SIMULATION BASED INVESTIGATION OF AN IMPROVEMENT FOR FASTER SIP RE-REGISTRATION

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

EDA TANRIVERDİ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

JULY 2004

Approval of the Graduate School of Natural and Applied Sciences

_____

Prof. Dr. Canan ÖZGEN

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

_____

Prof. Dr. Mübeccel DEMİREKLER

Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____

Prof. Dr. Semih BİLGEN

Supervisor

Examining Committee Members

Prof. Dr. Buyurman BAYKAL (METU, EE)_____

Prof. Dr. Semih BİLGEN (METU, EE)_____

Assoc. Prof. Dr. Gözde BOZDAĞI AKAR (METU, EE)_____

Asst. Prof. Dr. Cüneyt BAZLAMAÇCI (METU, EE) _____

Dr. Altan Koçyiğit (METU, II)_____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name :

Signature :

ABSTRACT

SIMULATION BASED INVESTIGATION OF AN IMPROVEMENT FOR FASTER SIP RE-REGISTRATION

TANRIVERDİ, Eda

M.Sc., Department of Electrical and Electronics Engineering

Supervisor: Prof. Dr. Semih BİLGEN

July 2004, 78 pages

In this thesis, the Session Initiation Protocol (SIP) is studied and an improvement for faster re-registration is proposed. This proposal, namely the "registration – activation", is investigated with a simulation prepared using OPNET.

The literature about wireless mobile networks and SIP mobility is reviewed. Conditions for an effective mobile SIP network simulation are designed using message sequence charts. The testbed in [1] formed by Dutta et. al. that has been used to observe SIP handover performance is simulated and validated. The mobile nodes, SIP Proxy

servers, DHCP servers and network topology are simulated on “OPNET Modeler Radio”. Once the simulation is proven to be valid, the “registration – activation” is implemented.

Different simulation scenarios are set up and run, with different mobile node speeds and different numbers of mobile nodes.

The results show that the re-registration delay is improved by applying the “registration – activation” but the percentage of improvement depends on the improvement in the database access delay in the SIP Proxy server.

Keywords: Session Initiation Protocol, SIP, re-registration delay

ÖZ

OTURUM BAŞLATMA PROTOKOLÜNDE YENİDEN KAYDOLMA SÜRESİNİ AZALTMAK İÇİN ÖNERİLEN BİR İYİLEŞTİRMENİN SİMÜLASYON YOLU İLE İNCELENMESİ

TANRIVERDİ, Eda

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Danışmanı: Prof. Dr. Semih BİLGEN

Temmuz 2004, 78 sayfa

Bu tezde, Oturum Başlatma Protokolü (SIP) incelenmiş ve bu protokolde yeniden kaydolma süresini azalmak için bir öneri yapılmıştır. "Kaydolma – aktifleştirme" adını verdiğimiz bu öneri, OPNET paket programı ile hazırlanan bir simulasyonda incelenmiştir.

Kablosuz mobil haberleşme ağları ve SIP'in mobil kullanımı hakkındaki yayınlar taranmıştır. Hazırlanacak olan simülasyon, mesajlaşma şemaları kullanımı ile en efektif

şekilde tasarlanmıştır. SIP'in aktarım performansının değerlendirilmesi için daha önce hazırlanmış olan [1] bir "testbed" hakkındaki makale incelenmiş, bu "testbed" benzer şekilde simüle edilip benzer sonuçlar alınarak simülasyon doğrulanmıştır.

OPNET ile hazırlanan simülasyon doğrulandıktan sonra, planlanan iyileştirme eklenmiştir. Değişik sayıda ve değişik hızda mobil SIP kullanıcıları ile, "kaydolma – aktifleştirme" önerimiz uygulanırken ve uygulanmazken, farklı simülasyon senaryoları koşturulmuştur.

Elde edilen sonuçlar, "kaydolma – aktifleştirme" önerisinin yeniden kaydolma süresini kısaltmakta başarılı olduğunu, ancak bu başarı yüzdesinin "SIP Proxy" elemanındaki veritabanına erişim süresindeki kısalmaya bağlı olduğunu göstermektedir.

Anahtar Kelimeler: Oturum Başlatma Protokolü, Yeniden Kaydolma Süresi

ACKNOWLEDGMENTS

I express very sincere appreciation to Prof. Dr. Semih Bilgen for his valuable supervision throughout the research. With his precious ideas that he has shared with me at every phase of the study, this thesis could be completed.

I wish to thank to Tolga İpek on behalf of ASELSAN Inc. for his encouragement and support. I am grateful to my chief engineer Özgü Özköse Erdoğan for her tolerance during the study.

I would like to thank to my father İsmail Tanrıverdi, and my mother Gönül Tanrıverdi for being guiding parents and for their generous, self-denying support throughout my education.

Finally, I have very special thanks to my dear fiancé Seçkin Gürler for his support and patience.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CH	Correspondent Host
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name Service
DRCP	Dynamic Registration and Configuration Protocol
HTTP	Hyper Text Transport Protocol
IETF	Internet Engineering Task Force
ITU	International Telecommunication Union
ITSUMO	Internet Technologies Supporting Universal Mobile Operations
MH	Mobile Host
MMUSIC	Multiparty Multimedia Session Control
MSC	Message Sequence Chart
MWIF	The Mobile Wireless Internet Forum
OPNET	Optimized Network Engineering Tool
RFC 2543	Request For Comments 2543
SIP	Session Initiation Protocol
UA	User Agent
UAC	User Agent Client

UAS	User Agent Server
UDP	User Datagram Protocol
URL	Uniform Resource Locator
3GPP	3rd Generation Partnership Project

CHAPTER I

INTRODUCTION

With recent advances in wireless communication technology, mobile computing is an increasingly important area of research. Mobility is rapidly becoming a rule rather than an exception in communication services. Internet access has already gained significant attention as wireless/mobile communications and networking are becoming widespread. A large amount of effort has been expended over the years on allowing computer and communication devices to continue communicating even when mobile.

However, the vast majority of mobile communication devices today continue to be cellular phones [5]. Alongside, the VoIP service is likely to play a key role in the convergence of the IP-based Internet and the conventional cellular networks. In providing the VoIP service, the most viable concern is the amount of time to process the handoff of an ongoing VoIP call [8]. Application layer support is necessary to offer more than just handoff between base stations and subnets, and it can, under some circumstances, compensate for the lack of deployment of network layer mobility support. The protocol at the heart of this effort is the Session Initiation Protocol (SIP), an Internet Engineering Task Force (IETF)-developed signaling protocol.

SIP is an application layer protocol used for establishing and tearing down sessions with one or more participants [3, 4]. It has been standardized within the IETF for the invitation to multicast conferences and Internet telephone calls. Entities in SIP are user agents, proxy servers and redirect servers. A user is addressed using an email – like address "sip:user@host", where "user" is a user name or phone number and "host" is a domain name or numerical address.

SIP defines a number of methods, namely INVITE, ACK, BYE, OPTIONS, CANCEL and REGISTER. Responses to these methods indicate success or failure, distinguished by status codes, 1xx (100 to 199) for progress updates, 2xx for success, 3xx for redirection, and higher numbers for failure. Each new SIP transaction has a unique call identifier, which identifies the session. If the session needs to be modified, e.g. for adding another media, the same call identifier is used as in the initial request, in order to indicate that this is a modification of an existing session.

The SIP user agent has two basic functions: Listening for incoming SIP messages, and sending SIP messages upon user actions or incoming messages. The SIP proxy server relays SIP messages, so that it is possible to use a domain name to find a user, rather than knowing the IP address or name of the host. A redirect server returns the location of the host rather than relaying the SIP messages. Both the redirect and proxy servers accept registrations from users, in which the current location of the user is given.

SIP can take communications beyond network-level convergence to a new era of application-level convergence and interactive communications [13]. It is positioned to enable the next revolution where communication services will be integrated seamlessly

and intuitively into the daily work environment. The web services and web based technologies available will determine whether SIP is successful, both from an ease of use and an ease of deployment perspective.

In this thesis work, an improvement has been proposed in order to decrease the re-registration time of mobile nodes in a wireless mobile SIP environment.

To achieve this goal, the basics of SIP operation, entities and messages are studied. The testbed that has been set in [1] in order to test the terminal mobility performance of a roaming user in a SIP signaling scheme is given particular attention. In the scope of this thesis, this testbed has been simulated using OPNET [15].

OPNET is a vast software package with an extensive set of features designed to support general network modeling and to provide specific support for particular types of network simulation projects. In this study, OPNET version 8.1.A has been used.

The simulation of the testbed in [1] has been validated by using the same topology, same number of mobile nodes, proxy servers, same message lengths and obtaining similar delay times. Once the simulation has been validated, new methods can be developed and tested to reduce the re-invite or re-registration time.

The simulation scenario can be described as follows: A number of MNs (1, 4, 7 and 10 nodes, changing in every configuration of the simulation) follow random trajectories, approaching one of the Proxy – DHCP pairs, broadcasting DHCP_DISCOVER messages. Each MN gets an IP address as soon as it can communicate with one of the DHCP servers, and afterwards, registers to the Proxy server. Then, each MN keeps on moving, following random trajectories. When a MN loses contact with the first Proxy, it gets a new IP address from the new DHCP server

and re-registers to the new Proxy server. If the MN is in the middle of a session, it can not receive data during this re-registration delay because its IP address has changed. Every MN makes several handovers during a session.

The new concept that we propose, namely “registration – activation” brings improvement to the re-registration delay of a mobile node. Once a mobile node registers to a proxy server, this proxy informs all the neighboring proxies about this node. The neighboring proxies record the registration data of the MN, but mark it as “inactive” since the MN is not in their neighborhood yet. When the MN makes a handoff, the new proxy already has got the node’s registration information, therefore only activates it in a shorter time compared to a registration. This concept brings improvement in two ways: First, the ACTIVATE message is shorter than REGISTER message, since it does not need to contain much information about the node. Second, the Proxy takes a shorter time to access its database to activate a node, compared to registering it, because less data has to be recorded.

Different configurations with different number of nodes, node speeds and activation delay times have been defined and the simulation has been run several times. The same configuration of the nodes with and without “registration – activation” has been run in order to make a good comparison. The re-registration delay and activation delay results have been collected. Two graphs that summarize these results have been plotted.

The thesis is organized as follows:

In chapter 2, SIP basics, mobility definitions and requirements, and mobility management with SIP are being studied. The entities and messages of SIP are explained. SIP operation examples are given. Mobility issues in a wireless network are classified into subtitles, and particularly SIP mobility support is explained in detail. The aim is to provide background information about the topic and determine the basics of mobility support of SIP.

In chapter 3, the wireless testbed architecture that has been implemented in [1] is explained. The message sequence charts that should be applied in our simulation program are given. A summary of the simulation tool, OPNET packet program is presented. The topology that has been implemented, the scenarios and the validation of the simulation are discussed in detail. The improvement that we bring to the re-registration time, namely the "registration – activation" concept is introduced. Finally, in this chapter, the simulation scenarios for evaluating the benefits of this new concept are explained and the summary of the results obtained from these scenarios are shown in figures.

In chapter 4, a conclusion is presented and some interesting topics related to this work for future studies are given.

The detailed results of the simulation scenarios are given in tables in the appendix.

CHAPTER II

MOBILITY MANAGEMENT WITH SIP

2.1. Introduction

In this chapter, SIP basics, mobility definitions and requirements, and mobility management with SIP are being studied. The aim is to provide background information about the topic and determine the basics of mobility support of SIP.

SIP is an application layer protocol for creating, modifying and terminating sessions with one or more participants [4]. These sessions may be multimedia conferences, internet telephone calls and similar applications consisting media types such as audio, video or whiteboard. SIP has been standardized within the IETF by RFC 2543 for the invitation to multicast conferences and Internet telephone calls. In the years 1999 – 2001 SIP was a very hot research topic [3, 5, 6, 8, 9 and 10]. It gained popularity and acceptance as the signaling protocol for next generation multimedia communication, which can be used for wireless Internet as well.

More recently, it has been stated that, SIP is promising because it is a simple, open protocol that can be deployed in carrier and enterprise networks and because it enables new multimedia applications in voice, data and video [13].

Mobility is rapidly becoming a rule rather than an exception in communication services. In this context, SIP is still a promising candidate to support mobile users for signaling and control.

2.2. The Session Initiation Protocol

SIP allows two or more participants to establish a session consisting of multiple media streams [4]. These media streams can be audio, video or any other Internet-based communication application such as distributed games, shared applications, shared text editors and white boards.

SIP is based on SMTP and HTTP, and it is a text oriented protocol like these, based on the client-server model, with requests generated by one entity (the client), and sent to a receiving entity (the server) which responds them. A request invokes a method on the server and can be sent either over TCP or UDP.

The User Datagram Protocol (UDP) is a connectionless protocol and uses the Internet Protocol to send a data unit (called a datagram) from one computer to another. Unlike TCP, UDP does not provide the service of dividing a message into packets (datagrams) and reassembling it at the other end. Specifically, UDP doesn't provide sequencing of the packets that the data arrives in or reliable delivery. This means that the application program that uses UDP must be able to make sure that the entire message has arrived and is in the right order. Network applications that want to save processing time because they have very small data units to exchange (and therefore very little message reassembling to do) may prefer UDP to TCP. This is why SIP over UDP is the commonly used architecture [9].

SIP is similar to messaging mechanisms such as Internet e-mail. For example, it shares the ability to deliver information without having to know the precise network location of the recipient. However, SIP messaging is synchronous, with end-to-end notification of success or failure, and designed for sub second delivery times.

There are many functions SIP explicitly does not provide [12]. It is not a session management or conference control protocol. The particulars of the communications within the session are outside the scope of SIP. This includes features such as media transport, voting and polling, flood control and feedback on session quality.

SIP is not a resource reservation protocol for sessions since SIP is independent of the underlying session it establishes, and the path of SIP messages is completely independent of the path packets for a session may take.

SIP is not a transfer protocol. It is not a means to send large amounts of data unrelated to SIP's operation, or to replace HTTP. This is for numerous reasons, one of which is that SIP's recommended mode of operation is over UDP. Sending large messages over SIP can lead to fragmentation at the IP layer and thus poor performance in even mildly lossy networks.

2.2.1. SIP Entities

SIP is comprised of two major architectural entities: User Agents (UAs) and the network servers. UAs reside at the SIP end stations and contain two components: user agent clients (UACs), which are responsible for processing SIP requests and user agent servers (UASs), which respond to such requests. Generally, UAs are the only elements where media and signaling converge.

There are two kinds of network servers, proxy servers and redirect servers. One other kind of network servers, that is, location servers may also be employed.

2.2.1.1 Proxy Server

Proxy servers perform application layer routing of the SIP requests and responses. Requests are serviced internally or by passing them on, possibly after translation, to other servers. A proxy interprets, and, if necessary, rewrites a request message before forwarding it. A proxy server can either be stateful or stateless. When stateful, a proxy remembers the incoming request which generated outgoing requests, and the outgoing requests. A stateless proxy forgets all information once an outgoing request is generated.

Furthermore, proxies can be either forking or non-forking. A forking proxy server can, for example, ring several phones at once until someone takes a call, similar to a home telephone. This feature supports a number of advanced telephony services, such as call forwarding to voice mail, automatic call distribution and user location. A forking proxy must be stateful.

2.2.1.2 Redirect Server

Redirect servers do not forward requests to the next server; they send redirect responses back to the client containing the address of the next server to contact.

2.2.1.3 Location Server

Finally, location servers, used in conjunction with redirect servers, may be employed to determine a user's location and presence (whether currently on-line).

2.2.2. SIP Addresses

To be able to locate and invite participants there has to be a way the called party could be addressed. In SIP, callers and callees are identified by SIP addresses. SIP addresses are referred to as SIP Uniform Resource Locators (SIP-URLs), which are of the form *sip:user@host.domain.* The user part can be a user name, a telephone number or a civil name and the host part is either a domain name or a numeric network address, such as *sip:eda@aselsan.com*, or *sip:chloe@139.179.11.11*

While this is not required, it appears likely that many users will re-use at least some of their e-mail addresses as SIP URLs in the future.

2.2.3. SIP Messages

SIP messages, specified in ISO 10646-1, are based on the Hyper Text Transport Protocol (HTTP) message format, which is a human readable, text-based encoding. There are two kinds of SIP messages, requests and responses. Requests are issued by clients, and responses are the answers of the servers. These messages include different headers to describe the details of communication.

2.2.3.1 Requests

When making a SIP call, a caller first locates the appropriate server and then sends a SIP request. There are six different kinds of request in SIP:

- INVITE: This method initiates sessions. It indicates that the user or service is being invited to participate in a session. For a two-party call, the caller indicates the type of media it is able to receive as well as their parameters such as network destination. A success response indicates in its message body which media the callee wishes to receive.

- ACK: This method confirms session establishment. The ACK request confirms that the client has received a final response to an INVITE. It may contain a message body with the final session description to be used by the callee. If the message body is empty, the callee uses the session description in the INVITE request. This method is only used with the INVITE request.
- BYE: This method terminates a session. The user agent client uses BYE to indicate to the server that it wishes to release the call.
- CANCEL: This request cancels a pending request, but does not affect a completed request. (A request is considered completed if the server has returned a final response).
- OPTIONS: This method solicits information about capabilities, but does not set up a connection.
- REGISTER: This method allows a client to bind a permanent SIP URL to a temporary SIP URL reflecting the current network location. In other words, it conveys information about a user's location to a SIP server. After the headers following the Request-Line the request may contain a message body which is separated from the headers with an empty line.

2.2.3.2 Responses

After receiving and interpreting a request message, the recipient responds with a SIP response message, indicating the status of the server, success or failure. The responses can be of different kinds and the type of response is identified by a status code, a 3-digit integer. The first digit defines the class of the response. The other two have no categorization role. The six different classes that are allowed in SIP are listed in

Table 1 with their meaning. These classes can be categorized by provisional and final responses. A provisional response is used by the server to indicate progress, but does not terminate a SIP request. A final response terminates a SIP request.

Table 1: SIP Response Methods

1XX	Informational	Provisional
2XX	Success	Final
3XX	Redirection	Final
4XX	Client Error	Final
5XX	Server Error	Final
6XX	Global Failure	Final

2.2.4. SIP Basic Operation Example

As previously mentioned there are two different ways of handling an incoming SIP request at a SIP server. These are here exemplified [9] using the most important SIP operation, inviting a participant to a call, to show the basic operation of SIP. Figure 1 and Figure 2 show a fictitious example with simplified messages and the provisional responses left out. The user *ffl* at host *science.fiction.com* wants to invite user *pgn*. He obtains *pgn*'s address from his email address, of form name@domain, which is pgn@example.se. The client then translates the domain part to a numeric IP address, by a DNS lookup, where a server may be found. This results in the server *sippo* of domain

example.se. An INVITE request is then generated and sent to this server (1). The Server accepts the invitation and contacts his location server for a more precise location (2). The location server returns the location of *pgn*, which is at host *pepperoni* (3). These steps are the same for both proxy and redirect server.

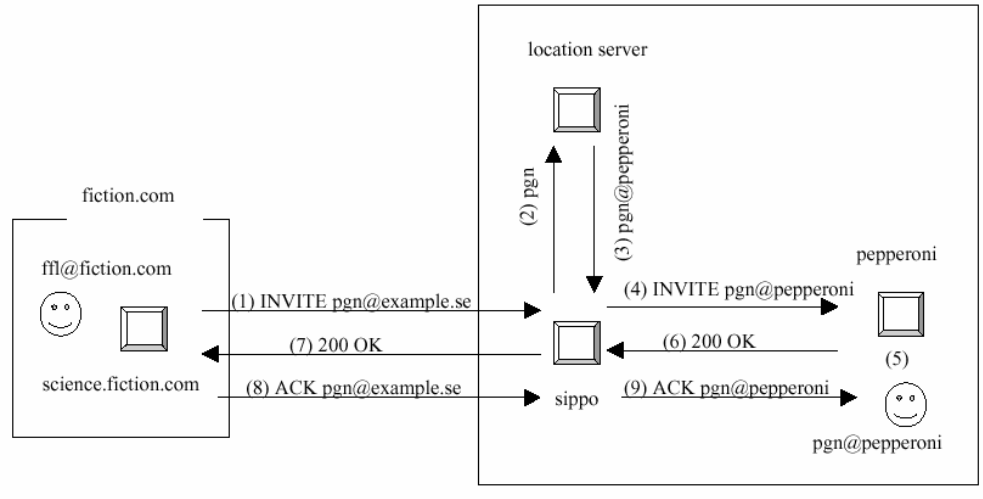

Figure 1: SIP Operation Example with Proxy Server [9]

In the proxy case (figure 1) the server then issues an INVITE request to the address given by the location server (4). The user agent server at *pepperoni* alerts the user (5), who is willing to accept the call. The acceptance is returned to the proxy server, by a 200 status response (6). A success response is then sent by the proxy to the original caller (7). This message is confirmed by caller, with an ACK request (8). The ACK request is then forwarded to the callee (9).

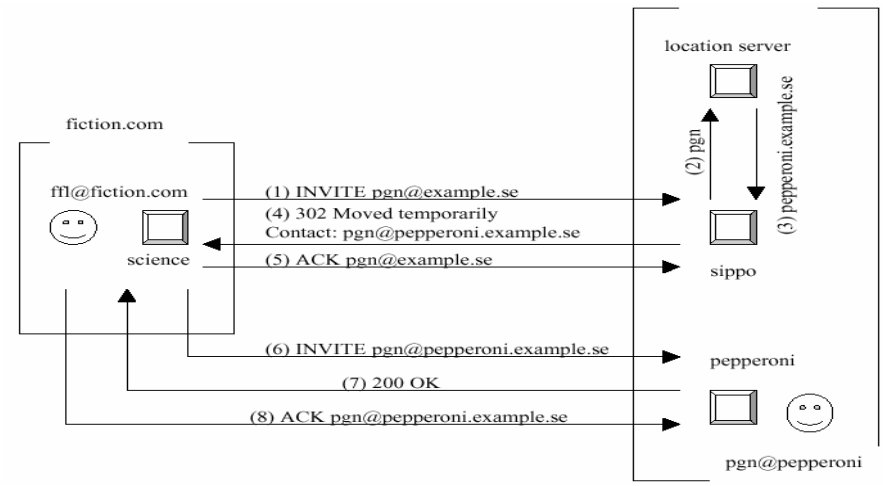

Figure 2: SIP Operation Example with Redirect Server [9]

In the redirect case (figure 2) the server returns a redirection response of class 300 giving the address to contact in the Contact header field (4). The caller acknowledges the response with an ACK request to the server (5). The caller issues a new INVITE request with the same Call-ID. This is sent to the address given by the server (6). In this case the call succeeds and a response indicating this is sent to the caller (7). The signaling is completed with an ACK from the caller (8).

2.3. Mobility Definitions and Requirements

Both the telephony and Internet are the most popular ways of communication. Two significant new features of Internet are the support of telephony and the support of mobility. It is desirable to integrate IP host mobility features to Internet telephony so that the IP host currently on the phone conversation can be on the move from network to network [5].

Mobility issues can be classified into following subtitles:

2.3.1. Personal Mobility

Personal mobility refers to the ability of end users to maintain the same identifier (address) even as they change attachment points to the network, or use different devices. This is similar to how an e-mail recipient may retrieve messages via a variety of devices from any network location. A roaming user can be accessible independent of the device he/she is logged in.

2.3.2. Service Mobility

Service mobility refers to the end user's ability to maintain ongoing sessions and obtain services in a transparent manner regardless of the end user's point of attachment. For instance, the user of a mobile terminal should be able to move a specific session to a laptop/DSL terminal without losing the session. The service mobility includes the ability of the home service provider to either maintain the control of the services it provides to the user in the visited network, or transfer their control to the visited network.

2.3.3. Terminal Mobility

Terminal mobility provides means of cell, subnet and domain hand-off while the session is in progress. It allows a device to move between IP subnets, while continuing to be reachable for incoming requests and maintaining sessions across subnet changes. A subset of terminal mobility, being able to be reached for new sessions after subnet changes, requires only DHCP and dynamic DNS.

In general, the mobility management scheme of wireless IP networks satisfies the following requirements [10]:

- It supports means of personal, service, and terminal mobility, i.e., it allows users to access network services anywhere, as well as to continue their ongoing communication and to access the network services anywhere using their own mobile terminal/station.
- It supports global roaming, i.e., it should allow users to roam across different technology platforms, and across subnets within the same administrative domain as well as across subnets that belong to different administrative domains.
- It is wireless "technology-independent", i.e., it should remain independent of the underlying wireless technology. This requirement allows the ISPs and network operators to upgrade their sub-systems independently and build multi-vendor solutions. Furthermore, this requirement ensures that the mobility management scheme can be transported over all members of the IMT-2000 family which comprises several wireless technologies such as W-CDMA and TDMA.
- It supports both real-time and non-real-time multimedia services such as mobile telephony, mobile web access, and mobile data services in a way that their prices and performance are comparable to those of their counterparts in today's mobile voice and data services. In order to do so, the mobility management scheme should interact effectively with the QoS management and authentication, authorization, and accounting (AAA) schemes of the end-to-end network to verify the users' identities and rights, as well as to ensure that the QoS requirements of applications are satisfied and maintained as users roam around.
- It transparently supports current TCP-based Internet application. It should support TCP as is without requiring any changes to TCP protocol or TCP-based

applications, i.e., it should spoof/maintain constant end-points for TCP connections.

- It allows a mobile station/user to decide whether it conveys its location to correspondent hosts or not. This requirement allows users to enhance their confidentiality and privacy, when necessary.
- It supports multicast and anycast trees efficiently as mobile stations/users move around.
- Last but not least, it interworks smoothly with PSTN and today's 1G/2G wireless telephony to facilitate interworking of new operators' all IP platforms with PSTN and the existing 1G/2G wireless telephony.

2.4. Mobility Support of SIP

One of the central tasks of SIP is to locate one or more IP addresses where a user can receive media streams, given only a generic, location-independent address identifying a domain [1, 2, 3, 6, 8 and 10]. It has already been accepted as a means for signaling for session management in many of the standard bodies such as MWIF, 3GPP who are designing the architecture for all IP wireless networks.

SIP supports personal mobility as part of its signaling mechanism. The step from personal mobility to IP mobility support is basically the roaming frequency, and that a user can change location (IP address) during a traffic flow. In order to support IP mobility, we need to add the ability to move while a session is active. In other words, SIP feature set should be extended to provide adequate means of terminal and service mobility. Mobility management protocol stack is shown in Figure 3.

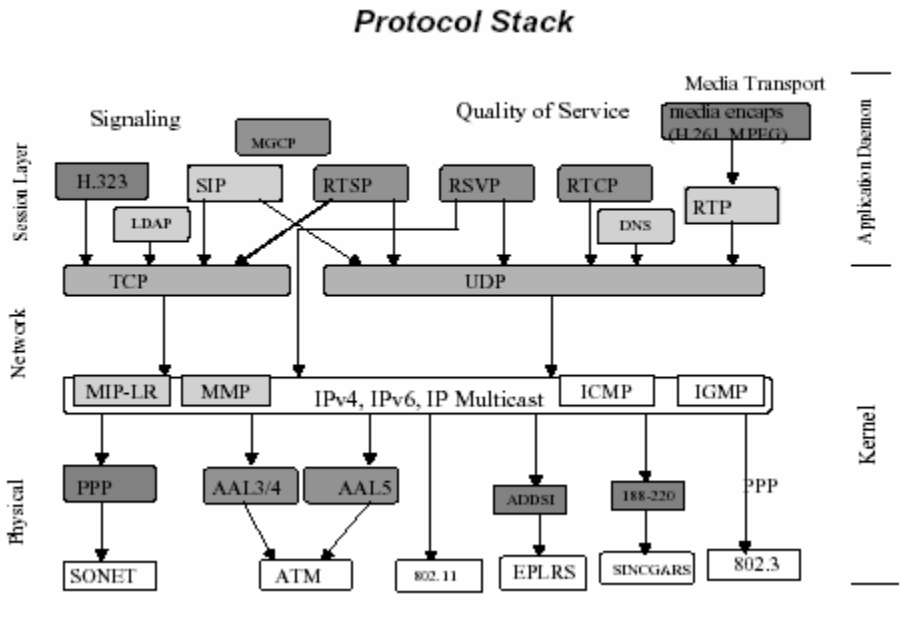

Figure 3: Mobility Management Protocol Stack [17]

Terminal mobility impacts SIP at three stages, pre-session, mid-session and to recover from network partitions.

2.4.1.1.Pre-Session Mobility

Pre-session mobility guarantees that a correspondent host (CH) can reach a mobile host (MH) before a session starts. The MH gets a new IP address prior to receiving or making a call. Each time the MH obtains a new IP address, it re-registers with its home registrar. The CH that calls the MH obtains the location of the MH from its home registrar. Pre-session mobility is shown in Figure 4.

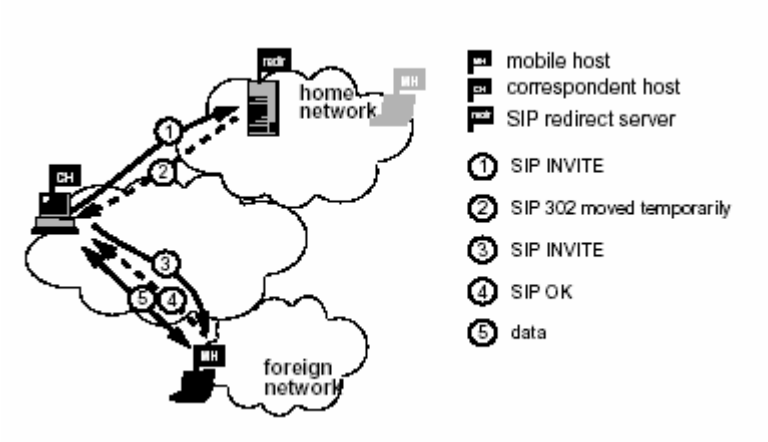

Figure 4: Pre-Session Mobility [5]

2.4.1.2.Mid-Session Mobility

Mid-session mobility allows a node to continue an ongoing session with its peer during handoff. The moving MH sends another INVITE request to the CH without going through any intermediate SIP proxies. This INVITE request contains an updated session description with the new IP address. Thus, the location update takes one one-way delay after the application in the MH recognizes that is has acquired a new IP address. Mid-session mobility is shown in Figure 5.

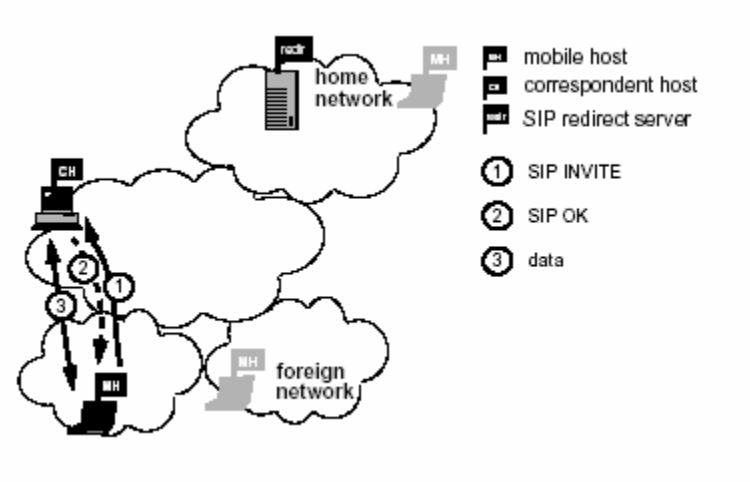

Figure 5: Mid-Session Mobility [5]

2.4.1.3.Recovery from Network Partitions

If the network partition lasts less than about 30 seconds, SIP will recover without further mechanisms, as it retransmits the request if there is no answer. If the network partitions last longer, updates may be lost and the other host may also have moved [3]. Recovery from network partitions is shown in Figure 6.

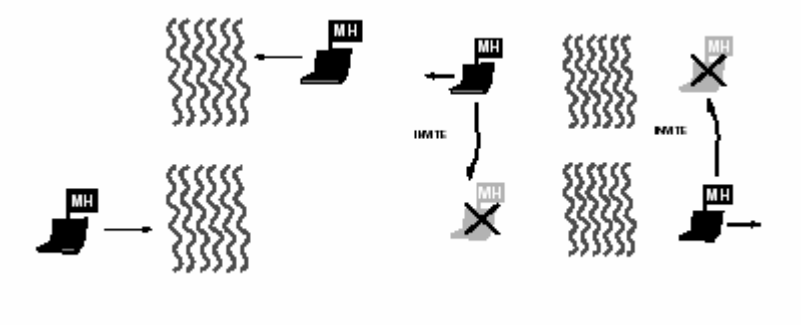

Figure 6: Recovery from Network Partitions [3]

One example of this is when we have two mobile hosts having a conversation, both lose contact for a while, by driving into a tunnel, and when they gain contact again, they have both got new IP addresses. In this case, to rendezvous again, each side should address the SIP INVITE request to the canonical address, the home proxy of the other side. Since the SIP servers have fixed addresses, the mobile hosts can always send registrations to them.

Application layer mobility support using SIP signaling in an Internet roaming environment needs to be investigated. Simulations and experiments are being carried out to measure the efficiency of SIP based mobility scheme, compared to other mobility management schemes, like Mobile IP. There are some papers [6] for extending SIP to support location management, registration of roaming mobiles and handoffs. Some

other architectures [3] recommend to use SIP in cooperation with Mobile IP; SIP over UDP handling the real-time traffic and Mobile IP over TCP handling non real-time traffic.

A significant alternative to SIP is H.323. H.323 is a series of recommendations of ITU that define protocols and procedures for multimedia communications on the Internet. Many evaluation studies have been published comparing H.323 with SIP [9, 11]. The general conclusion in these evaluations is that, H.323 is much more complex than SIP, with hundreds of elements; and H.323 is more limited for usage in a WAN. However, in the enterprise market, H.323 has gained significant support because of its manageability, reliability and interoperability with the PSTN and will continue to do so for some time [13]. Most serious implementers of SIP are considering a SIP to H.323 inter-working function / gateway as an essential element of any network.

Although for interactive sessions, SIP-based mobility is considered to provide all common forms of mobility, for terminal mobility, an IPv6-based solution is said to be more preferable [5]. The reason is that, an IPv6-based solution applies to all IP-based applications, rather than just Internet telephony and conferencing.

In another analysis [8], Shadow Registration concept has been proposed to reduce the disruption time in the inter-domain handoff. Mobile IP and SIP approaches have been compared in terms of mobility management. The disruption for handoff of the Mobile IP approach is smaller than that of the SIP approach in most situations; however, the SIP approach shows shorter disruption when the MN and CN are close.

In this thesis, we shall focus on the handover performance evaluation of mobile packet networks using SIP for mobility management.

CHAPTER III

MOBILE SIP NETWORK SIMULATION

3.1. Introduction

In this chapter, SIP simulation details and a summary of the simulation tool, OPNET packet program, is presented. A detailed explanation of OPNET simulation model with state diagrams and message sequence charts are given. The topology that has been implemented, the scenarios and the results of the simulation are discussed in detail.

3.2. SIP Simulation Requirements

The aim of this thesis work is to simulate a mobile SIP environment, and then to develop a new idea to make faster handover. In order to achieve this goal, the first step is to simulate a mobile SIP environment, and to prove that this simulation is valid. Once the validity of the simulation is proven, some improvements can be made to the application, and the results of these improvements can be tracked.

3.2.1. Wireless Testbed Implementations

The members of Internet Technologies Supporting Universal Mobile Operations (ITSUMO) team have designed and implemented some wireless testbed

architectures, and some papers have been published [1, 16]. In order to integrate SIP based terminal mobility, Columbia University's SIP Client and SIP Server models have been used in both reference testbeds.

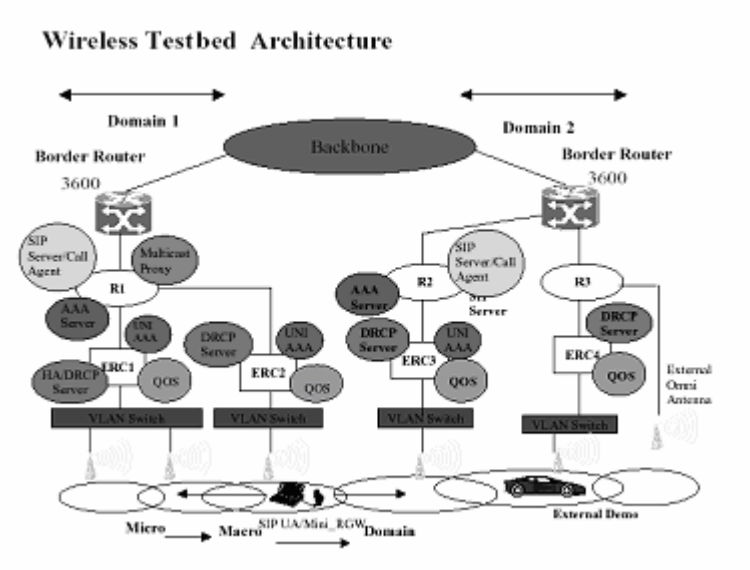

Figure 7: Wireless Testbed Architecture Example [1]

The testbed whose snapshot is shown in Figure 7 was used for prototyping different functionality of application layer mobility management scheme including different types of mobility such as micro, macro and domain mobility [1]. As part of the experiment, SIP based terminal mobility for real-time traffic was tried. A multimedia SIP session for real-time traffic was established between two clients in an IEEE 802.11 environment. The MH moves away from the CH while the session is in progress. As soon as the MH discovers that it is in a new subnet, it gets a new IP address using DRCP [14], a light-weight version of DHCP, sends a Re-Invite to the CH and sends a new registration message to the registrar.

Packet sizes for the SIP messages that have been used in this reference are also used in our simulation. These messages and their packet sizes are shown in Table 2.

Table 2: SIP Messages Packet Sizes

INVITE	455 bytes
ACK	216 bytes
REGISTER	370 bytes
REGISTER_OK	412 bytes
RE-REGISTER	425 bytes
RE-REGISTER_OK	410 bytes
RE-INVITE	450 bytes

In the testbed in [1], Linux based MH has been used. It takes about 100 msec for the MH to process a message and send an appropriate reply. It takes about 5 msec to forward the INVITE packet to traverse between MH and CH when MH is at its home network. As the MH moves away to a foreign network with more network routers in between, RE_INVITE takes about 70 msec because of the queuing delay at the routers enroute. It takes about 150 msec to complete the whole re-registration process. These figures would vary depending upon the operating system, and network topology. It would be desirable to reduce the re-invite time and re-registration time in a wireless roaming environment for faster handoff and avoiding loss of data.

In a mobile SIP simulation, if these SIP request and response messages are used between MN, CH, Proxy Server and DHCP Server, and if these delay times can be achieved, the simulation can be assumed to be valid.

3.2.2. Simulation Messaging and Message Sequence Charts

For a mobile SIP environment, once the testbed that has been described in [1] is simulated, some changes can be done to the messages and message sequences to bring some improvement to the handover time.

The simulation has 2 MNs, 2 Proxy server nodes and 2 DHCP server nodes, a total of 6 nodes. The Proxy and DHCP nodes provide two Proxy – DHCP pairs. There is a proposal that DHCP_OFFER message includes SIP information [8], which is assumed in this simulation. The DHCP server of the DHCP – Proxy pairs has got the SIP information of the Proxy server, i.e. the IP address of the Proxy server. When a new node comes to the neighborhood of the DHCP server and broadcasts DHCP_DISCOVER message, the DHCP server provides the IP address of the Proxy server in that neighborhood with DHCP_OFFER message.

When a MN comes to the neighborhood of a Proxy – DHCP pair, it broadcasts DHCP_DISCOVER messages periodically until it receives a DHCP_OFFER message from a DHCP server. This message is replied with a DHCP_REQUEST if the MN wants to take an IP address from that DHCP server, and to be a registered node in that region. The DHCP server replies this request with a DHCP_ACK message. Once the MN takes a new IP address, it can register to the Proxy server in that domain.

With its new IP address, and the SIP information of the Proxy server that it got from the DHCP_OFFER message, the MN registers to the Proxy with a

SIP_REGISTER message. This message carries the SIP address (that never changes) and IP address (that changes as the node keeps roaming) information of the MN. If the Proxy server can make the registration successfully, it replies with a SIP_REGISTER_OK message. Now that the MN is registered, it can initiate sessions by inviting nodes using their SIP addresses, and it can be invited to sessions.

A registered MN can invite another registered node by transmitting INVITE messages. The INVITE message that contains the SIP address of the correspondent node (CN) is transmitted to the Proxy server by the MN. The Proxy server finds out the location of the CN and transmits the invitation to it. If the CN wishes to be a part of the conversation, it transmits an ACK to the MN and data transfer begins between these two nodes.

A MN checks its connection with its Proxy server by sending a PROXY_PING message periodically. The Proxy answers this message with PING_OK. If the MN can not receive any PING_OK message for a while, it understands that it has gone beyond the reach of its old Proxy, and starts to look for a new one.

In our simulation, MN is named as MN_Yavuz and CN is named as MN_Cakmak. Initially MN_Cakmak is in the neighborhood of Proxy1 – DHCP1 pair. When the simulation begins, MN_Cakmak gets an IP address from DHCP1 and registers to Proxy1. It keeps wandering around its initial location. It does not pass to the neighborhood of Proxy2 – DHCP2 pair. It does not send session invitation to any node.

By this time, MN_Yavuz is far away from both Proxy – DHCP pairs and it keeps on looking for a DHCP server, broadcasting DHCP_DISCOVER message. It approaches Proxy1 – DHCP1 pair first, gets an IP address from DHCP1, and registers to

Proxy1. As soon as it receives SIP_REGISTER_OK message, it invites MN_Cakmak for a conversation. MN_Cakmak accepts this invitation with an ACK. The registration and session setup messaging is shown in Figure 8.

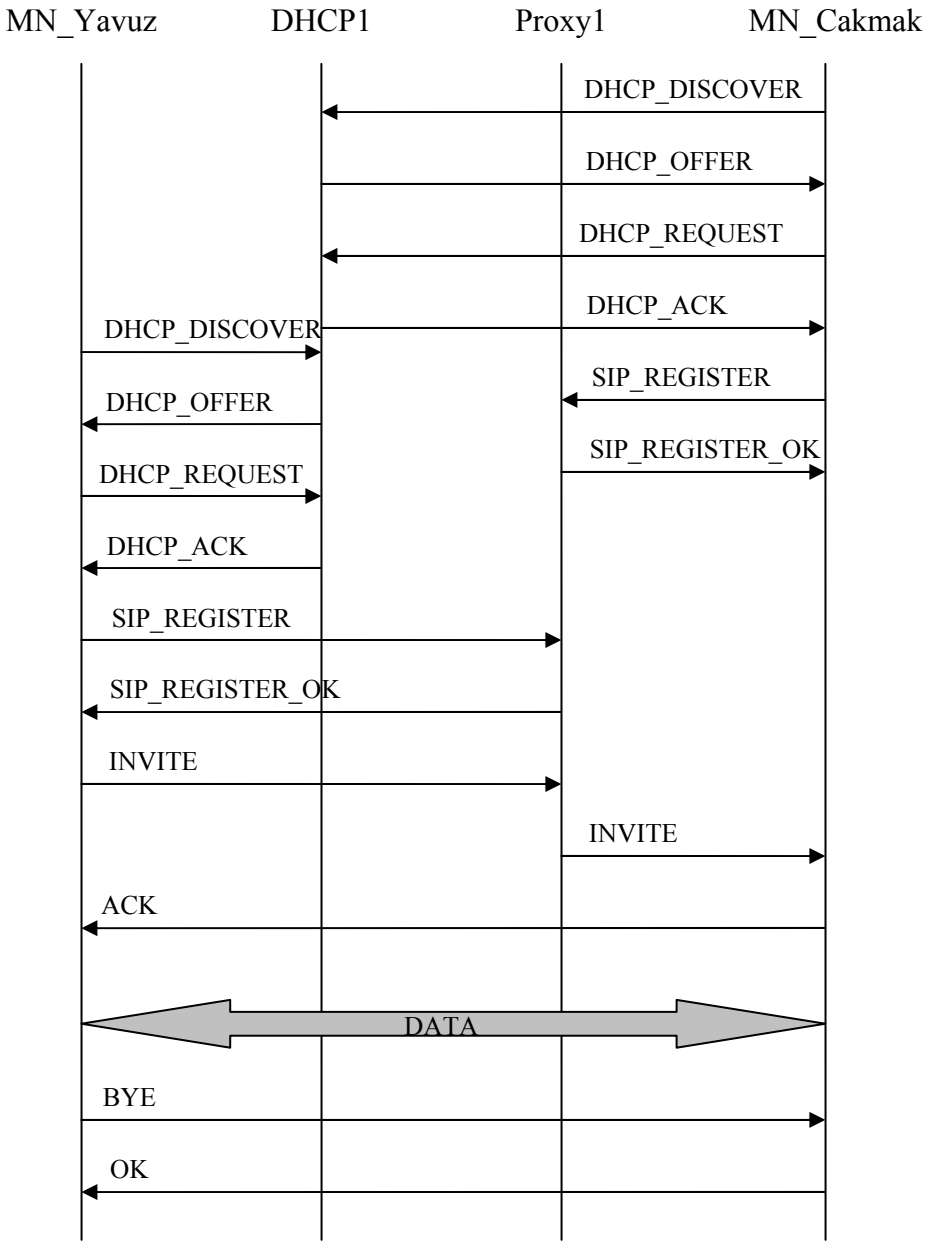

Figure 8: Registration and Session Setup MSC

MN_Yavuz keeps on transmitting PROXY_PING messages to Proxy1 periodically in order to be sure that they can still communicate. As long as MN_Yavuz receives PING_OK messages from its Proxy, it does not look for a new Proxy. However, if no PING_OK messages are received for a fixed number of consecutive PROXY_PING messages, MN_Yavuz should start a handover by broadcasting DHCP_DISCOVER messages. The handover messaging is shown in Figure 9.

While MN_Cakmak and MN_Yavuz continue their conversation, MN_Yavuz goes out of the region of Proxy1 and enters the region of Proxy2. Following this region change, MN_Yavuz begins to broadcast DHCP_DISCOVER messages. These messages are received by DHCP2 and a new IP address is offered to MN_Yavuz, including the SIP information, the IP address of the new Proxy server. Since MN_Yavuz is in the middle of a conversation with MN_Cakmak, as soon as it receives its new IP address, it sends a RE-INVITE to MN_Cakmak. This RE-INVITE message includes the new IP address of MN_Yavuz. If this handover can be done quickly, the conversation can be continued without interrupts. In order not to delay the re-inviting process, Proxy registration is done after RE-INVITE is acknowledged. The mid-session mobility messaging is shown in Figure 10**.**

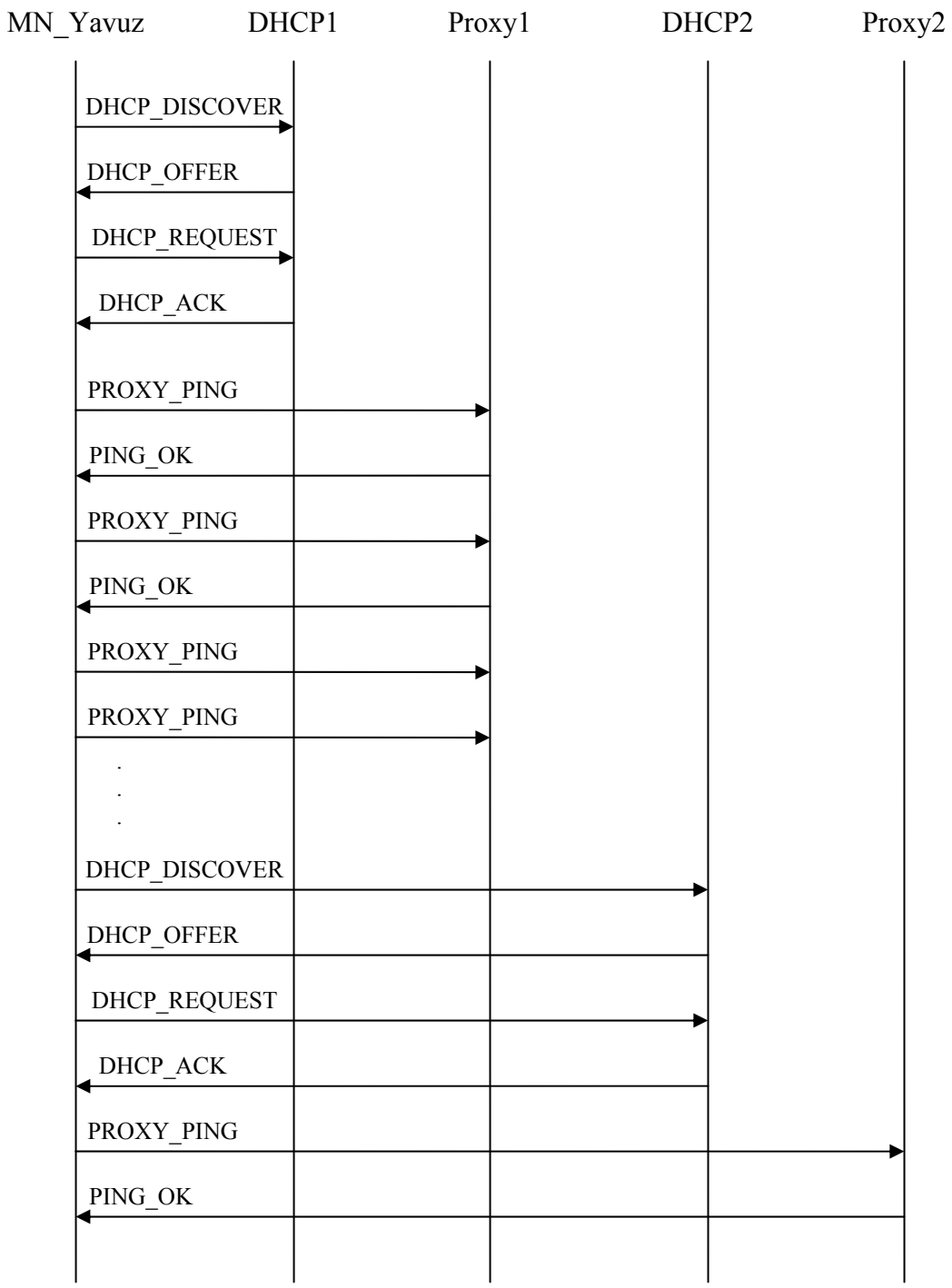

Figure 9: Handover MSC

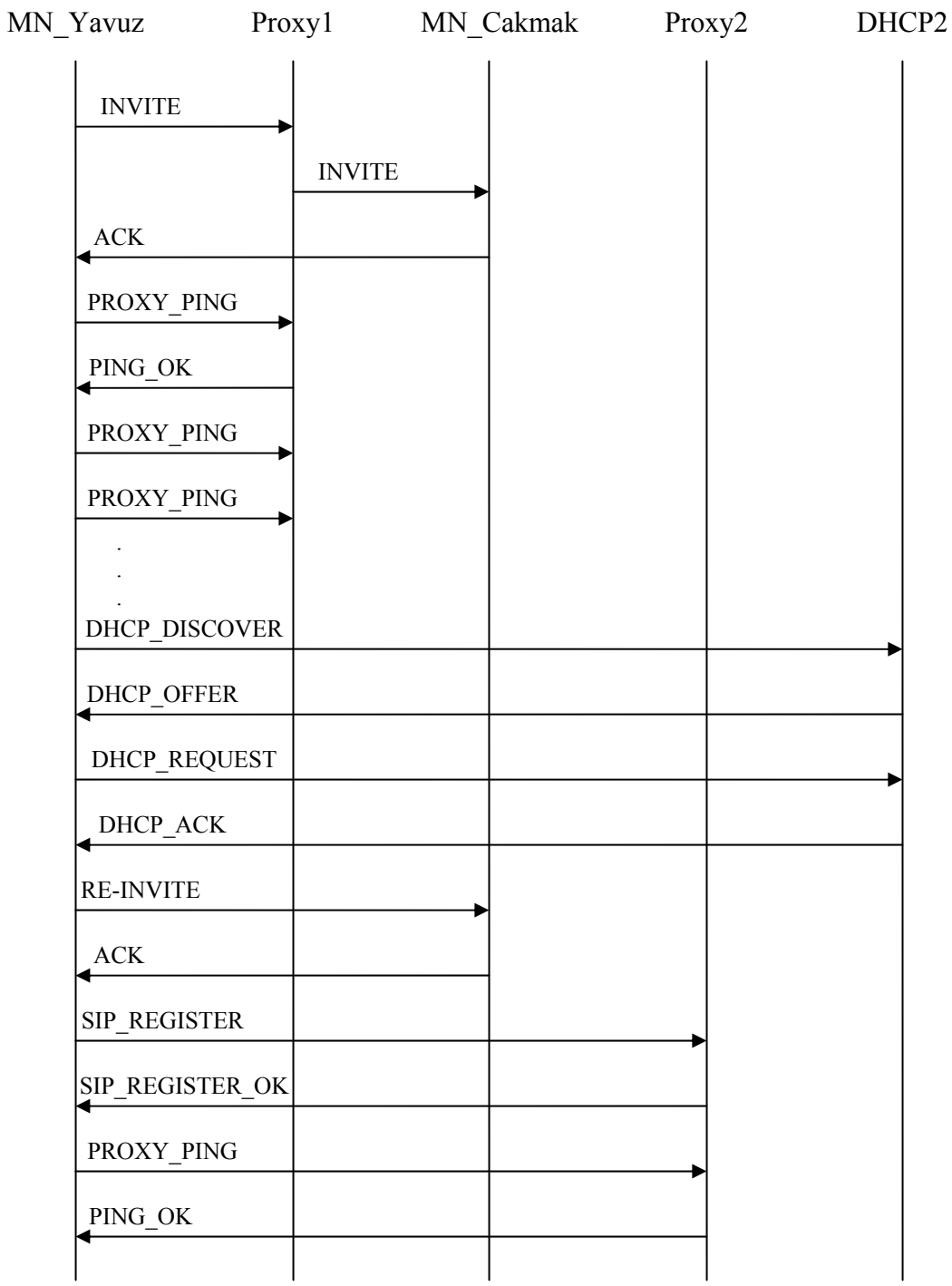

Figure 10: Mid – Session Mobility MSC

3.3. The Simulation Tool: OPNET

In this thesis work, OPNET, version 8.1.A is used [15]. OPNET is developed by OPNET Technologies Inc. It is a powerful network simulation tool that can be used to model communication systems and predict network performance. Its accuracy and ease-of-use make it a valuable tool for network planners and administrators.

OPNET provides a comprehensive development environment supporting the modeling of communication networks and distributed systems. Both behavior and performance of modeled systems can be analyzed by performing discrete event simulations. The OPNET environment incorporates tools for all phases of a study, including model design, simulation, data collection, and data analysis.

Systems specified in OPNET consist of objects, each with configurable sets of attributes. Objects belong to "classes" which provide them with their characteristics in terms of behavior and capability. Definition of new classes is supported in order to address as wide a scope of systems as possible. Classes can also be derived from other classes, or "specialized" in order to provide more specific support for particular applications.

OPNET provides many constructs relating to communications and information processing, providing high leverage for modeling of networks and distributed systems.

Simulation models in OPNET are hierarchical, naturally paralleling the structure of actual communication networks. This tool provides a flexible, high-level programming language with extensive support for communications and distributed systems. This environment allows realistic modeling of all communications protocols, algorithms, and transmission technologies.

Model specifications are automatically compiled into executable, efficient, discrete-event simulations implemented in the C programming language. Advanced simulation construction and configuration techniques minimize compilation requirements.

OPNET provides numerous built-in performance statistics that can be automatically collected during simulations. In addition, modelers can augment this set with new application-specific statistics that are computed by user-defined processes.

OPNET has got some special versions for more specific simulations. OPNET Radio Version provides specific support for mobile nodes, including predefined or adaptive trajectories; predefined and fully customizable radio link models and geographical context.

As a result of the capabilities that were described above, OPNET Radio Version 8.1.A has been selected as the simulation tool and used in this thesis work.

OPNET provides three main modeling domains for three hierarchical levels of a network. These are the Network Domain, Node Domain and the Process Domain. Each modeling domain has its own specific set of objects and operations that are appropriate for the modeling task on which it is focused.

OPNET Graphical Editors for Network, Node and Process Models is shown in Figure 11.

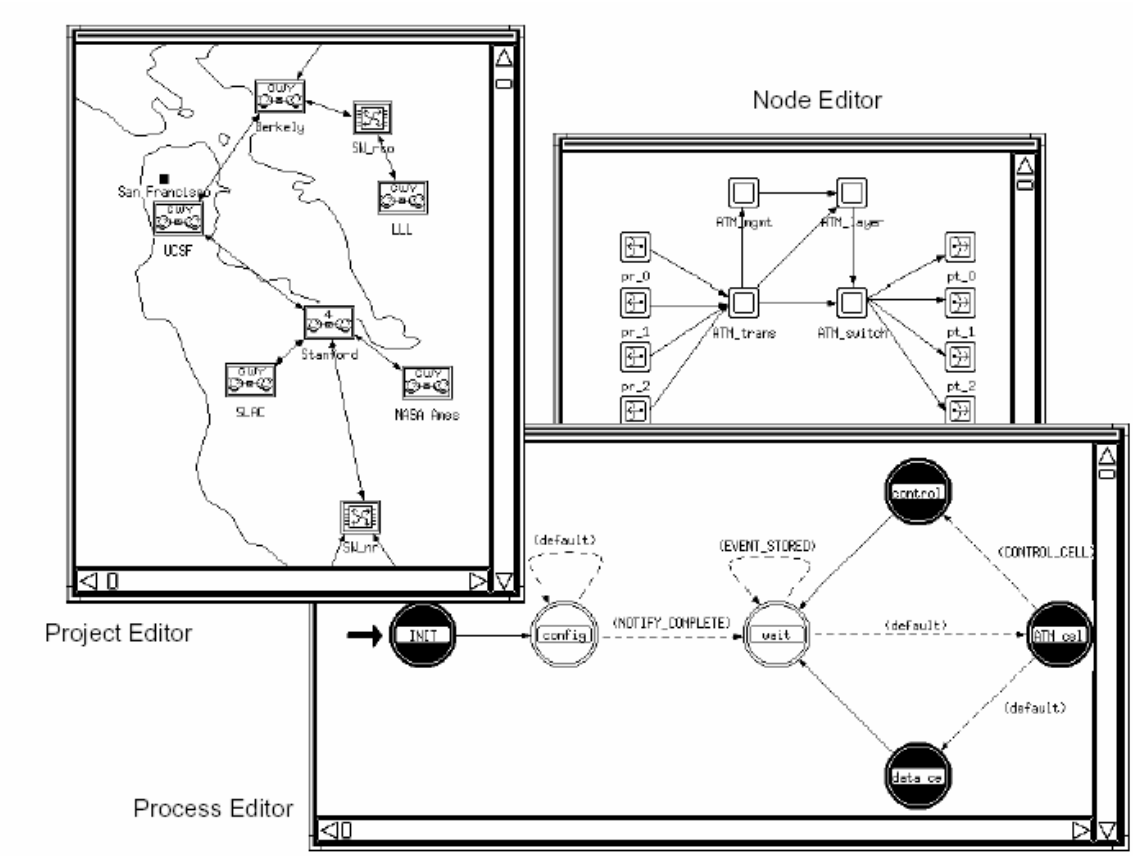

Figure 11: OPNET Graphical Editors for Network, Node and Process Models

3.3.1. Network Domain

The Network Domain's role is to define the topology of a communication network. The communicating entities are called *nodes* and the specific capabilities of each node are defined by designating their model. A network model may contain any number of nodes. Modelers can develop their own library of customized node models, implementing any functionality they require.

Network models consist of nodes and links which can be deployed within a geographical context. OPNET provides fixed nodes and point-to-point and bus links; the Radio version in addition provides mobile and satellite nodes, and radio links. While

bus and point-to-point links are modeled as explicit objects that you must create, radio links are dynamically evaluated based on characteristics of the communicating nodes.

To break down complexity and to simplify network protocols and addressing, many large networks make use of an abstraction known as a *subnetwork*. A subnetwork is a subset of a larger network's devices that forms a network in its own right. OPNET provides fixed, mobile, and satellite subnetworks to enhance network models. These subnetworks can then be connected by different types of communication links, depending on the type of subnetwork. The larger network can then be viewed as a network of its subnetworks. This abstraction can be carried out at many levels. In other words, one can form networks of subnetworks, which in turn are formed of other subnetworks, and so on. At the bottom of this hierarchy, the lowest level subnetwork is composed only of nodes and links, but no other subnetworks.

3.3.2. Node Domain

The Node Domain provides nodes for the modeling of communication devices that can be deployed and interconnected at the network level. In the real world nodes may correspond to various types of computing and communicating equipment such as routers, bridges, workstations, terminals, mainframe computers, file servers, fast packet switches, satellites, and so on.

Nodes are expressed in smaller building blocks called *modules*. Some modules offer capability that is substantially predefined and can only be configured through a set of built-in parameters. These include various transmitters and receivers allowing a node to be attached to communication links in the network domain. Other modules, called

processors and *queues*, are highly programmable, their behavior being prescribed by an assigned process model.

3.3.3. Process Domain

As mentioned earlier in the discussion of the Node Domain, queue and processor modules are user-programmable elements that are key elements of communication nodes. The tasks that these modules execute are called *processes*. A process can in many ways be thought of as similar to an executing software program, since it includes a set of instructions and maintains state memory.

OPNET places no limits on the number of processes that may be created in a particular processor or queue. Processes may be created and destroyed based on dynamic conditions that are analyzed by the logic of the executing processes. This paradigm provides a very natural framework for modeling many common systems. In particular, multitasking operating systems where the root process represents the operating system itself and the dynamically created processes correspond to new tasks; and multi-context protocols where the root process represents a session manager, for example, and each new session that is requested is modeled by creating a new process of the appropriate type.

Only one process may be *executing* at any time. A process is considered to be executing when it is progressing through new instructions that are part of its process model. When a process begins execution it is said to be *invoked*. A process that is currently executing can invoke another process in its process group to cause it to begin executing. When this happens, the invoking process is temporarily suspended until the invoked process *blocks*. A process blocks by indicating that it has completed its

processing for its current invocation. Once the invoked process has blocked, the invoking process resumes execution where it had left off, in a manner similar to the procedure-call mechanism in a programming language such as C.

Processes in OPNET are designed to respond to interrupts and/or invocations. *Interrupts* are events that are directed at a process and that may require it to take some action. They may be generated by sources external to a process group, by other members of a process group, or by a process for itself. Interrupts typically correspond to events such as messages arriving, timers expiring, resources being released, or state changes in other modules. Once a process has been invoked due to an interrupt, it may invoke other processes in the group and these may in turn invoke other processes. An interrupt's processing is completed when the first process that was invoked blocks.

3.4. Simulation Models

The nodes, the topology, the scenario and the messages that should be passed between nodes are clearly specified, and the network is set up using OPNET. One of the major advantages of OPNET is that, it has got a vast library of built-in models. There is already a mobile node model, which uses all of the OSI layers.

Since SIP is an application layer protocol that runs over UDP, it is placed on top of UDP in OPNET's built-in mobile node model. The TCP layer has been erased. The built-in mobile node model is shown in Figure 12 and the node model of MN_Yavuz is shown in Figure 13. The Proxy server and DHCP server node models are shown in Figure 14 and Figure 15 respectively. They have been inherited from the MN model; therefore these three node types are very similar to each other.

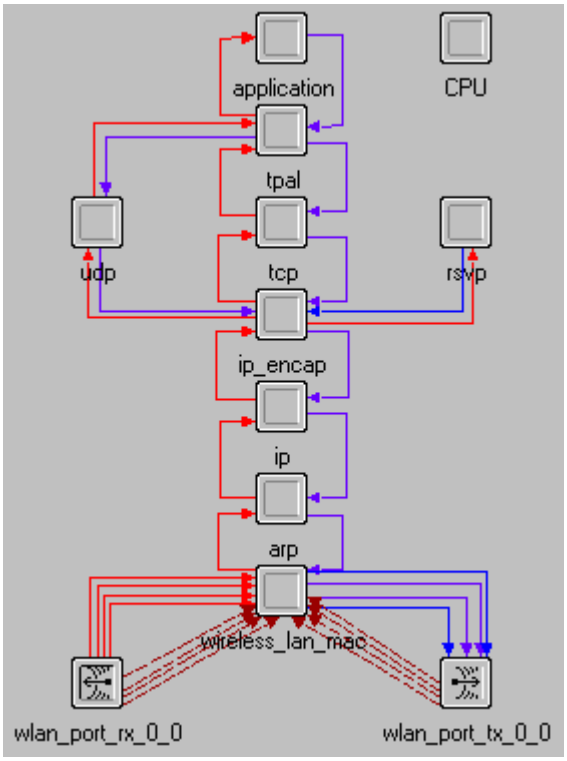

Figure 12: OPNET Mobile Node Model

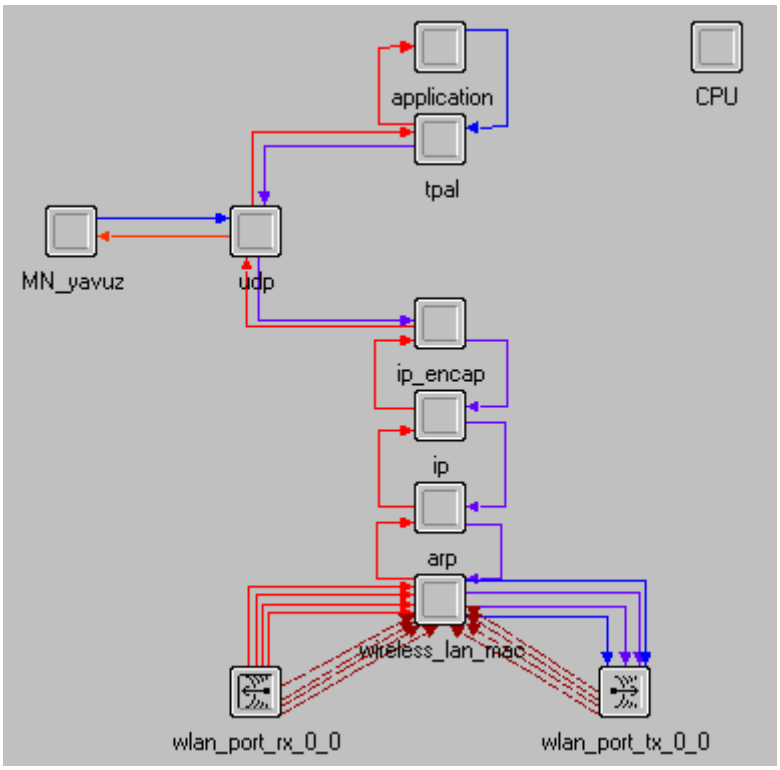

Figure 13: Node Model of MN

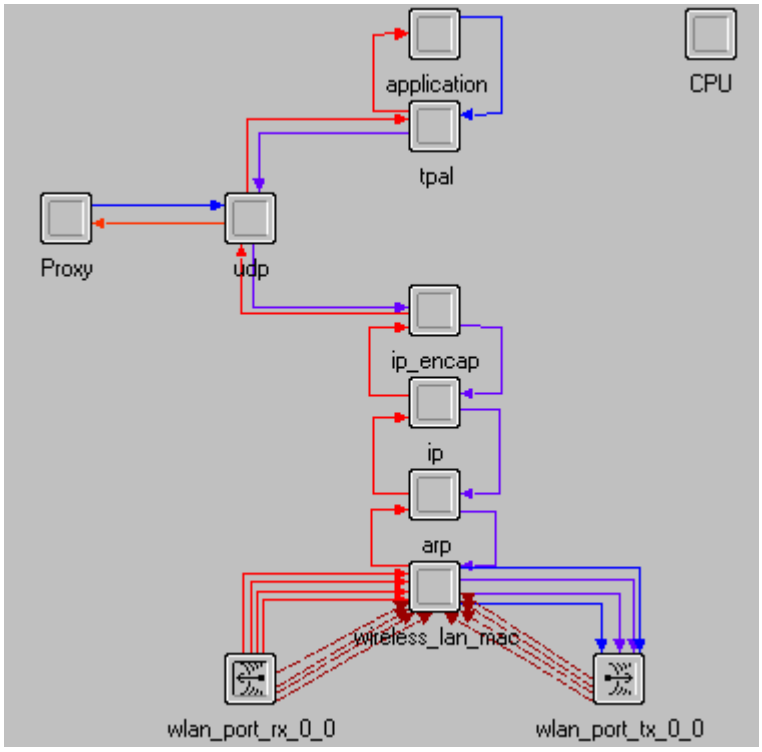

Figure 14: Node Model of Proxy Server

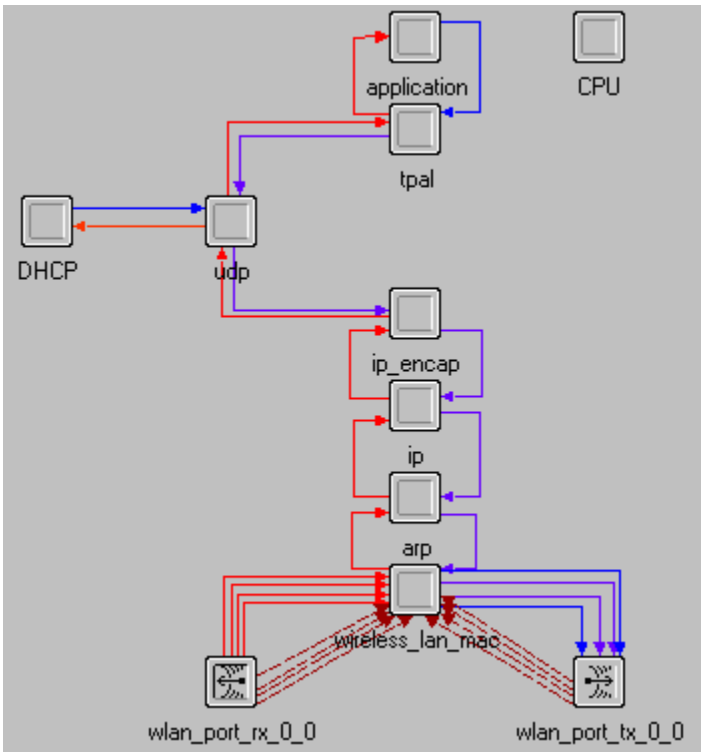

Figure 15: Node Model of DHCP Server

In this simulation, some models are used as available in OPNET, some models have been modified, and some models are implemented exclusively within the scope of the study. These models are listed below:

1) Models used as available in OPNET library:

- **wlan_port_rx process model**
- **wlan_port_tx process model**
- **wlan_mac process model**
- **ip_arp_v4 process model**
- **ip_encap process model**

2) Models modified within the study, and the modifications:

- **ip_dispatch process model:** Any mobile node used to assign an IP address to itself in this process model. However in our simulation, the IP addresses should be given to the mobile nodes via our DHCP server. Since each MN has a unique ID in the simulation, this ID is used to determine the IP address of each MN. For example: MN that has an ID of 12 has an IP address such as “177.77.1.12”, MN 8 has an IP address such as “177.77.1.8”. As long as the “seed” variable (that is used to determine a set of random number required in a simulation in OPNET) is fixed, these IDs do not change. Any MN that initiates itself transmits its ID to the DHCP server, and receives its IP address.

- **rip_udp_v3 process model:** Any packet ariving from a lower layer is destined to the process model that has been added for the simulation. This process model can be Proxy, MN or DHCP. These packets used to be destined to the application layer of the original mobile node before this modification. An ICI (Interface Control Information) has been defined in order to control the arriving packets.

3) Process Models that are implemented exclusively within the scope of this study:

- **MN process model (two instances, MN_Yavuz and MN_Cakmak, have been used)**
- **DHCP process model**
- **Proxy process model**
- **MN trajectories**

The process models of DHCPs are composed of five states. Most of the time is spent in the "wait" state. Messages that arrive from the lower layer (UDP) generate PK_ARRVL event, and these messages are received and processed in the "discard" state. When there is a message that has to be sent to the lower layer, PACKET_GENERATE event is generated. Messages are sent in the "generate" state. The process model of DHCP1 is shown in Figure 16.

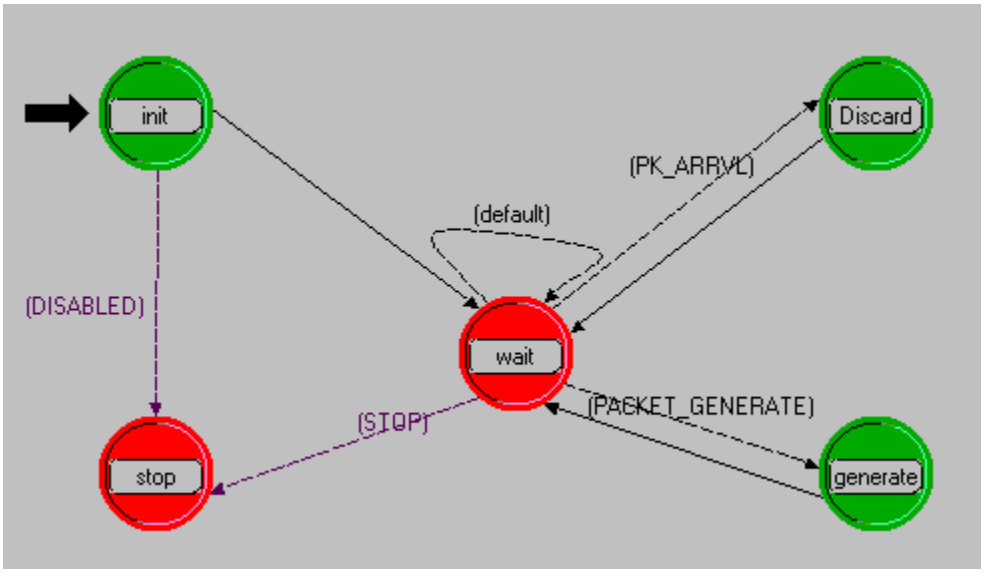

Figure 16: Process Model of DHCP Server

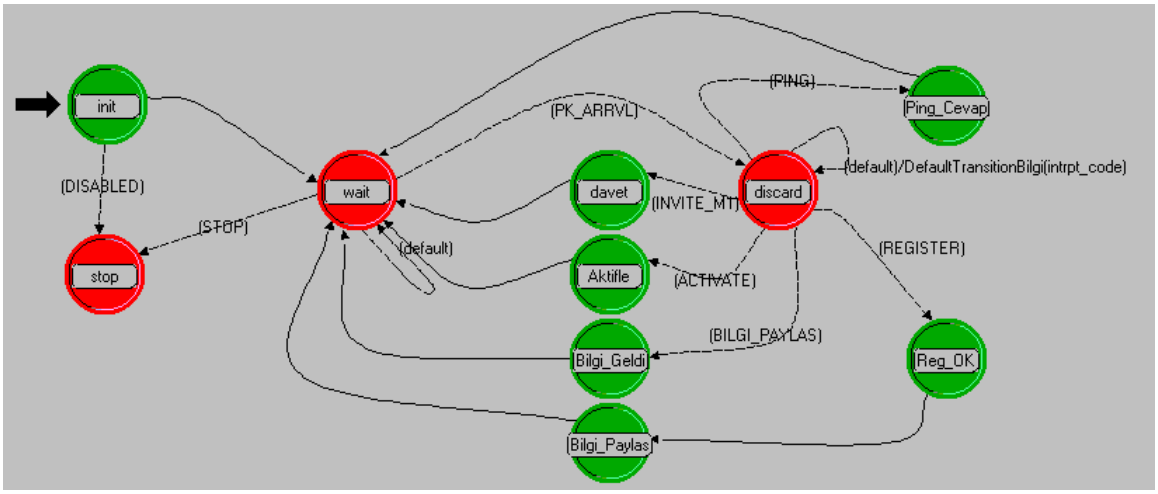

Figure 17: Process Model of Proxy Server

The process models of Proxies are composed of ten states. The logic of the statechart is quite similar to the DCHP Server statechart. Most of the time is spent in the "wait" state. Messages that arrive from the lower layer (UDP) generate PK_ARRVL event, and these messages are received and processed in the "discard" state. Depending on the type of the message, a pre-defined delay time is spent in "discard" state and then an appropriate event is generated (ACTIVATE, INVITE_M1 etc). The proper reply is prepared and transmitted within the new state, and Proxy starts to wait for a new message in "wait" state. The process model of Proxy1 is shown in Figure 17.

The process model of a MN is more complicated since the operations of a MN are much more complex than a DHCP or Proxy. The statechart is composed of twenty four states. The previous statecharts had only one "discard" state, but this one has eight "discard" states. The reason is that, the MN expects different packet types during the simulation. For example, while the MN is transmitting DHCP_DISCOVER messages, it only can receive DHCP_OFFER messages. All other kinds of messages are discarded in state "discard1". The state "discard3" only accepts "REGISTER_OK" messages from a Proxy Server. The statechart is designed to make several handovers. When the MN can not receive PING_OK message from its Proxy for a fixed amout of time, it generates itself HANDOVER event and goes through a handover.

The process model of a MN is shown in Figure 18.

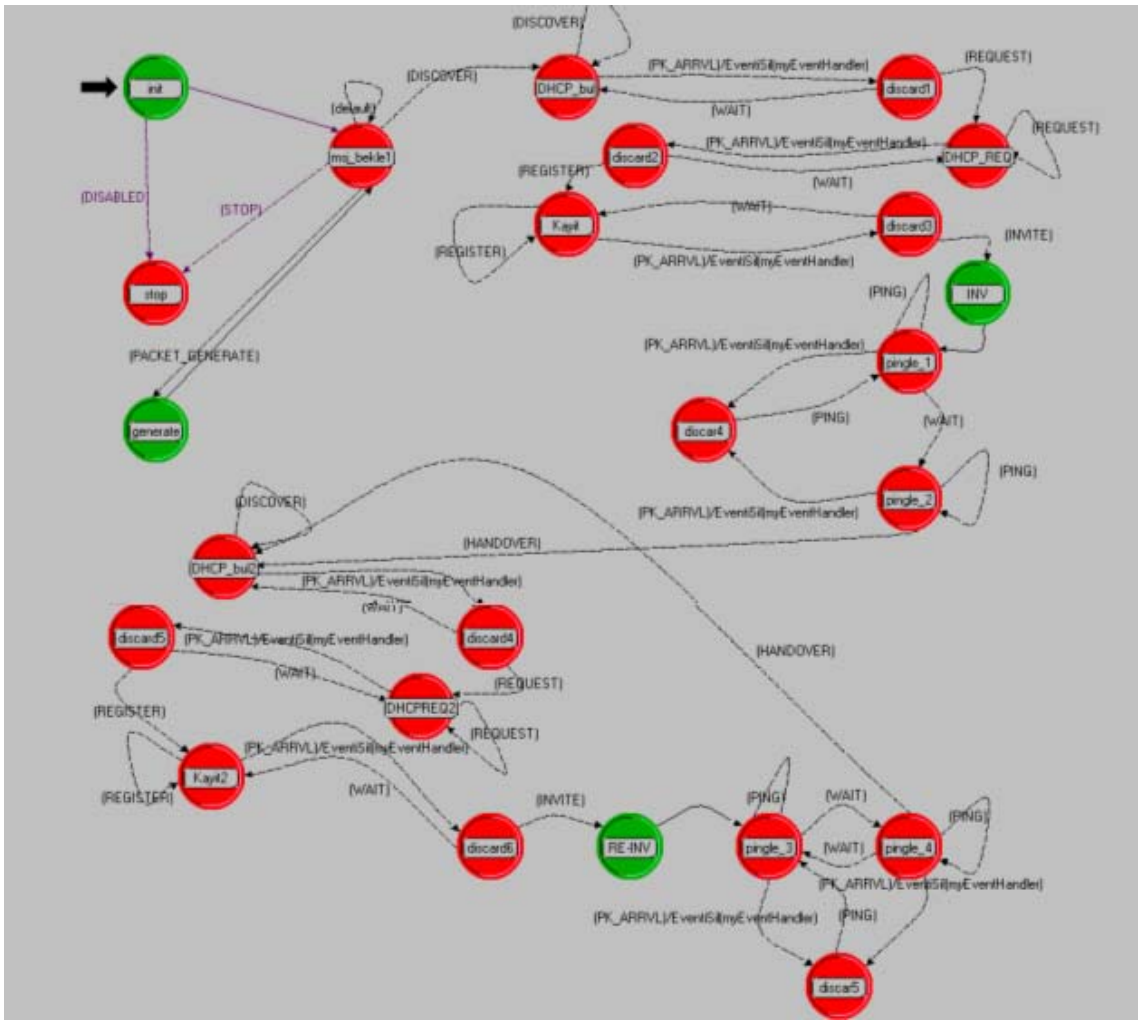

Figure 18: Process Model of MN

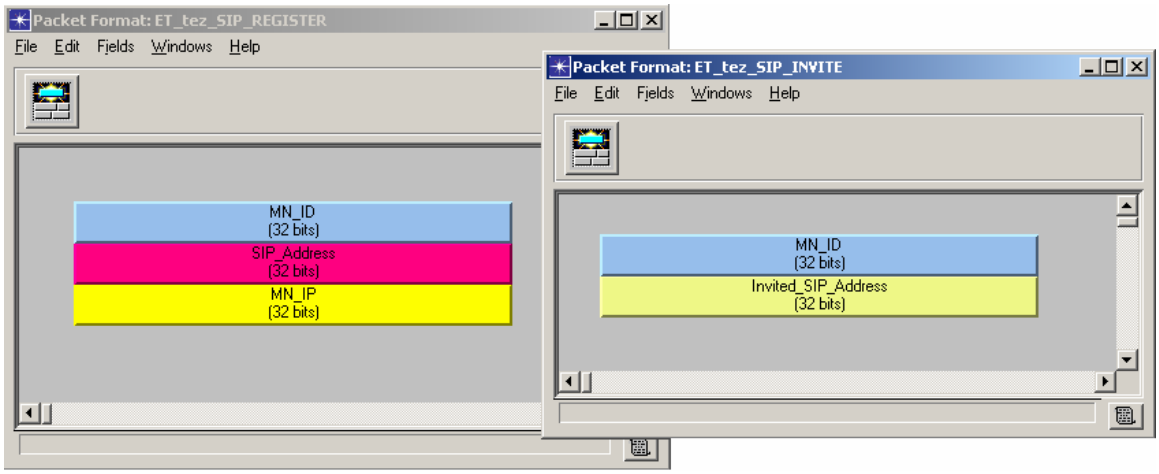

Figure 19: Packet Formats

The messages in OPNET are carried in user-defined packets. The packet formats are prepared by adding as many blocks as the user wants. The size of the blocks, and the data type that each block carries are also defined by the user. Although the size of each packet can be defined as the sum of the sizes of its blocks, the user can specify a size to each packet, during creation. This size should not be less than the sum of the sizes of its blocks. In this simulation, the size of each packet is specified as shown in Table 2. SIP_REGISTER and SIP_INVITE packet formats are shown in Figure 19.

The network topology of the simulation in the OPNET project editor is shown in Figure 20. Proxies and DHCP servers are stationary although they also are mobile nodes. The lines with arrows are the trajectories that MNs follow. MN_Cakmak gets close to DHCP2 – Proxy2 pair but it can still ping Proxy1, therefore does not register to Proxy2. MN_Yavuz loses its contact with Proxy1 after it passes to the right hand side of Proxy2. It then gets a new IP from DHCP2 and registers to Proxy2.

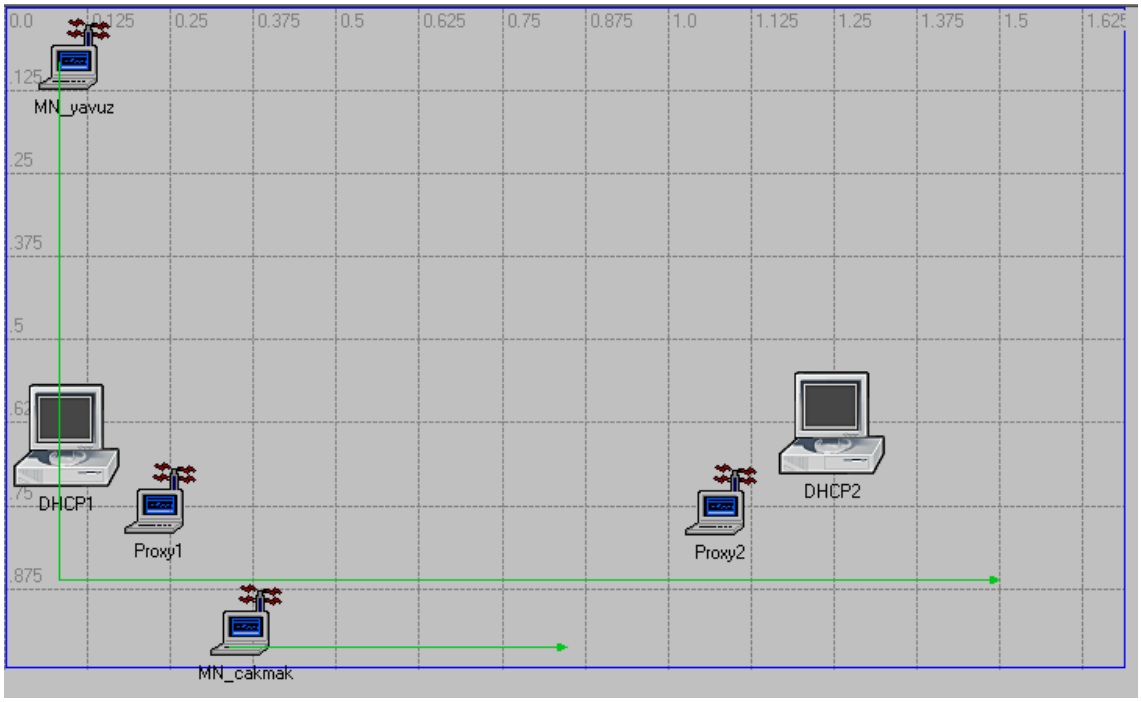

Figure 20: Simulation Scenario with one MN

3.5. Simulation Timings and Validation

The network topology is set up, similar to the testbed shown in Figure 7**.** The lengths of the messages are set as shown in Table 2. The MN_Yavuz and MN_Cakmak add a delay of 100 msec. to every message because the processing time on Linux based MNs in [1] is this much. The Proxy Server also adds a delay of 100 msec (time for database access operations) to every message except PING_OK. It does not add any extra delay to PING_OK message because there needs no database access to transmit PING_OK. If PROXY_PING message can be received by Proxy, PING_OK is transmitted as soon as possible.

During the simulation, the speed of MN_Yavuz is set as 50 km/h. This speed is considered to be representative of a car's speed in the city. With this speed, MN_Yavuz

discovers DHCP1 and registers to Proxy1 within 4th second of the simulation. It sends INVITE to MN_Cakmak, who has already registered to Proxy1 within 2nd second of simulation. This invitation is accepted with an ACK message by MN_Cakmak. MN_Yavuz also sends PROXY_PING to Proxy1 with a period of 250 msecs while it keeps going away from Proxy1 closer to Proxy2.

When the simulation time is in 80th second, MN_Yavuz can no longer get PING_OK from Proxy1. It therefore discovers DHCP2 (DHCP_ACK arrives at 80.353) re-invites MN_Cakmak (RE-INVITE is transmitted at 80.453, OK is received at 80.561) and re-registers to Proxy2 (RE-REGISTER is transmitted at 80.453, OK is received at 80.564). The re-registration time is 111 msec. It is less than the re-registration time in [1] which is about 150 msec because the latency is lower in our simulation. The latency measured in our simulation is about 5-10 msec. at most. In the testbed [1], it increases up to 70 msec. probably because of the physical distance, unspecified. Moreover, there will be more mobile nodes following similar paths to MN_Yavuz in the following scenarios, and re-registration delays will definitely increase.

The testbed has been simulated and verified, so the node models and the network topology is ready for further experiments. The number of mobile nodes can be increased in order to increase the amount of packet traffic, and the speed of the mobile nodes can be changed. New methods can be developed and tested to reduce the re-invite or re-registration time.

3.6. An Improvement for the Re-Registration Time

If the mobile node makes a handover during a conversation, it has to get a new IP address and then send a re-invite to the correspondent node as soon as possible. Afterwards, it re-registers to the SIP Proxy server in that region. The re-registration time can be shorter if the new proxy has already got the information about that node.

This concept is similar to the *shadow registration* concept given in [8]. The key idea in the shadow registration is to establish a registration status in the neighboring administrative domains a priori anticipating possible handoffs when the user registers in a given wireless/mobile network. However, this shadow registration concept applies to only the AAA (Authentication, Authorization and Accounting) servers. It does not bring any improvement to SIP Proxy re-registration time.

In this thesis, we propose a similar concept, for the MN to re-register to its new Proxy server. We call it "registration – activation". With this concept, when the MN registers to a Proxy server in the network, this Proxy transmits its registration information to all of the Proxy servers in its neighborhood. The neighboring proxies record this information, but since the MN is not in their domain yet, they mark it as "inactive". The MN is marked as active when it enters the domain of one of these Proxy servers, and transmits SIP_ACTIVATE message. Since SIP_ACTIVATE message does not need to carry much information about the node (only the SIP address), it is much smaller than the SIP_REGISTER message. MN re-registration time is therefore decreased. The registration – activation messaging is shown in Figure 21.

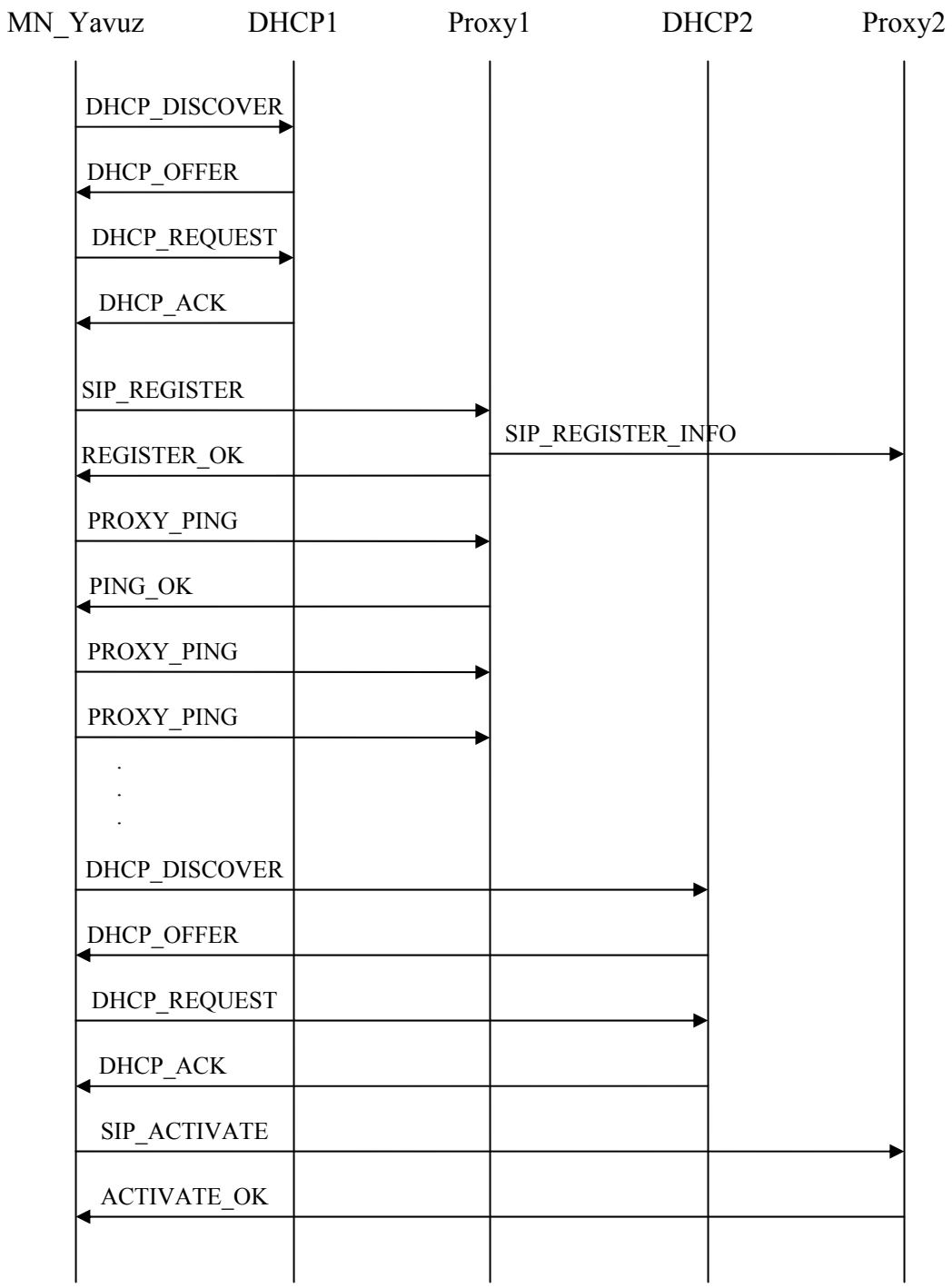

Figure 21: Registration – Activation MSC

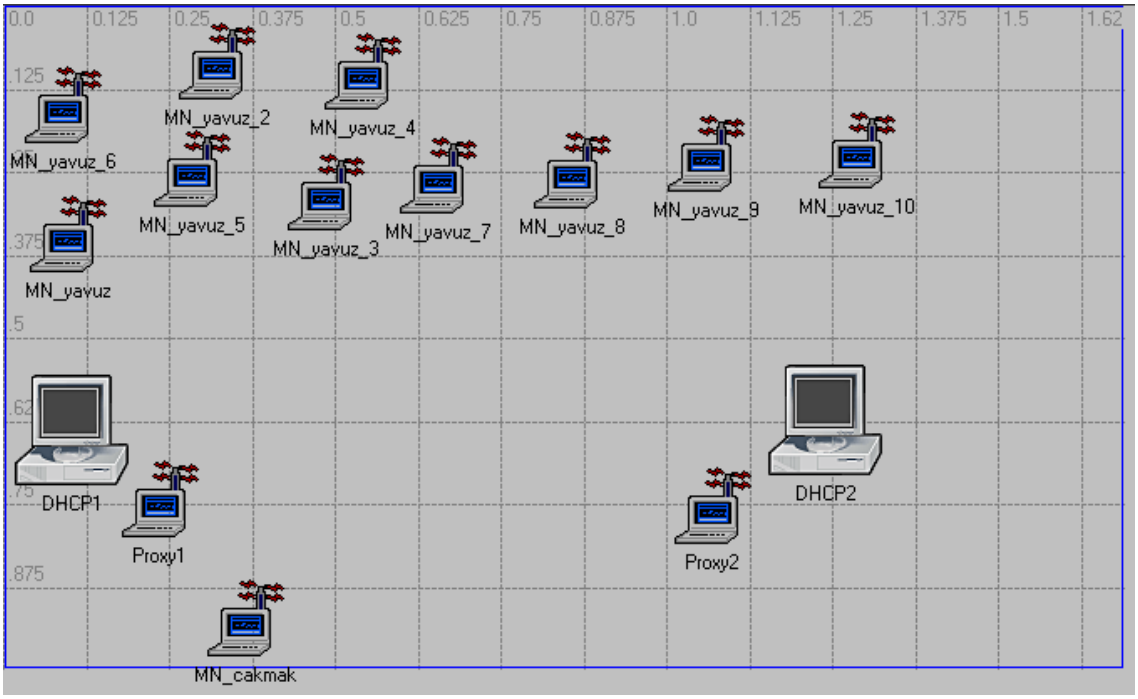

Figure 22: Simulation Scenario with 10 MNs

The simulation topology has been changed in order to test the "registration – activation". There are 9 new MNs, which are identical to MN_Yavuz, except the invitation of MN_Cakmak. Since the improvement is for the re-registration delay, the new MNs do not invite any node for a session. They follow random trajectories, register to one of the Proxies, and when they lose contact; they re-register to the other Proxy. The new topology is shown in Figure 22. The trajectory followed by each MN constitutes a random walk generated differently over each simulation run. Every 5 msec, each MN determines its movement direction according to a uniform distribution.

The improvement that "registration – activation" brings is the decreased size of the message in re-registration. The length of the RE-REGISTER message is 425 bytes, and the length of the ACTIVATE message is the length of the MN ID, a short integer,

16 bytes. Moreover, the database access of the Proxy Server for activation also takes a short time. All the required data of MN has been recorded to the database when SIP_REGISTER_INFO message has arrived. The MN only transmits its ID to be checked in the database of Proxy. For this reason, the delay added for SIP_REGISTER message in Proxy is 100 msec. and the delay added for SIP_ACTIVATE message is 90 msec or 75 msec.

3.7. Results of the Improvement

There have been several experiments with many different parameter sets in order to test the improvement that "registration – activation" brings. The period that MNs transmit PROXY_PING, the number of MNs in the experiment, the speed of the MNs, the delay added by MNs and Proxies are the parameters that affect the results.

There have been experiments with 1, 4, 7 and 10 nodes in order to see the effect of heavy traffic over Proxies. As the traffic increases, the re-registration delays are increased drastically and the "registration – activation" brings significant improvement to these delays. When the speed of each MN is 50 km/h, the re-registration delays are lower than those with the speed of 5 km/h. The reason for this difference can be explained by the location of DHCP and Proxy pairs. When a MN can communicate with a DHCP Server, it can not necessarily communicate with its Proxy Server. These servers are located about 200 meters apart from each other. When the speed of a MN is high, this distance is negligible. The MN starts to communicate with both Proxy and DHCP Server at the same time. However, when the MN is slow, the MN may have to travel a few more seconds to communicate with the Proxy. Therefore, the delay observed in slow MNs re-registration is a few seconds longer.

The most significant results are presented in the following subsections. The re-registration delays averaged over 10 simulation runs, over all nodes and over all handovers are shown in Figure 23 and Figure 24.

3.7.1. MN Speed: 50km/h

The set of the parameters are listed in configurations 1, 2 and 3. The average number of handovers that each MN faces over 10 runs and the average re-registration delays over these handovers is shown in tables in the appendix. If "registration – activation" is applied, the transmission time of ACTIVATION and the reception time of OK messages are shown instead. The simulations have been run for 10 times for each configuration and for each number of nodes.

Configuration 1:

Node Speed:	50 km / h
MN Operation Time:	100 msec
MN Ping Period :	250 msec
Proxy Ping Answer Delay:	0 msec
Simulation Duration:	1500 sec
Number of simulation runs:	10
Proxy Registration Delay:	100 msec
Registration – Activation:	Not Applied

Table 3: Summary of Results of Configuration 1

Number of Nodes	1	4	7	10
Re-registration Time	109 msec	2966 msec	4565 msec	5056 msec

Configuration 2:

Node Speed:	50 km / h
MN Operation Time:	100 msec
MN Ping Period:	250 msec
Proxy Ping Answer Delay:	0 msec
Simulation Duration:	1500 sec
Number of simulation runs:	10
Proxy Registration Delay:	100 msec
Proxy Activation Delay:	90 msec
Registration – Activation:	Applied

Table 4: Summary of Results of Configuration 2

Number of Nodes	1	4	7	10
Re-registration Time	96 msec	1895 msec	2956 msec	3168 msec

Configuration 3:

Node Speed:	50 km / h
MN Operation Time:	100 msec
MN Ping Period:	250 msec
Proxy Ping Answer Delay:	0 msec
Simulation Duration:	1500 sec
Number of simulation runs:	10
Proxy Registration Delay:	100 msec

Proxy Activation Delay: 75 msec

Registration – Activation: Applied

Table 5: Summary of Results of Configuration 3

Number of Nodes	1	4	7	10
Re-registration Time (average over nodes, over handovers)	82 msec	755 msec	1563 msec	2133 msec

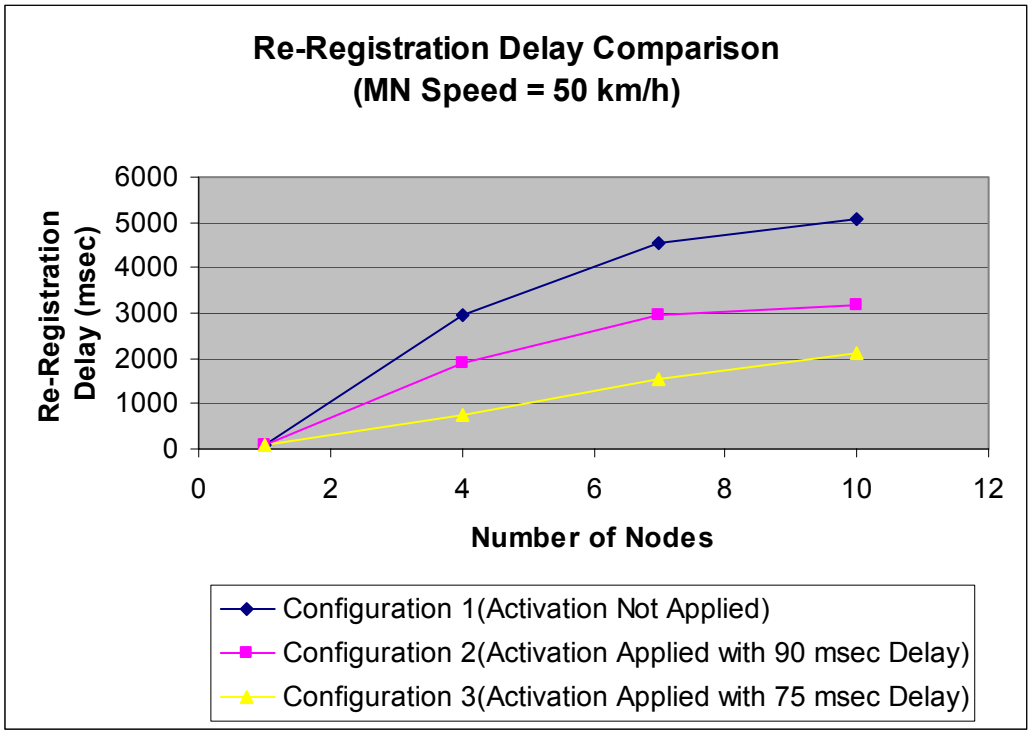

Figure 23: Re-Registration Delay Comparison for 50 km/h MN speed

3.7.2. MN Speed: 5 km/h

The set of the parameters are listed in configurations 1, 2 and 3. The average number of handovers that each MN faces over 10 runs and the average re-registration delays over these handovers is shown in tables in the appendix. If "registration – activation" is applied, the transmission time of ACTIVATION and the reception time of OK messages are shown instead. The simulations have been run for 10 times for each configuration and for each number of nodes.

Configuration 1:

Node Speed: 5 km / h

MN Operation Time: 100 msec

MN Ping Period: 250 msec

Proxy Ping Answer Delay: 0 msec

Simulation Duration: 6000 sec

Number of simulation runs: 10

Proxy Registration Delay: 100 msec

Registration – Activation: Not Applied

Table 6: Summary of Results of Configuration 1

Number of Nodes	1	4	7	10
Re-registration Time	108 msec	4865 msec	8025 msec	8659 msec

Configuration 2:

Node Speed:	5 km / h
MN Operation Time:	100 msec
MN Ping Period:	250 msec
Proxy Ping Answer Delay:	0 msec
Simulation Duration:	6000 sec
Number of simulation runs:	10
Proxy Registration Delay:	100 msec
Proxy Activation Delay:	90 msec
Registration – Activation:	Applied

Table 7: Summary of Results of Configuration 2

Number of Nodes	1	4	7	10
Re-registration Time	94 msec	2860 msec	6008 msec	6985 msec

Configuration 3:

Node Speed:	5 km / h
MN Operation Time:	100 msec
MN Ping Period:	250 msec
Proxy Ping Answer Delay:	0 msec
Simulation Duration:	6000 sec
Number of simulation runs:	10
Proxy Registration Delay:	100 msec

Proxy Activation Delay: 75 msec

Registration – Activation: Applied

Table 8: Summary of Results of Configuration 3

Number of Nodes	1	4	7	10
Re-registration Time	79 msec	1940 msec	3708 msec	4402 msec

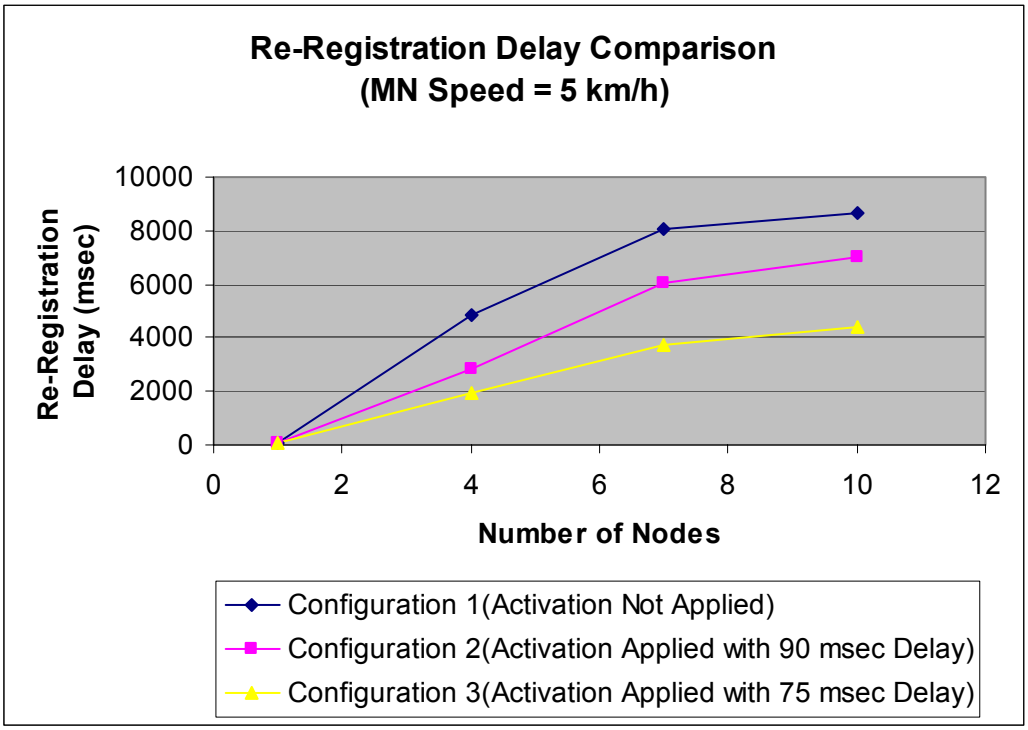

Figure 24: Re-Registration Delay Comparison for 5 km/h MN speed

CHAPTER IV

CONCLUSIONS

In this thesis work, an improvement has been proposed and simulated in order to decrease the re-registration time of mobile nodes in a wireless mobile SIP environment.

To achieve this goal, the basics of SIP operation, entities and messages are studied. Many papers that discuss the mobility support of SIP are collected and studied. The testbed that has been set in [1] in order to test the terminal mobility performance of a roaming user in a SIP signaling scheme is given particular attention. In the scope of this thesis, this testbed has been simulated in a network simulation program called OPNET.

The simulation of the testbed in [1] has been validated by using the same topology, same number of mobile nodes, proxy servers, same message lengths and obtaining similar delay times. Once the simulation is validated, new methods could be developed and tested to reduce the re-invite or re-registration time.

The new concept that has been proposed, namely the “registration – activation” intends to bring improvement to the re-registration delay of a mobile node. Once a mobile node registers to a proxy server, this proxy informs all the neighboring proxies about this node. The neighboring proxies record the registration data of the MN, but

mark it as “inactive” since the MN is not in their neighborhood yet. When the MN makes a handoff, the new proxy already has got the node’s registration information, therefore only activates it in a shorter time compared to a registration. This concept brings improvement in two ways: First, the ACTIVATE message is shorter than REGISTER message, since it does not need to contain much information about the node. Second, the Proxy takes a shorter time to access its database to activate a node, compared to registering it, because less data has to be recorded.

From the simulation results, it is observed that the re-registration delay is improved by applying the “registration – activation” but the percentage of improvement definitely depends on the improvement in the database access delay in the Proxy server.

This study tries to highlight some of the issues involved with the handover delays of application layer mobility management using SIP. SIP is an easy protocol to understand and simulate. The improvement presented in this study is also easy to implement, since it does not require any change to mobile users. This improvement may bring SIP to a better position in the competition between network layer mobility support and application layer mobility support.

In general, while this work has intended to provide a preliminary study of improving the re-registration delay time of mobile SIP networks, much work remains to be done in the direction of determining how to do this most effectively and efficiently.

The database access delay of the Proxy for activating an arriving MN has been assumed as 75% (in configuration 2) and 90% (in configuration 3) of the delay for registering a MN. This assumption has to be validated by experiments and measurements on actual database access.

As a further study, more Proxy – DHCP pairs can be added to the topology and the random trajectories of MNs can be longer to observe a higher number of handovers. The simulation duration can be longer to achieve this goal.

Network layer mobility support (i.e. Mobile IP) can be added to the simulation, by modifying the network layer model provided by OPNET. Simulation based comparison of Mobile IP handover issues and SIP mobility support could bring new clues to ongoing discussions in this platform.

REFERENCES

[1] A. Dutta, S. Baba, H. Schulzrinne et. al. "Application Layer Mobility Management Scheme for Wireless Internet", 3Gwireless 2001, San Francisco, pp 7 - 14, May 2001.

[2] A. Dutta, H. Schulzrinne, "Multilayered Mobility Management for Survivable Network", Milcom, Vienna, Virginia, November 2001.

[3] E. Wedlund, H. Schulzrinne, "Mobility support using SIP", Proc. The Second ACM International Workshop on Wireless Mobile Multimedia, ACM/IEEE, August 1999, pp 76 – 82.

[4] M. Handley, H. Schulzrinne, E. Schooler and J. Rosenberg, "SIP: Session Initiation Protocol", RFC 2543, IETF, March 1999.

[5] E. Wedlund, H. Schulzrinne, "Application-Layer Mobility Using SIP", Mobile Computing and Communications Review, Volume 1, Number 2, July 2000.

[6] M. Moh, G. Berquin, and Y. Chen, "Mobile IP Telephony: Mobility Support of SIP," Eighth International Conference on Computer Communications and Networks, 1999.

[7] H. Schulzrinne, J. Rosenberg, "The Session Initiation Protocol: Internet-Centric Signaling", IEEE Communications Magazine, October 2000, pp 134-141.

[8] T. Kwon, M. Gerla, S. Das et. al "Mobility Management for VoIP Service: Mobile IP vs. SIP", IEEE Wireless Communications, Vol. 9, No. 5, pp 66 – 75, October 2002.

[9] F. Fingal, P. Gustavsson "A SIP of IP-telephony", Master's Thesis, Lund Institute of Technology, Lund University, February,1999.

[10] F. Vakil, A. Dutta, J-C. Chen, Telcordia Technologies, S. Baba, N. Nakajima, H. Schulzrinne et. al," Mobility Management in a SIP Environment, Requirements, Functions and Issues", IETF Internet Draft, December 2000.

[11] H. Schulzrinne and J. Rosenberg, "A Comparison of SIP and H.323 for Internet Telephony", Network and Operating System Support for Digital Audio and Video (NOSSDAV), Cambridge, England, July 1998.

[12] H. Schulzrinne, J. Rosenberg, Columbia U. "Guidelines for Authors of Extensions to the Session Initiation Protocol", IETF Internet Draft, December 2002.

[13] Alcatel White Paper, "Session Initiation Protocol – SIP, Launching the IP Communication Revolution", July 2003.

[14] A. McAuley, S. Das, S. Madhani, Telcordia Technologies, S. Baba, Y. Shobatake, Toshiba America Research Inc."Dynamic Registration and Configuration Protocol (DRCP)", IETF Internet Draft, July 2000.

[15] OPNET Technologies Inc., "OPNET Modeler Documentation Version 8.1.A", USA, March 2002.

[16] N. Nakajima, A. Dutta, S. Das, H. Schulzrinne, "Handoff Delay Analysis and Measurement for SIP Based Mobility in IPv6", IEEE International Conference on Communications 2003, Volume 2, pp. 1085 – 1089.

[17] http://www.cs.columbia.edu/¨hgs/internet/, last update: 25 September 2003 by H. Schulzrinne; last access: July 2004

APPENDIX A

RE-REGISTRATION DELAY TABLES

The detailed tables that are prepared for the evaluation of the "registration – activation" improvement are presented in this appendix.

The tables obtained when each MN travels at a speed of 50 km/h are given below. The configuration details are given in Chapter 3.7.1.

Table 9: Results of Configuration 1 with 10 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	2,9	5299 msec
MN 9	3,1	4922 msec
MN 10	3,4	4219 msec
MN 11	2,8	4968 msec
MN 12	3,2	5046 msec
MN 13	2,6	5369 msec
MN 14	2,8	5330 msec
MN 15	3,3	4867 msec
MN 16	2,9	5236 msec
MN 17	3,0	5301 msec
Average Re-Registration Delay Per Node	5056 msec	

Table 10: Results of Configuration 2 with 10 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	2,8	3469 msec
MN 9	3,2	3026 msec
MN 10	2,7	3488 msec
MN 11	2,8	3302 msec
MN 12	3,3	2968 msec
MN 13	3,2	3004 msec
MN 14	3,4	3289 msec
MN 15	3,3	3055 msec
MN 16	2,9	3256 msec
MN 17	3,5	2826 msec
Average Activation Delay Per Node	3168 msec	

Table 11: Results of Configuration 3 with 10 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	2,8	2099 msec
MN 9	3,0	1986 msec
MN 10	3,1	2158 msec
MN 11	2,9	2230 msec
MN 12	3,3	2270 msec
MN 13	3,3	2308 msec
MN 14	3,1	2081 msec
MN 15	2,6	2316 msec
MN 16	2,9	1868 msec
MN 17	3,0	2017 msec
Average Activation Delay Per Node	2133 msec	

Table 12: Results of Configuration 1 with 7 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,0	4478 msec
MN 9	3,2	4752 msec
MN 11	2,8	4460 msec
MN 12	3,2	4687 msec
MN 13	3,6	4361 msec
MN 15	3,4	4525 msec
MN 17	3,1	4693 msec
Average Re-Registration Delay Per Node	4565 msec	

Table 13: Results of Configuration 2 with 7 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,1	3102 msec
MN 9	3,1	3055 msec
MN 11	2,6	2860 msec
MN 12	3,3	2958 msec
MN 13	3,4	2888 msec
MN 15	3,7	3126 msec
MN 17	3,6	2704 msec
Average Activation Delay Per Node	2956 msec	

Table 14: Results of Configuration 3 with 7 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	2,9	1399 msec
MN 9	3,4	1636 msec
MN 11	2,8	1580 msec
MN 12	3,2	1428 msec
MN 13	3,6	1716 msec
MN 15	3,9	1588 msec
MN 17	3,3	1593 msec
Average Activation Delay Per Node	1563 msec	

Table 15: Results of Configuration 1 with 4 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,3	3025 msec
MN 9	3,1	3162 msec
MN 12	3,9	2801 msec
MN 13	3,5	2876 msec
Average Re-Registration Delay Per Node	2966 msec	

Table 16: Results of Configuration 2 with 4 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,0	1767 msec
MN 9	3,3	1926 msec
MN 12	3,6	1889 msec
MN 13	3,5	1997 msec
Average Activation Delay Per Node	1895 msec	

Table 17: Results of Configuration 3 with 4 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,9	769 msec
MN 9	3,0	726 msec
MN 12	3,2	778 msec
MN 13	3,6	748 msec
Average Activation Delay Per Node	755 msec	

Table 18: Results of Configuration 1 with 1 Node

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,5	109 msec
Average Re-Registration Delay Per Node	109 msec	

Table 19: Results of Configuration 2 with 1 Node

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,7	96 msec
Average Activation Delay Per Node	96 msec	

Table 20: Results of Configuration 3 with 1 Node

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,9	82 msec
Average Activation Delay Per Node	82 msec	

The tables obtained when each MN travels at a speed of 5 km/h are given below. The configuration details are given in Chapter 3.7.2.

Table 21: Results of Configuration 1 with 10 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,3	8566 msec
MN 9	3,0	8650 msec
MN 10	2,6	8789 msec
MN 11	2,9	8503 msec
MN 12	3,0	8690 msec
MN 13	2,8	8769 msec
MN 14	3,4	8409 msec
MN 15	2,6	8787 msec
MN 16	2,8	8828 msec
MN 17	3,1	8600 msec
Average Re-Registration Delay Per Node	8659 msec	

Table 22: Results of Configuration 2 with 10 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	2,9	7030 msec
MN 9	3,4	7109 msec
MN 10	2,8	6908 msec
MN 11	3,0	6854 msec
MN 12	2,9	6801 msec
MN 13	3,1	6958 msec
MN 14	3,3	7098 msec
MN 15	2,6	7156 msec
MN 16	3,2	6799 msec
MN 17	3,1	7137 msec
Average Activation Delay Per Node	6985 msec	

Table 23: Results of Configuration 3 with 10 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	2,9	4055 msec
MN 9	3,3	4356 msec
MN 10	3,2	4386 msec
MN 11	2,8	4302 msec
MN 12	2,9	4659 msec
MN 13	3,1	4455 msec
MN 14	3,0	4610 msec
MN 15	2,9	4525 msec
MN 16	3,3	4231 msec
MN 17	3,1	4442 msec
Average Activation Delay Per Node	4402 msec	

Table 24: Results of Configuration 1 with 7 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,2	7811 msec
MN 9	2,9	7969 msec
MN 11	3,4	8125 msec
MN 12	3,0	7990 msec
MN 13	3,2	8103 msec
MN 15	2,9	8009 msec
MN 17	3,4	8169 msec
Average Re-Registration Delay Per Node	8025 msec	

Table 25: Results of Configuration 2 with 7 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,0	5968 msec
MN 9	3,2	5898 msec
MN 11	3,0	6201 msec
MN 12	2,9	6026 msec
MN 13	3,4	5926 msec
MN 15	3,5	6125 msec
MN 17	3,5	5911 msec
Average Activation Delay Per Node	6008 msec	

Table 26: Results of Configuration 3 with 7 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,1	3077 msec
MN 9	3,4	3829 msec
MN 11	2,9	3905 msec
MN 12	3,4	3699 msec
MN 13	3,6	3788 msec
MN 15	2,8	3691 msec
MN 17	3,3	3969 msec
Average Activation Delay Per Node	3708 msec	

Table 27: Results of Configuration 1 with 4 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,3	5480 msec
MN 9	3,6	4369 msec
MN 12	3,4	5022 msec
MN 13	3,8	4587 msec
Average Re-Registration Delay Per Node	4865 msec	

Table 28: Results of Configuration 2 with 4 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,1	2989 msec
MN 9	3,9	2659 msec
MN 12	3,5	2898 msec
MN 13	3,0	2894 msec
Average Activation Delay Per Node	2860 msec	

Table 29: Results of Configuration 3 with 4 Nodes

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,2	1579 msec
MN 9	3,0	1874 msec
MN 12	3,5	2096 msec
MN 13	3,8	2210 msec
Average Activation Delay Per Node	1940 msec	

Table 30: Results of Configuration 1 with 1 Node

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,6	108 msec
Average Re-Registration Delay Per Node	108 msec	

Table 31: Results of Configuration 2 with 1 Node

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,7	94 msec
Average Activation Delay Per Node	94 msec	

Table 32: Results of Configuration 3 with 1 Node

Node ID	**Average Number of Handovers (over 10 simulation runs)**	**Average Re-Registration Time**
MN 5	3,5	79 msec
Average Activation Delay Per Node	79 msec	