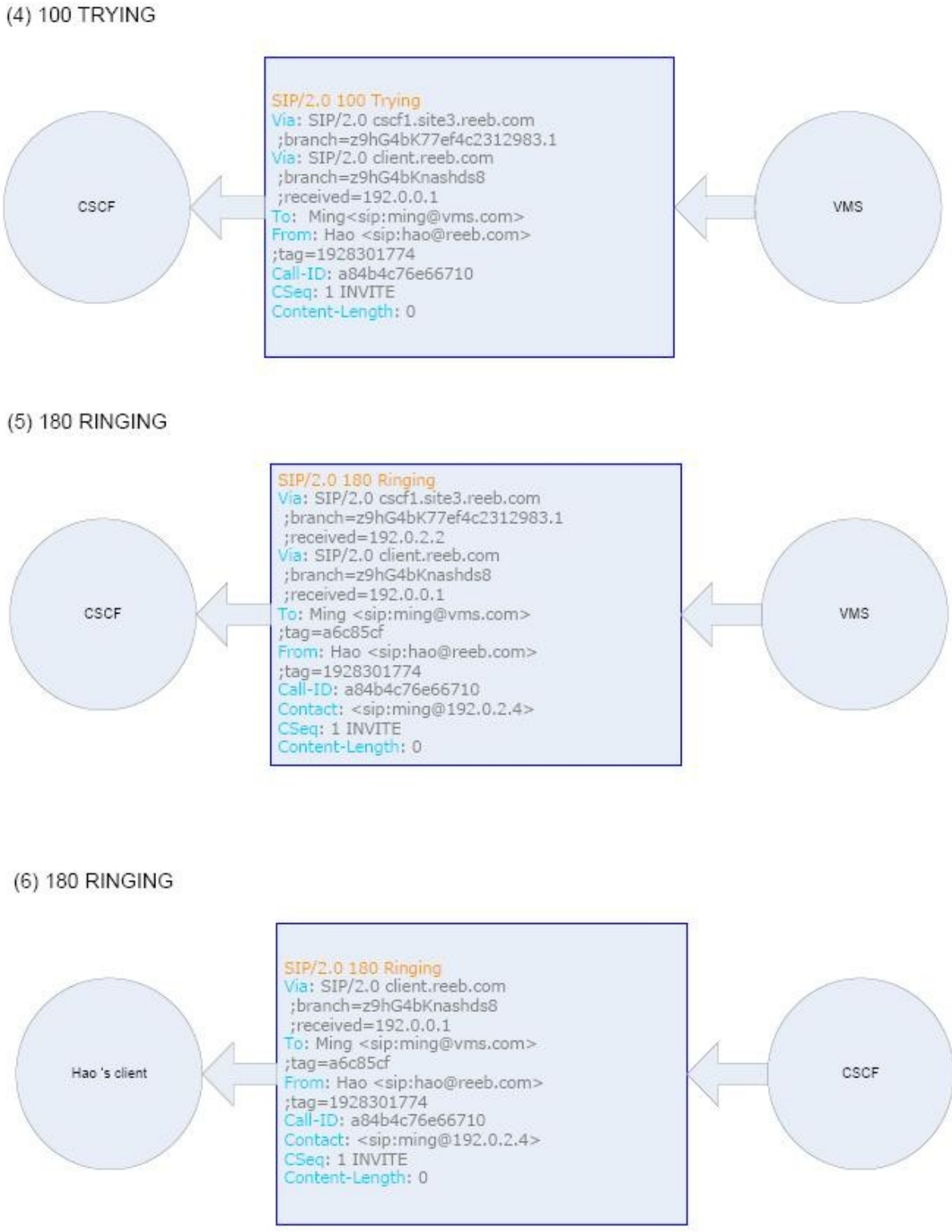

FIGURE B.3: SIP Signaling Flow Path 1 Message Contents (continue)

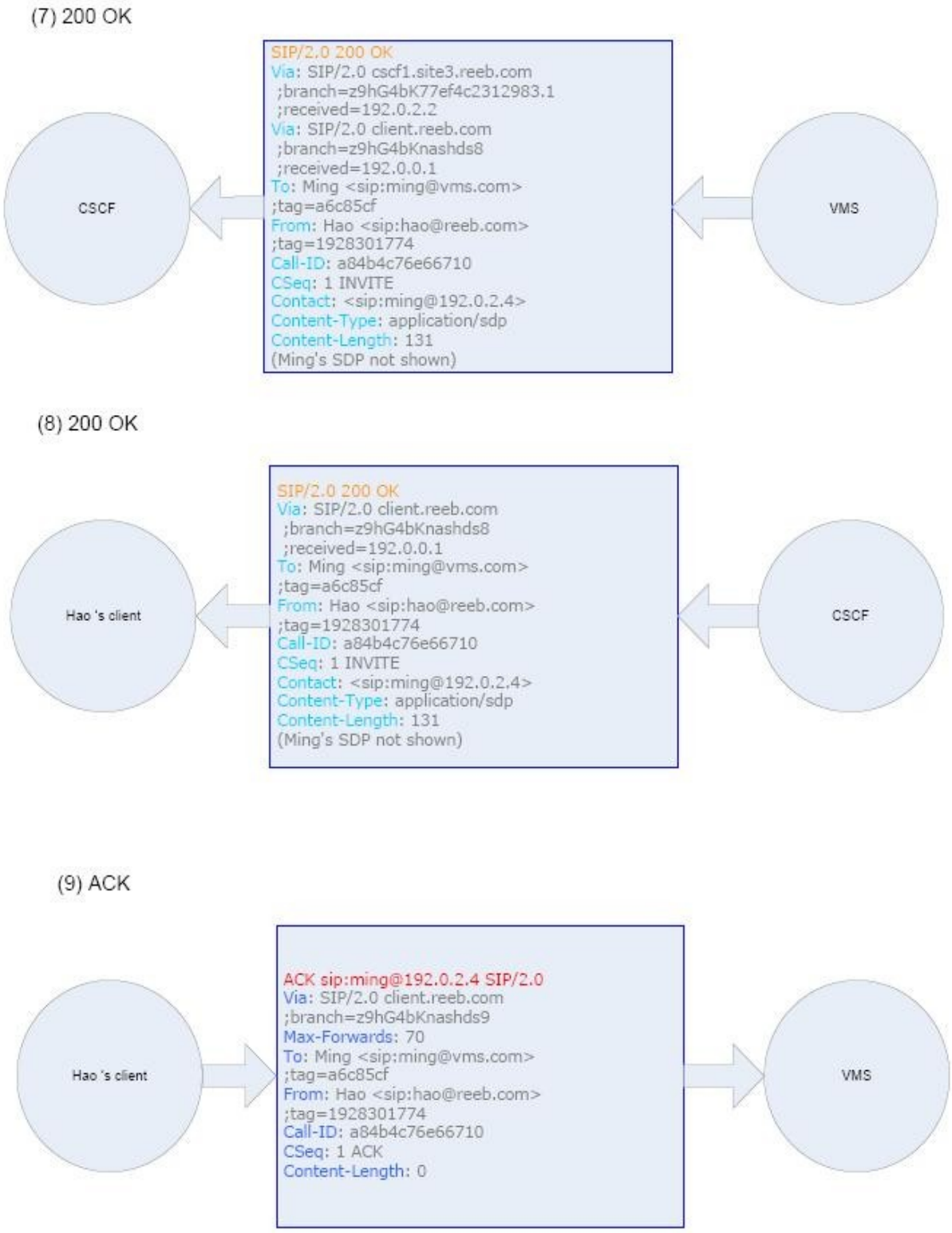

FIGURE B.4: SIP Signaling Flow Path 1 Message Contents (continue)

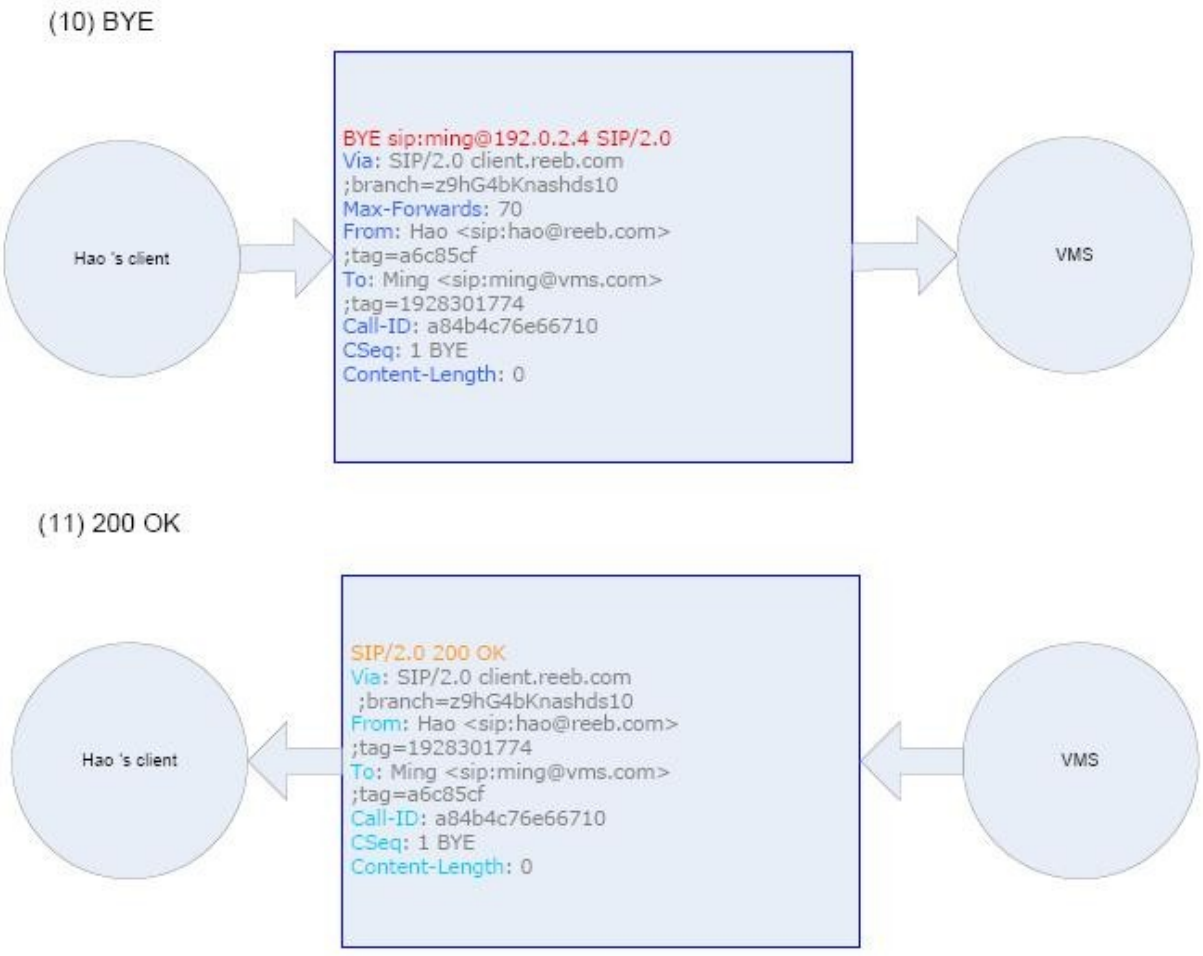

FIGURE B.5: SIP Signaling Flow Path 1 Message Contents (continue)

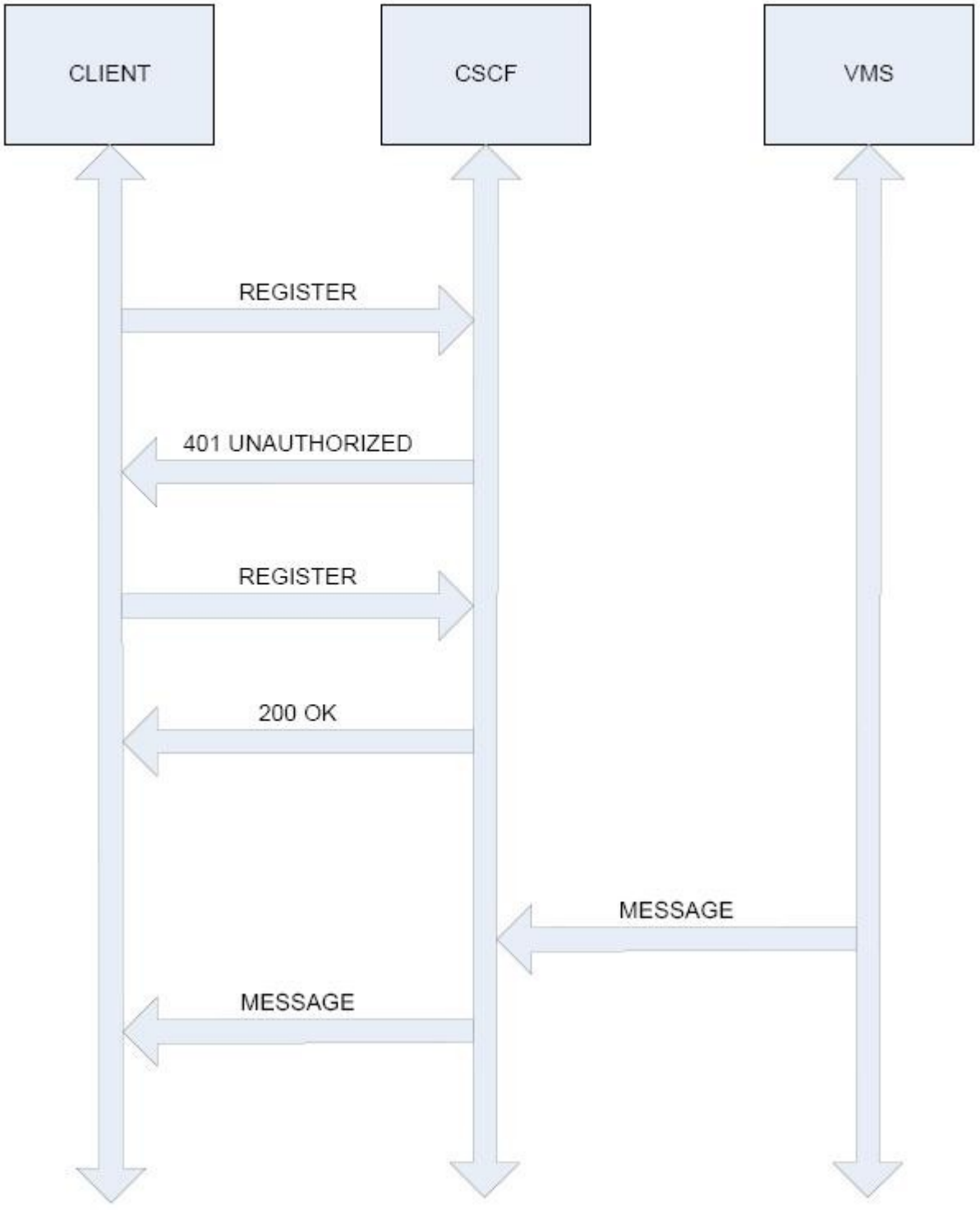

FIGURE B.6: SIP Signaling Flow Path 2 in IMS Test Environment Simulator

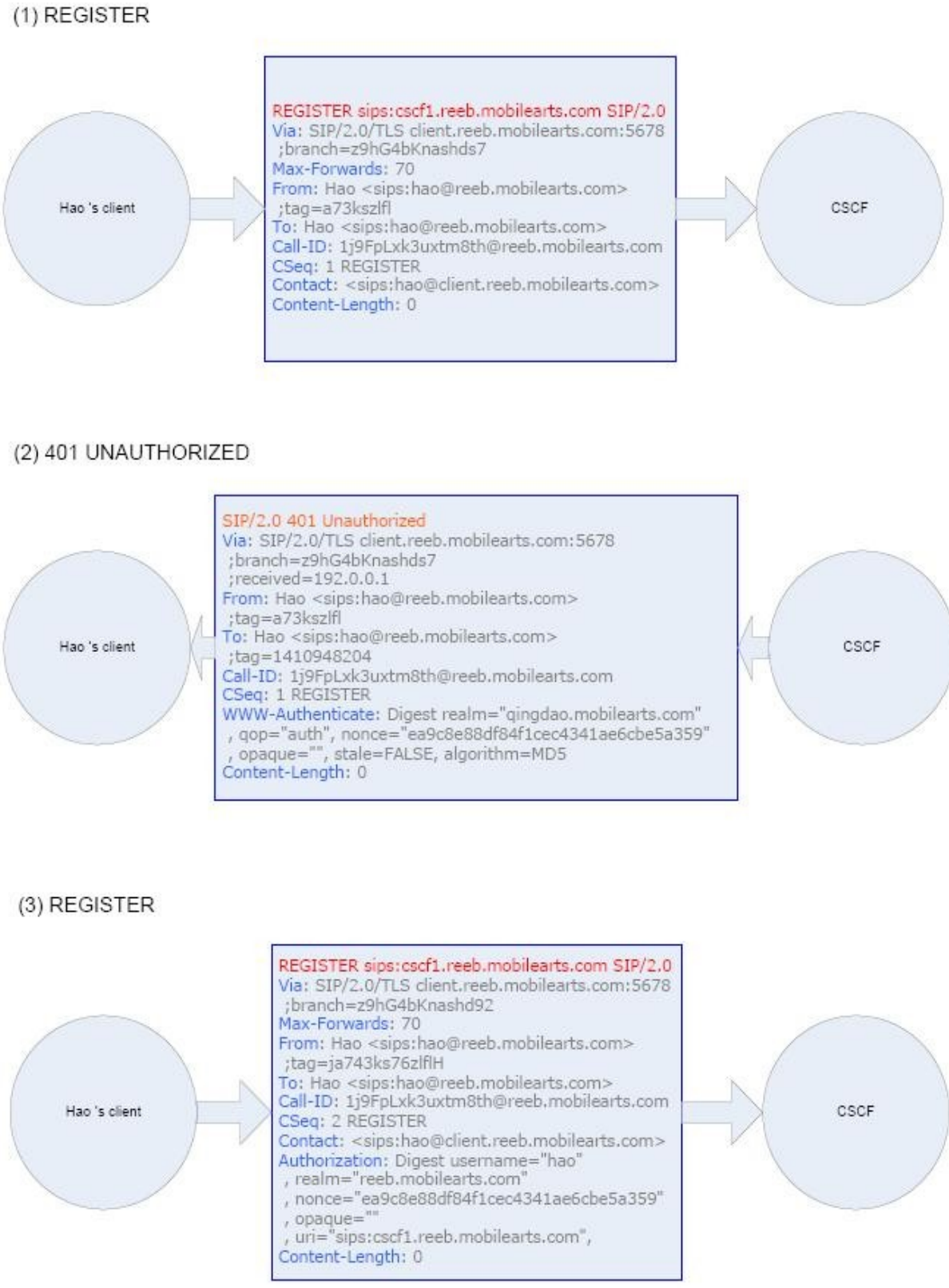

FIGURE B.7: SIP Signaling Flow Path 2 Message Contents

FIGURE B.8: SIP Signaling Flow Path 2 Message Contents (continue)

Appendix C

Diameter Commands

Each command is assigned a command code, which is used for both Requests and Answers.

Table C.1: Diameter Commands

Command Name	Abbreviation	Command Code
Capabilities-Exchange-Request	CER	257
Capabilities-Exchange-Answer	CEA	257
Re-Auth-Request	RAR	258
Re-Auth-Answer	RAA	258
AA-Request	AAR	265
AA-Answer	AAA	265
Diameter-EAP-Request	DER	268
Diameter-EAP-Answer	DEA	268
Accounting-Request	ACR	271
Accounting-Answer	ACA	271
Credit-Control-Request	CCR	272
Credit-Control-Answer	CCA	272
Abort-Session-Request	ASR	274
Abort-Session-Answer	ASA	274
Session-Termination-Request	STR	275
Session-Termination-Answer	STA	275
Device-Watchdog-Request	DWR	280
Device-Watchdog-Answer	DWA	280
Disconnect-Peer-Request	DPR	282
Disconnect-Peer-Answer	DPA	282

Command Name	Abbreviation	Command Code
User-Authorization-Request	UAR	300
User-Authorization-Answer	UAA	300
Server-Assignment-Request	SAR	301
Server-Assignment-Answer	SAA	301
Location-Info-Request	LIR	302
Location-Info-Answer	LIA	302
Multimedia-Auth-Request	MAR	303
Multimedia-Auth-Answer	MAA	303
Registration-Termination-Request	RTR	304
Registration-Termination-Answer	RTA	304
Push-Profile-Request	PPR	305
Push-Profile-Answer	PPA	305
User-Data-Request	UDR	306
User-Data-Answer	UDA	306
Profile-Update-Request	PUR	307
Profile-Update-Answer	PUA	307
Subscribe-Notifications-Request	SNR	308
Subscribe-Notifications-Answer	SNA	308
Push-Notification-Request	PNR	309
Push-Notification-Answer	PNA	309
Bootstrapping-Info-Request	BIR	310
Bootstrapping-Info-Answer	BIA	310
Message-Process-Request	MPR	311
Message-Process-Answer	MPA	311

References

[1] *3GPP TS 22.228 Service requirements for the IP multimedia core network subsystem.* Service aspects ("stage 1"), 3GPP, 2007.

[2] *3GPP TS 23.228 IP Multimedia Subsystem (IMS).* Technical realization ("stage 2"), 3GPP, 2007.

[3] *IETF RFC 3261 SIP: Session Initiation Protocol.* Standards Track, IETF, 2002.

[4] *3GPP TS 24.229 IP Multimedia Call Control Protocol based on SIP and SDP.* Signalling protocols ("stage 3") - user equipment to network, 3GPP, 2006.

[5] *IETF RFC 3550 RTP: A Transport Protocol for Real-Time Applications.* Standards Track, IETF, 2003.

[6] *IETF RFC 3588 Diameter Base Protocol.* Standards Track, IETF, 2003.

[7] Henrik Back and Ming Zhao. "Voice Mail System for IP Multimedia Subsystem (VMSIPMS)". Master's thesis, Department of Information Technology, Uppsala University, 2008.

[8] Miikka Poikselka, Georg Mayer, Hisham Khartabil, and Aki Niemi. *"The IMS: IP Multimedia Concepts and Services".* John Wiley Sons, second edition, 2006.

[9] Gonzalo Camarillo and Miguel A. Garcia-Martin. *"The 3G IP Multimedia Subsystem (IMS): Merging the Internet and the Cellular Worlds".* John Wiley Sons, second edition, 2006.

[10] Martin Koukal and Robert Bestak. "Architecture of IP Multimedia Subsystem (AIPMS)". Zadar, Croatia, 2006. 48th International Symposium ELMAR.

[11] V.S. Abhayawardhana and R. Babbage. "A Traffic Model for the IP Multimedia Subsystem (ATMIPMS)". *IEEE*, pages 1550–2252, 2007.

[12] *3GPP TS 29.328 IMS Sh interface: Signaling flows and message contents.* Signalling protocols ("stage 3") - intra-fixed-network, 3GPP, 2006.

[13] *3GPP TS 29.329 Sh interface based on the Diameter protocol: Protocol details.* Signalling protocols ("stage 3") - intra-fixed-network, 3GPP, 2006.

[14] *3GPP TS 29.228 IMS Cx and Dx Interfaces: Signaling flows and message contents.* Signalling protocols ("stage 3") - intra-fixed-network, 3GPP, 2006.

[15] *3GPP TS 29.229 Cx and Dx Interfaces based on the Diameter protocol: Protocol details.* Signalling protocols ("stage 3") - intra-fixed-network, 3GPP, 2007.

[16] *IETF RFC 2916 E.164 number and DNS.* Standards Track, IETF, 2000.

[17] *IETF RFC 3428 Session Initiation Protocol (SIP) Extension for Instant Messaging.* Standards Track, IETF, 2002.

[18] *3GPP TS 24.247 Messaging using the IP Multimedia (IM) Core Network (CN) subsystem.* Signalling protocols ("stage 3") - user equipment to network, 3GPP, 2007.

[19] Chang-Ki Kim, Sang-Chul Oh, and Yeon-Seung Shin. "A Design of Home Subscriber Server for IP Multimedia Service in All-IP UMTS Network (ADHSSIPM-SAIPUMTSN)". *IEEE*, pages 7803–7822, 2003.

[20] *3GPP TS 24.228 Signaling Flows for the IP Multimedia Call Control based on SIP and SDP.* Signalling protocols ("stage 3") - user equipment to network, 3GPP, 2006.

[21] *IETF RFC 3665 Session Initiation Protocol (SIP) Basic Call Flow Examples.* Best Current Practice, IETF, 2003.

[22] *IETF RFC 4566 SDP: Session Description Protocol.* Standards Track, IETF, 2006.

[23] *IETF RFC 3264 An Offer/Answer Model with the Session Description Protocol (SDP).* Standards Track, IETF, 2002.

[24] *IETF RFC 3589 Diameter Command Codes for Third Generation Partnership Project (3GPP).* Informational, IETF, 2003.

[25] *IETF RFC 3761 The E.164 to Uniform Resource Identifiers (URI) Dynamic Delegation Discovery System (DDDS) Application (ENUM).* Standards Track, IETF, 2004.