PERFORMANCE EVALUATION OF SIP BASED VOICE CONFERENCING OVER THE IEEE 802.11 WIRELESS NETWORKS

by

Yufen Jiang
B.Eng., Electrical Engineering,
Beijing University of Aeronautics and Astronautics, 2003

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
in the School
of
Interactive Arts and Technology

SIMON FRASER UNIVERSITY
Fall, 2006

APPROVAL

Name:	Yufen Jiang
Degree:	Master of Science
Title of thesis:	PERFORMANCE EVALUATION OF SIP BASED VOICE CONFERENCING OVER THE IEEE 802.11 WIRELESS NETWORKS

Examining Committee: Dr. Chris Shaw
Chair

Dr. Tom Calvert, Senior Supervisor
Professor of School of Interactive Arts and Technology, Simon Fraser University

---

Dr. Vive Kumar, Supervisor
Assistant Professor of School of Interactive Arts and Technology, Simon Fraser University

---

Dr. Belgacem Ben Youssef, SFU Examiner
Assistant Professor, School of Interactive Arts and Technology, Simon Fraser University

Date Approved: May 23, 2006

Abstract

There are various types of conferencing models to transfer multimedia data such as voice and video, but they all have certain limitations when applied to the IEEE 802.11 wireless networks using handheld devices. To address these, we have proposed a conferencing architecture that improves the user scalability.

We simulated the traditional Session Initiation Protocol based on existing conferencing models and our proposed conferencing model under different number of users using the Optimum Network Engineering Tool. We implemented a process node model in the application layer to create our conferencing server model. We conducted a series of simulation scenarios to measure the performance of the conferencing models in terms of user scalability and QoS. We also demonstrated how the mobile users affect the conferencing performance. Our simulation results indicate that our proposed conferencing model provides a more flexible way to expand the conferencing user scale while maintaining the required QoS.

Keywords: IEEE 802.11; Voice over IP; Session Initiation Protocol

Acknowledgments

Thanks to everyone who helped me to achieve this accomplishment. The first to thank are my parents, and my brother, who provided me great support, encouragement and endless love through it all.

I would like to thank Dr. Tom Calvert for being my senior supervisor and providing me with insightful and valuable advice.

My former senior supervisor, Dr. Paris Polydorou was also of great help to me. I would like to thank him for his support and guidance during the past two years, especially for his valuable advice in the time he spent working with me throughout the development of my thesis.

I would also like to thank my supervisor Dr. Vive Kumar, who provided me generous support and valuable comments on my work.

I also want to thank Dr. Belgacem Ben Youssef for being my examiner. I also thank my colleague and friend, Herbert Tsang for his help and encouragement during the past two years.

I also like to acknowledge the OPNET Technologies Inc. for providing our research group with the OPNET educational license.

Without you all, this thesis would not exist!

Contents

List of Tables

List of Figures

List of Abbreviations

3PCC	Third-Party Call Control
ATM	Asynchronous Transfer Mode
BER	Bit Error Ratio
BISDN	broadband ISDN
DHCP	Dynamic Host Configuration Protocol
GPRS	General Packet Radio Service
HTTP	Hyper Text Transport Protocol
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISDN	Integrated Services Digital Network
ITU	International Telecommunication Union
LAN	Local Area Network
MAC	Medium Access Protocol
MCU	Multipoint Control Units
OPNET	Optimum Network Engineering Tool
OSI	Open Systems Interconnection
PC	Personal Computer
PCM	Pulse Code Modulation
PDA	Personal Digital Assistant
PIM-SM	Protocol Independent Multicast Sparse Mode
PSTN	Public Switched Telephone Network
QoS	Quality of Service

RFC	Request for Comments
RP	Rendezvous Point
RSVP	Resource Reservation Protocol
RTCP	Real Time Control Protocol
RTP	Real Time Protocol
SCN	Switched Circuit Network
SIP	Session Initiation Protocol
SMTP	Simple Mail Transfer Protocol
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
VoIP	Voice over Internet Protocol
WLAN	Wide Area Network
WiMax	Worldwide Interoperability for Microwave Access
WWW	World Wide Web

Chapter 1

Introduction

1.1 Introduction

The development of IEEE 802.11 wireless networks [1] enables users to transfer high-speed multimedia data in the form of voice, video and computer games through the wireless medium. Voice over IP, which allows users to talk over the IP networks in a more flexible and less expensive way than the traditional Public Switched Telephone Network (PSTN), is now also used to deliver voice services over the IEEE 802.11 wireless networks.

VoIP has several advantages over the wide area 3G cellular networks. First, it costs less to make a phone call over the IP networks, especially for long distance calls. Second, it enables heterogeneous access methods: from PC to PC, from PC to phone or from phone to phone. Third, it supports more services, such as instant messages, video and conferencing calls.

Handheld mobile devices with small screen sizes are becoming popular. They are much cheaper than the personal computer and easier to carry. Conducting a voice conference with small handheld devices enables real-time collaboration between multiple users and teams participating from different locations. However, there are several challenging issues in the deployment of voice conferencing via handheld devices. It is known that the Quality of Service (QoS) is the main concern for real time multimedia communications [2]. Real time voice communication has very stringent requirements on packet delay and delay jitter. To achieve the QoS requirements, the goal is to minimize the delay, the packet loss and the delay variation (jitter). A delay of 0 to 150 ms is acceptable for telephony communication, but more than 400 ms is not acceptable. Compared with the wired networks, the IEEE 802.11 wireless networks have a higher error rate and a longer end to end delay [2]. The reason is that the wireless network topology is highly dynamic, and the transmission quality of radio is affected by objects in the environment such as buildings, moving objects, and the atmosphere, etc which cause fading, and multipath interference to bring more errors.

In addition to the challenges of the IEEE 802.11 wireless networks, our target users, the small handheld devices, also have their own limitations. One of the major challenges for effective use of the small handheld devices is to reduce their energy consumption thereby extending the battery life. In addition, handheld devices have relatively limited CPU processing power and a small buffer size, which places a constraint on the applications which can run on such devices. The delay variation of packets due to network congestion and the high bit error rate degrades the quality of voice at the handheld device. Multimedia applications need to use a buffer at the handheld device to smooth out the delay variation and improve the quality of voice. However, the buffer size has to be kept small in order to reduce the size, weight and power consumption of handheld devices. On the other hand, if we model the network traffic of the user agent as an M/M/S/1 queue, the maximum queue size will be limited by the buffer size, and the service rate will be limited by the CPU processing power. Based on queuing theory [3], there will be a lower service rate and a smaller buffer size compared with the desktop PC. As a result, there will be a higher probability that the incoming media packets are blocked or dropped for the small handheld device users.

For the reasons described above, a major challenge for researchers and practitioners is to design a conferencing architecture in a way that can optimize its quality of service by taking into account the limitations of both the IEEE 802.11 wireless networks and the small handheld devices.

1.2 The Motivation

When deploying a VoIP conferencing network, we need to consider the following issues due to the above listed limitations of VoIP and wireless handheld devices: performance, quality of service (QoS), reliability, availability, scalability, network traffic overhead and bandwidth. The goal is to find ways to provide equivalent or even better voice quality than the Public Switched Telephone Network (PSTN) at a lower cost with more flexibility.

It is important to design a conferencing architecture that can optimize its performance by taking into account the limitations of both the IEEE 802.11 networks and the small handheld devices. We primarily consider voice-only conferences by weighting their performance without considering the added complexities of video and whiteboard media.

Setting up an Internet voice conference requires conferencing signaling control to establish, modify, and terminate the conference. There are two principal signaling and control protocols that support Internet based voice conferencing: Session Initiation Protocol (SIP) from the IETF [4] and H.323 from the ITU-T [5]. Henning Schulzrinne and Jonathan Rosenberg have compared the first version of H.323 with SIP [6]. After H.323 version 4 was released, I. Dalgic and H. Fang compared H.323 and SIP in terms of functionality, Quality of Service (QoS), scalability, flexibility, interoperability, security, and ease of implementation [7]. Another comprehensive comparison of H.323 version 4 and SIP can also be found in [8]. In Katrinis' paper [9], SIP and H.323 were also evaluated and compared in terms of scalability and heterogeneity in multimedia conferencing environments. Since SIP is simple and is considered to be the future protocol of choice for conferencing signalling, it has been extended to support instant messages and event notification [10, 11]. Therefore, we focus our study on SIP as the conferencing signalling and control protocol.

In SIP, there are various types of conferencing models [12, 13], but they all have certain limitations when applied to the IEEE 802.11 wireless networks and used on small handheld devices. These models are incapable of providing guaranteed QoS. Therefore, we propose a conferencing architecture which improves the user scalability and conserves battery energy on the handheld devices by using a centralized conferencing server to mix the media streams, thus reducing the end user data processing requirements. Furthermore, we utilize a voice codec which supports silent suppression in order to increase the bandwidth efficiency. Silent suppression means that the user agent will not send any voice data when the user is not talking.

We simulate our proposed conferencing model as well as the traditional conferencing models under different conferencing user scales using the Optimum Network Engineering Tool (OPNET) simulation software [14]. The simulation results are analyzed and compared to determine whether the proposed conferencing model has a shorter delay for large scale conferences. We then generalize our simulation results to find which SIP based VoIP conferencing architecture performs best over the IEEE 802.11 wireless networks using small handheld devices.

1.3 Thesis Organization

The thesis is organized as follows. Chapter 2 provides the basic background knowledge of IEEE 802.11 wireless networks, Voice over IP technology and analyzes our target end users, small handheld devices. After that, we give a brief introduction to the Session Initial Protocol (SIP). This introduces the typical SIP request and response messages, and also gives examples of simple SIP call set up and tear down, as well as SIP call set up using proxy and redirect servers. Furthermore, we summarize the typical conferencing models and analyze their limitations using small handheld devices over the IEEE 802.11 wireless networks. After that, we classify the SIP mobility categories and analyze precall mobility as well as midcall mobility. In the end, we briefly compare SIP with another popular signaling protocol, H.323.

Chapter 3 first introduces the Optimum Network Engineering Tool (OPNET) simulation tool, which provides the network simulation environment for network modeling and design. Then, we present our proposed conferencing model and the conferencing entities, including User Agent (UA), Conferencing Server (CSer) and SIP Server (SSer). Each of the entities is analyzed and modeled in the OPNET simulation tool. The simulation implementation of process models that are used in our simulation will also be discussed.

In Chapter 4, we conduct simulation experiments on three network scenarios to analyze and compare the performance in various QoS aspects of our conferencing model with other traditional conferencing models, such as end system mixing, unicast receive and unicast send, unicast receive and multicast send. We use the OPNET simulation tool to simulate these network scenarios. In order to illustrate the bandwidth efficiency of using silence suppression voice codec, we selected a G.729 voice codec in the first scenario and G.729 (silence) voice codec in scenario two. In the first scenario, we compare the voice conference QoS performance under a small number of users for all the conferencing models while in scenario two, we increased the number of users for the UIMO conferencing model as well as our proposed conferencing model using G.729 (silence) voice codec to evaluate their performance. After that, we compared the performance of static users with mobile users using our proposed conferencing model in scenario three. We kept track of the node statistics as well as the global wireless statistics such as the end to end delay, jitter, packet loss, and the amount of traffic sent or received by the UA and the CSer in different simulation network topologies. We draw conclusions from the simulation results and address the possible future research in Chapter 5.

Chapter 2

Background

2.1 IEEE 802.11 Wireless Networks

The IEEE 802.11 is a family of wireless network standards developed by the IEEE 802.11 working group [1]. The IEEE 802.11a/b/g are currently the most widely used wireless networking standards, which provide the maximum transmission data rates of 54Mbps, 11Mbps and 54Mbps respectively. They support both Ad Hoc and infrastructure network architectures. In Ad Hoc networks, any two stations can establish a direct communication link between them, while in infrastructure networks communication between stations is done through a central Access Point (AP). In our simulations, the infrastructure network architecture will be considered.

In the IEEE 802.11 wireless networks, the physical layer is responsible for transmitting packets over the radio medium. In contrast to wireline transmission links where the bandwidth can be easily increased and the channel quality can be guaranteed, the wireless channel bandwidth is limited because of spectrum allocation and physical limitations. Data packet transmission over the radio medium suffers higher error rates and longer delays than wireline transmission link, especially in a highly dynamic network topology where mobile users change their locations and access points frequently [2]. Moreover, the transmission tends to be affected by the buildings and the atmosphere, which cause fading, and multipath interference. The 802.11b and

802.11g standards transfer data over the unlicensed 2.4 GHz band while the 802.11a standard uses the 5 GHz band. Since they operate in an unlicensed frequency band, 802.11b and 802.11g devices are often affected by interference from home appliances which also use the same 2.4 GHz band, such as microwave ovens and cordless phones.

Three physical layer implementations are specified for the IEEE 802.11 networks: Frequency Hopping Spread Spectrum (FHSS), Direct Sequence Spread Spectrum (DSSS), and Orthogonal Frequency Division Multiplexing (OFDM). The 802.11g specification employs Orthogonal Frequency Division Multiplexing (OFDM), the modulation encoding scheme used in 802.11a, to obtain higher data speed while the 802.11b uses the Direct Sequence Spread Spectrum (DSSS) modulation scheme to encode data. DSSS will be used for our simulations.

According to Janevski [15], there simultaneously exist different traffic types in a wireless IP network, such as voice, video, multimedia, and data. Applications can be classified into real time and non real time, such as voice service and email service. Different traffic types have different characteristics and thus require different quality of service (QoS) demands.

All these factors in wireless networks give rise to challenging issues to transfer real time data over the wireless network, such as effective bandwidth allocation, high channel error bit rate, and longer delay.

2.2 VoIP

Voice over Internet Protocol (VoIP) refers to IP Telephony, which delivers voice information in digital form packets over the Internet rather than the traditional circuit-switched network as in the public switched telephone network (PSTN) for example. It allows users to talk in a more flexible and less expensive way. With the rapid development of IEEE 802.11 wireless networks, there is a trend to migrate VoIP into the cellular wireless networks.

VoIP provides a range of services which are difficult or prohibitively expensive for PSTN, such as inexpensive voice mail, instant messages, as well as video and conferencing calls, all of which are easy to access from the Internet. Compared with PSTN and the 3G cellular networks, it costs less to make a phone call over the IP networks, especially for long distance calls. In addition, it is more flexible. It enables heterogeneous access methods: from PC to PC, from PC to phone or from phone to phone.

However, real time voice communications have very stringent requirements on packet delay and delay jitter. When people are interactive in real time, a delay or jitter more than a few hundred milliseconds causes a significant impact on the quality of communication. According to ITU-T G.114 [16], a delay of 0 to 150 ms is acceptable for telephony communications, between 150 ms and 400 ms can also be acceptable, but more than 400 ms is not acceptable. Furthermore, packet losses are not desirable, although limited losses can easily go unnoticed by using error concealment techniques [17].

As mentioned in [18], when deploying a VoIP conferencing network, we still need to consider the following issues due to the above listed characteristics of telephone: performance, quality of service (QoS), reliability, availability, scalability, network traffic overhead and bandwidth. The goal is to find ways to provide the equivalent or even better voice quality as the PSTN at a lower cost with more flexibility.

There are also studies that focus on integrating the existing PSTN networks with the wide area cellular networks and the VoIP networks to provide seamless and value added services for the users. Our research here will only focus on the deployment of VoIP over the IEEE 802.11 wireless networks using small handheld devices.

As seen in Figure 2.1, there are two main aspects in VoIP: the call signaling and controlling information, as well as the media information. Setting up an Internet voice call requires call signaling control to establish, modify, and terminate the conference. There are two principal signaling protocols: Session Initiation Protocol (SIP) from IETF [4] and H.323 from ITU-T [5]. Both signaling protocols support TCP and UDP transport protocols. Since SIP is considered to be the future trend for conferencing signalling protocol, it has been extended to support instant messages and event notifications [10, 11]. Therefore, we will focus on SIP and study it in more detail in Section 2.4.

A VoIP call can support heterogeneous physical terminal access from a traditional wired network PC, a stand alone VoIP phone to a mobile handheld device. The Internet protocol (IP) layer operates above the network interface, and transfers data to the transport layer. The user datagram protocol (UDP) was designed as the interface for the Internet layer and application layer. However, it is inadequate to transfer time sensitive packets. Hence, a companion transport layer protocol operates above UDP to provide specific support for the real time multimedia data. This protocol suite actually consists of two protocols: The real time transport protocol (RTP) [19] and the real time transport control protocol RTCP. The RTP protocol makes it possible to encode and split the data into packets and transfer such data packets over the Internet. Furthermore, it also provides packet sequence information that are not available in UDP, so endpoints can determine arrival order to allow reduction in jitter.

As defined in the Internet Draft with title "The Internet Multimedia Conferencing Architecture" [20], the term "conferencing" is often used in two different ways. First, it refers to bulletin boards and mail list style asynchronous exchanges of messages between multiple users. Second, it means synchronous or real-time conferencing, including audio/video communication and shared tools such as whiteboards and other applications. Our research is about the architecture for the latter application, multimedia conferencing in an Internet environment. We primarily consider voice-only conferences by looking into their performance without considering video and whiteboard media.

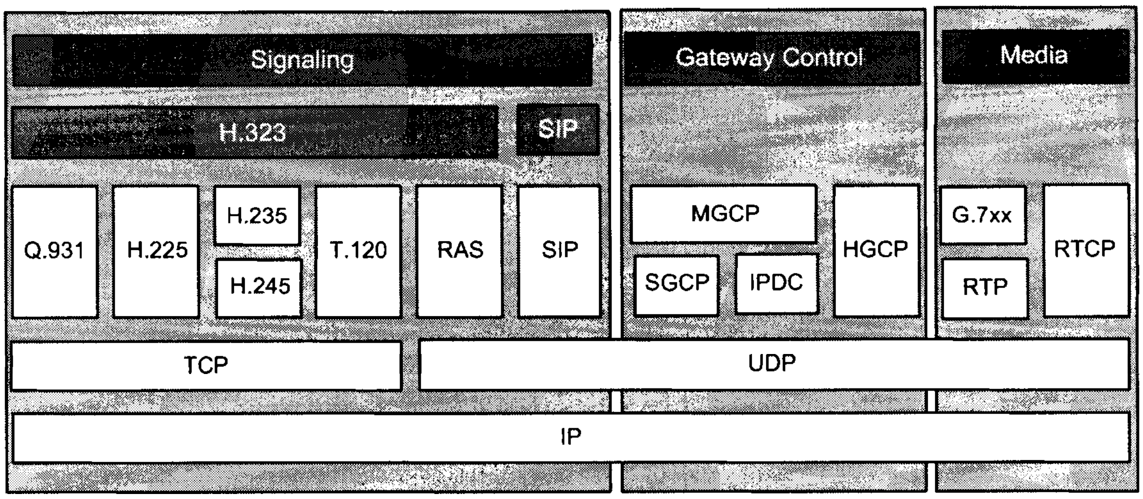


Figure 2.1: VoIP Protocol Stack

2.3 Handheld Devices

Small handheld devices, such as PDAs, Palms and Pocket PCs, are quickly becoming common tools for work, study and play due to their compact size and practical functions. They can be carried by users inside buildings where they are connected to the IEEE 802.11 wireless networks or maybe anywhere connected to the cellular networks, such as GPRS. They have two significant advantages over traditional devices. First, they are much cheaper than the personal computer. There are products that integrate the PDA functionalities with the cell phone available in the market and are typically very small to carry. Second, they have adequate processing power to handle light video and audio communications. On the other hand, they generally have small screen sizes, limited battery life and small secondary storage. Energy conservation is the major concern for handheld devices.

Regardless of their limitations, there are more and more applications being developed for the small handheld devices to meet a variety of needs in education, communications and entertainment since they are inexpensive, compact, and lightweight. Conducting a voice conference with small handheld devices enables real time collaboration between multiple users and teams participating in different locations. Our research investigates their VoIP conferencing performance by analyzing the QoS performance of several typical conferencing models under different user levels in the simulation environments.

When making a voice call, the analog voice message is converted into digital form and then coded using the voice codec before being transmitted to the radio channel. At the receiver, all the incoming data packets are decoded and converted to analog form. There are various types of voice codecs available for the small handheld device, such as G.711, G.723.1, G.726, and G.728 defined by the ITU-T [21, 22, 23, 24]. Table 2.3 gives some of the characteristics of several standard codecs.

Table 2.1: Characteristics of Several Standard Codecs

Codec	**Algorithm**	**Frame Size/Lookahead**	**Bitrate**
G.711	PCM	0.125 ms/0	64 Kbps
G.723.1	MPC-MLQ	30 ms/7.5 ms	6.3 Kbps
G.726	ADPCM	0.125 ms/0	32 Kbps
G.728	LD-CELP	0.625 ms/0	16 Kbps
G.729	CS-ACELP	10 ms/0	8 Kbps

The voice codecs provide a range of maximum transmission data rates, such as 64Kbps, 6.3Kbps, 32Kbps, 16Kbps and 8Kbps. And most provide an option to support silent suppression. Codecs with lower transmission rates conserve bandwidth and improve the user capacity. On the other hand, these codecs introduce longer delays since they require a longer time to process the packet compression and decompression. Therefore, there is a trade-off between the codec selection and the packet delay. Given a specific codec, the QoS performance depends mostly on the network topology and conferencing architecture.

Packet loss causes voice clipping and skips and long end-to-end delay which have a significant impact on the quality of communication. A buffer in the receiving end always compensates for jitter. If the jitter exceeds the size of the device buffer, there will be buffer overflow and as a result, a packet that arrives later than the length of the jitter buffer will be lost in the transmission path. Based on queuing theory [3], when the end user does not have enough buffer room and CPU processing power to store and process the incoming packet streams, there is a higher probability that the incoming packets are blocked or dropped. Our target users, small handheld devices, tend to have limitations on their CPU processing speed and small buffer size as well as limited battery life, so our goal is to reduce the packet blocking probability for the end users by reducing the data processing requirements and thus improving the voice quality during the conference and conserving battery life.

2.4 Overview of SIP

Session Initiation Protocol (SIP) is a text based application layer signaling protocol which has been developed and designed by the Internet Engineering Task Force (IETF). It can be used as the signaling control protocol for Internet phone calls. The original protocol specifications can be found in RFC 3261 [4].

A SIP session can be an Internet telephone call, a multimedia conference or a distributed computer game. It defines different message types and response codes to represent different requests and responses messages as shown in Table 2.4 and Table 2.4. SIP messages are exchanged when a call is set up, modified, and terminated, and the response message are possible responses to the requests.

2.4.1 SIP Call Flow

A SIP based call requires several logical entities, including the SIP enabled User Agent, Proxy Server, Redirect Server, and Registrar. Each entity will be discussed in more details in Chapter 3. A conferencing call can be set up by dialing to the Conference Server a SIP URI or an email address, such as sip: math@209.87.56.200 and sip: conf@surrey.sfu.ca.

Table 2.2: SIP Request Messages

Request	**Purpose**
INVITE	To initiate a dialog between two participants
BYE	To terminate a connection between two users in a session
OPTIONS	To request information about capability
ACK	For reliable message exchange following a successful INVITE
REGISTER	To convey location information to a SIP server
CANCEL	To cancel an initiated session not yet finalized
REFER	To transfer calls and To contact any external resource
NOTIFY	To provide information about a state change that is not related to a specific session
SUBSCRIBE	To indicate a desire for NOTIFY requests

SIP has two types of URIs. The first one is known as the address of record (AOR) and corresponds to a user [25]. An AOR is frequently thought of as the public address" of the user. When initiating a SIP call, the end user populates the address-of-record (AOR) of its target in its request message header field that requires database lookups. The second one corresponds to a single device or end point [26]. A device URI is known as a contact, and typically does not require database lookups. The difference between an address-of-record and a contact address is similar to the difference between a user and a device.

Several simple SIP based call examples will be introduced using call flow diagrams between a calling and called party in the following sections. The calling and called parties could be SIP phones, handheld devices or SIP enabled cell phones. The examples are shown as defined by RFC 3261 [4] and RFC3665 [27]

A Simple Session Establishment Example

A simple session establishment example is shown in Figure 2.2 to demonstrate the basic message exchange between the calling and called party to set up and tear down

Table 2.3: SIP Response Messages

Response	Purpose
1xx	Provisional (Request received, continuing to process the request)
2xx	Success (Action successfully received, understood, and accepted)
3xx	Redirection (Further action must be taken in order to complete the request)
4xx	Client Error (Request contains bad syntax or cannot be fulfilled at this server)
5xx	Server Error (Server failed to fulfill an apparently valid request)
6xx	Global Failure (Request unable to be fulfilled at any server)

a call session. The calling party, A, initiates the call by sending a SIP INVITE request message to the called party, B. The media session will not begin until the calling party receives the 200 OK response from the called party, and sends the ACK message to confirm the successful connection. Any of the call parties can tear down the call simply by sending a BYE request messages to the other one.

SIP Call Example with Redirect Server

In the example of Figure 2.3, A knows exactly the IP address of B, thus he is able to send the INVITE request directly to B. However, it is not the case in general. One reason is that, the IP address is often dynamically assigned due to the shortage of available IP addresses. Moreover, the IP address is not always unique for a SIP user. For example, you have one IP address at school and another IP address at home. Since the calling party does not know the IP address of the called party, a server is required to route the INVITE request. There are two types of servers. The first one is a redirect proxy, which receives a request and then looks up the intended recipient of the request in the location database, then creates a list of current locations of the called party and sends it to the calling party in a response within 3xx class Response. The other one is a proxy server, which is in the middle of a SIP message exchange, receiving messages and forwarding them as shown in Figure 2.5.

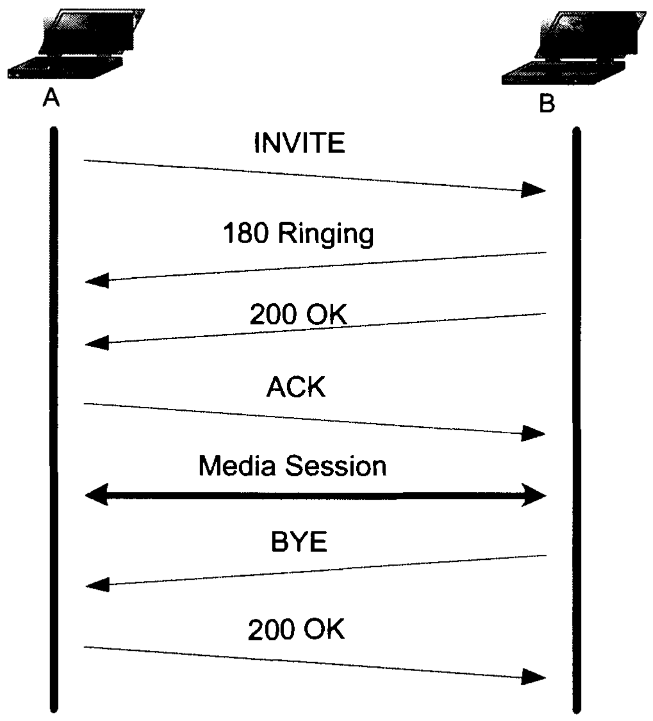


Figure 2.2: A Simple Session Establishment Example

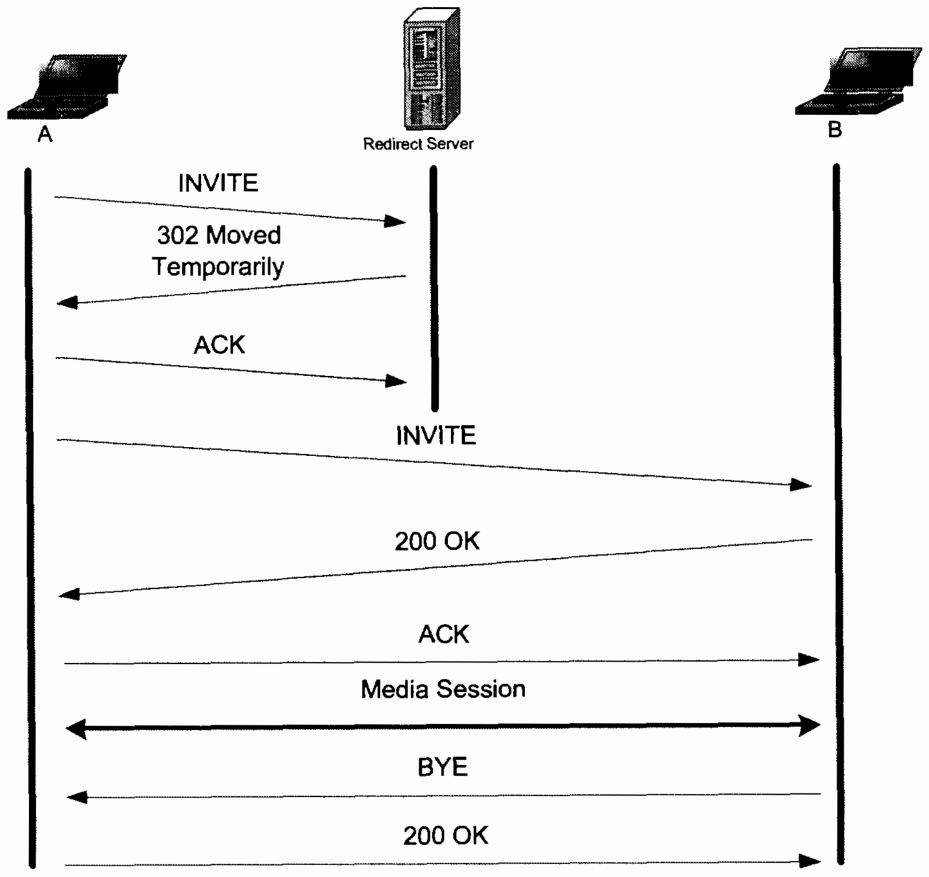


Figure 2.3: SIP Call Example with Redirect Server

SIP Call Example with Proxy Server

The proxy server is a very important entity in the SIP infrastructure. It operates in a similar way to a proxy in HTTP and other Internet Protocols. There are two basic types of SIP proxy server: stateful server and stateless server. Stateless proxy is simpler and faster, but it does not response with call transaction state information while stateful proxy maintains states for the call transaction and provides more advanced features, such as forking or recursive traversal [26]. Figure 2.4 illustrates the difference between the stateful and the stateless proxy. The stateful proxy returns a 100 Trying Response message to the INVITE while the stateless does not.

Figure 2.5 shows a typical SIP call example with one proxy server. There can be multiple proxy servers in the signaling path. In the example, the calling party, A, calls B through a SIP proxy server. The SIP proxy server does not set up or terminate sessions, instead, it performs the SIP message exchange, receiving messages and forwarding them. The SIP proxy server performs a look up in the database to locate B's IP address after it receives the INVITE request message from A and obtains parameters from the Header, then it forwards the request to B to set up the call.

2.4.2 SIP Conferencing Models

Advances in Internet and wireless network technologies have stimulated the integration of multimedia communications over wireless networks, leading to the development of various high performance computer assisted collaboration applications. Large scale multimedia conferencing for supporting collaborative communication between geographically separated groups of users is an example of a more efficient and low cost collaboration application.

SIP supports different types of conferencing models. Below, we first summarize the general types of media distribution models in multimedia conferencing as described in the Internet Draft with title "Models for Multi Party Conferencing in SIP" [12] as well as in Singh and Nair's paper [13]. Then, we explore those model limitations

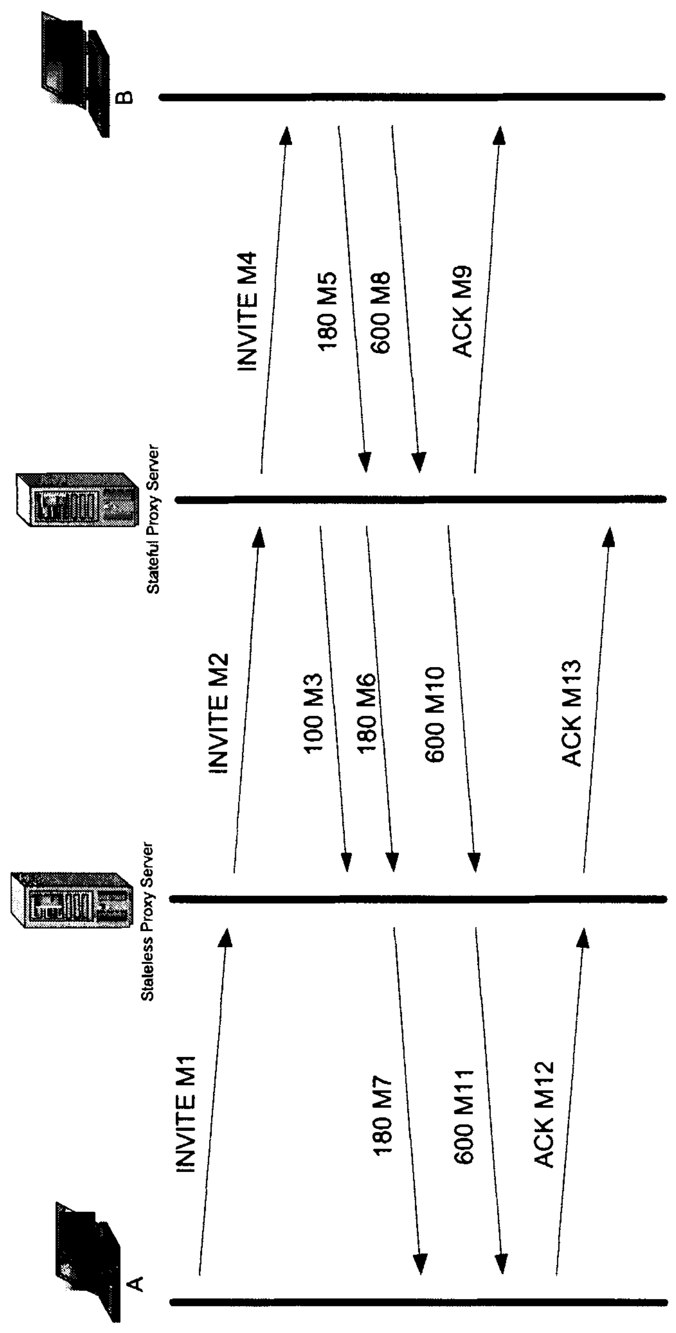


Figure 2.4: SIP Call Example with Stateless and Stateful Proxy Server

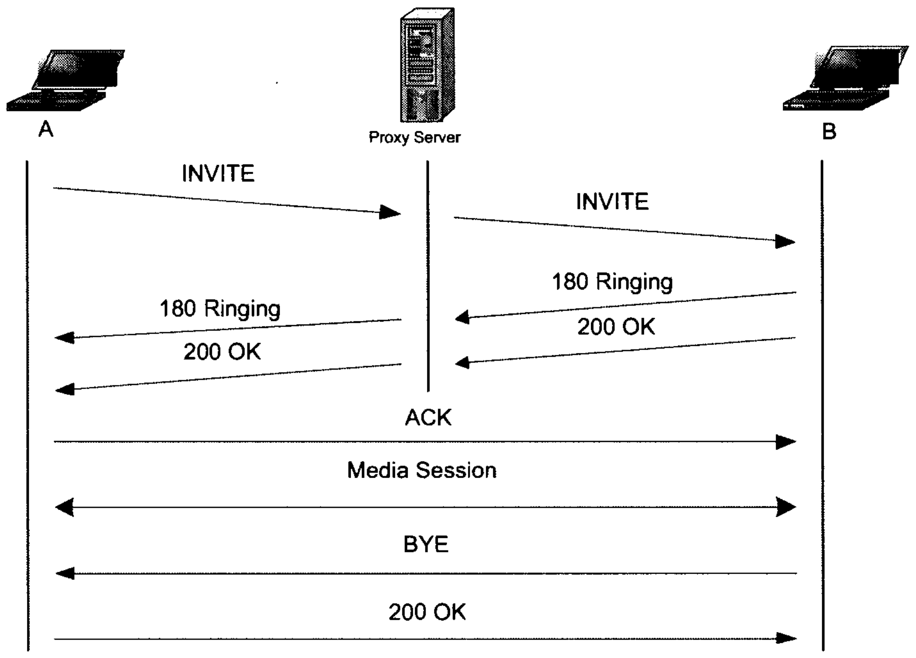


Figure 2.5: SIP Call Example with Proxy Server

when applied to the IEEE 802.11 wireless networks using small handheld devices.

End System Mixing

In the end system mixing model, the user agent functions as both the server and the client. In other words, the user agent handles both the signaling and media mixing. Figure 2.6 illustrates an example of end point mixing. If A and B are in a call, A can invite C to join their conversation by initializing a completely separate SIP call with C. A sends the mixed media stream of A and B to C, and sends the sum of A and C to B. B and C do not need to be aware of the service provided by A, and they can mix other participants's media streams independently.

This model is very flexible and inexpensive for small size Ad Hoc network conferencing, since it does not require any additional server. However, the small handheld device may be incapable of handling both the signaling and the media mixing under

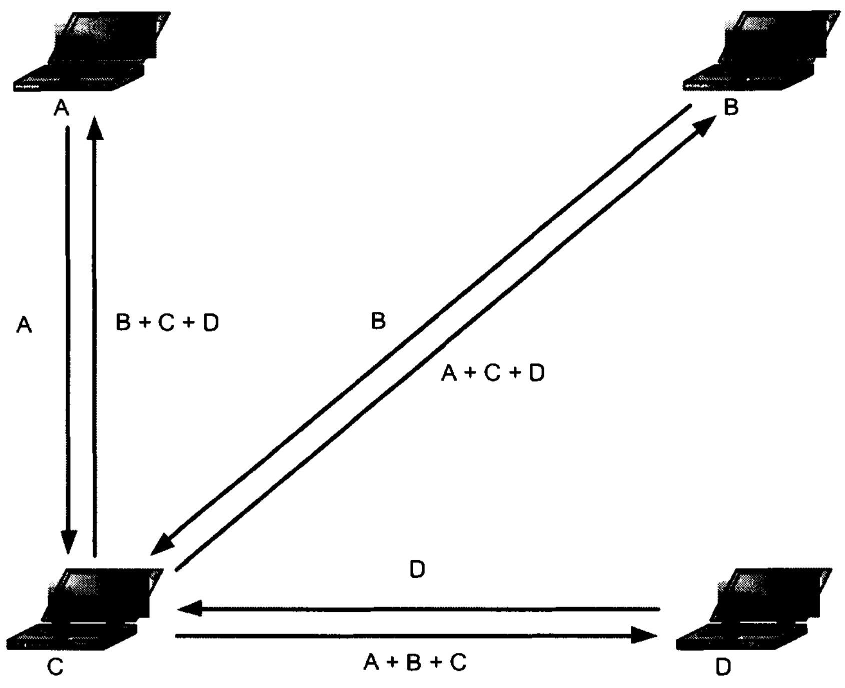


Figure 2.6: End System Mixing

heavy traffic load. It thus, limits the user capacity and decreases the performance dramatically as the number of users increases.

Unicast Receive and Unicast Send

In the unicast receive and unicast send model, the central conference server receives and mixes all the incoming media streams. The server is also responsible for sending the appropriate mixed media stream to the destined user agents. The main advantage of this model is that the central conference server reduces the user data processing requirements to some extent. However, the conference server sends the mixed media streams via unicast, which unnecessarily duplicates media streams and occupies more bandwidth. As a result, it causes more traffic on the radio channels and introduces more interference and time delay. Furthermore, the user capacity is limited by the processing power of the central conferencing server.

The model is depicted in Figure 2.7. The conferencing server sends a copy of the mixed media streams via unicast to each participant.

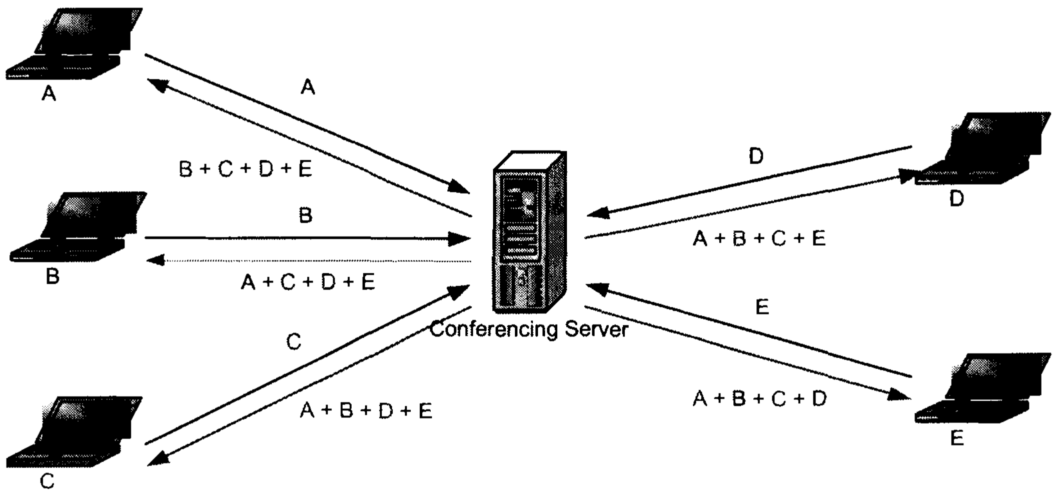


Figure 2.7: Unicast Receive and Unicast Send

Unicast Receive and Multicast Send

In unicast routing, a packet is routed to a single destination. In multicast routing, a single packet is routed to a group of destinations. Multicast Internet addresses are reserved in the range 224. 0. 0. 0 to 239. 255. 255. 255. Although SIP supports both TCP and UDP transport protocols, the multicast transport protocol is always UDP since in real time multimedia communication, the handshake and acknowledgments of TCP are not possible and, in fact, unnecessary. Multicasting is becoming a part of the public Internet as service providers begin supporting it, thus enabling the use of multicast conferencing model.

In the unicast receive and multicast send model, the central conference server is mainly responsible for mixing the incoming media streams and transmitting the mixed media steams to the destined user agents via multicast, when multicast is available. Thus, unlike pure multicast, user agents do not need to filter or mix the incoming media streams. Every participant receives the mixed stream which includes his own stream.

Unless a sender maintains a copy of its original data, it will have difficulty removing its own stream from the mixed stream. This model reduces the bandwidth requirements by utilizing central IP multicast. The only problem is that the user capacity is limited by the central conferencing server as for the unicast receive and unicast send model.

As we can see from Figure 2.8, the conferencing server combines all the media streams and sends the mixed streams via multicast.

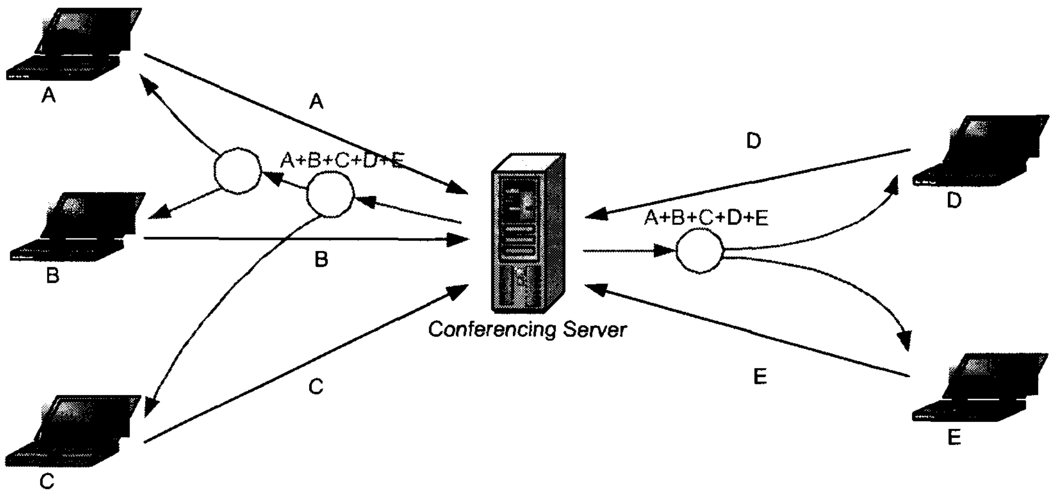


Figure 2.8: Unicast Receive and Multicast Send

Full Mesh

In the Full Mesh model, each user agent sends a copy of its media streams to all the participant users via unicast. This model minimizes the transmission time delay since there is no central conference server for mixing the media streams. The drawback is that without a central conferencing server, the user agent needs to sum all the incoming media streams. In other words, there are more data processing requirements for the user agent and as a result the blocking probability increases and the end user consumes the battery more quickly especially when the number of participants increases. Furthermore, unnecessary copies of media streams are transmitted, which increases the bandwidth requirements as well as introduces additional interference over the radio channel.

Figure 2.9 shows an example of a full mesh conferencing with four participants. Generally, there is only one active speaker during the conference at a time, so the end user can handle a small scale conference like this as long as silence suppression is utilized.

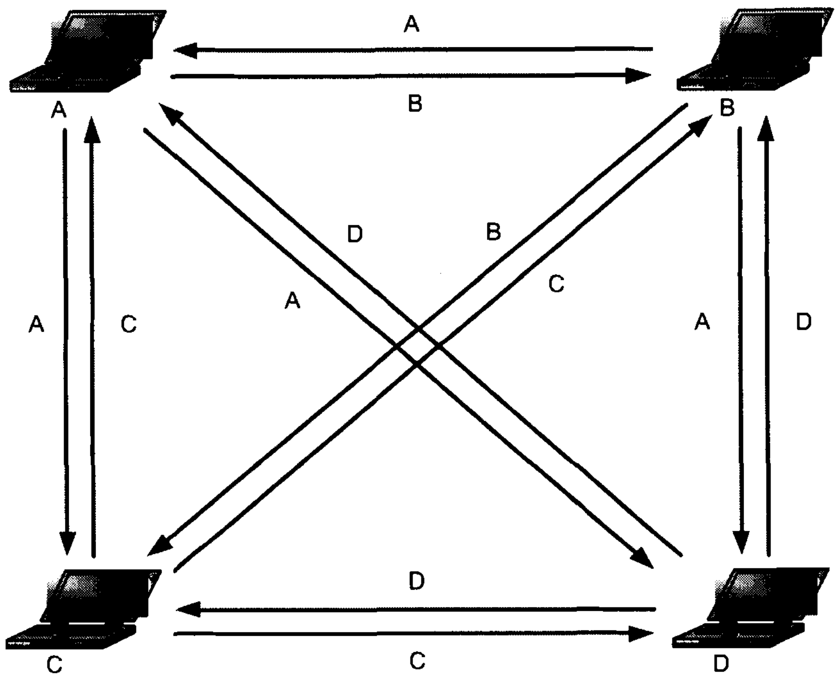


Figure 2.9: Full Mesh

As shown from the above discussion, these four types of conferencing models are unable to satisfy the QoS requirements when being applied to VoIP conferencing over the IEEE 802.11 networks using small handheld devices. Therefore, we propose a conferencing architecture which avoids unnecessary transmissions of duplicated media streams through the radio channel and reduces the data processing requirements for the user agents.

2.4.3 SIP Mobility

It is important to account for mobility in IP networks in order to provide better service for mobile devices such as laptop and small handheld devices. Mobility can refer to one of the following types:

- Personal Mobility : Personal mobility allows a user to have a constant identifier across a number of terminal devices, such as the work station in an office and the personal computer at home. The user can use the identifier in different terminals at the some time or alternately. SIP can support personal mobility easily since it uses SIP URI such as email address to identify the user. Users are only required to send a REGISTER request message to update their IP addresses and point of connection to the Internet.
- Service Mobility : Service mobility allows a user to keep the same service when mobile or change terminal device.
- Session Mobility : Session mobility allows a user to maintain a media session even while changing terminals. In the SIP case, there are two ways to support session mobility, one is third-party call control (3PCC) and the other is the REFER mechanism [28] [29].
- Terminal Mobility : Terminal mobility allows a device to move between IP subnets, while continuing to be reachable for incoming requests and maintaining the ongoing sessions regardless of the location change.

SIP can support terminal mobility with minimal extensions. The IETF has proposed an approach in their Internet Draft, Supporting Mobility for TCP with SIP[30], which uses SIP to provide a means of terminal mobility for TCP applications. Examples of terminal mobility are given in Figure 2.10 and Figure 2.11.

As shown in Figure 2.10, when A temporarily moves to another SIP network before it sets up a call, he obtains a temporary IP address that is automatically assigned through DHCP. He is then required to perform a double registration. The first DHCP

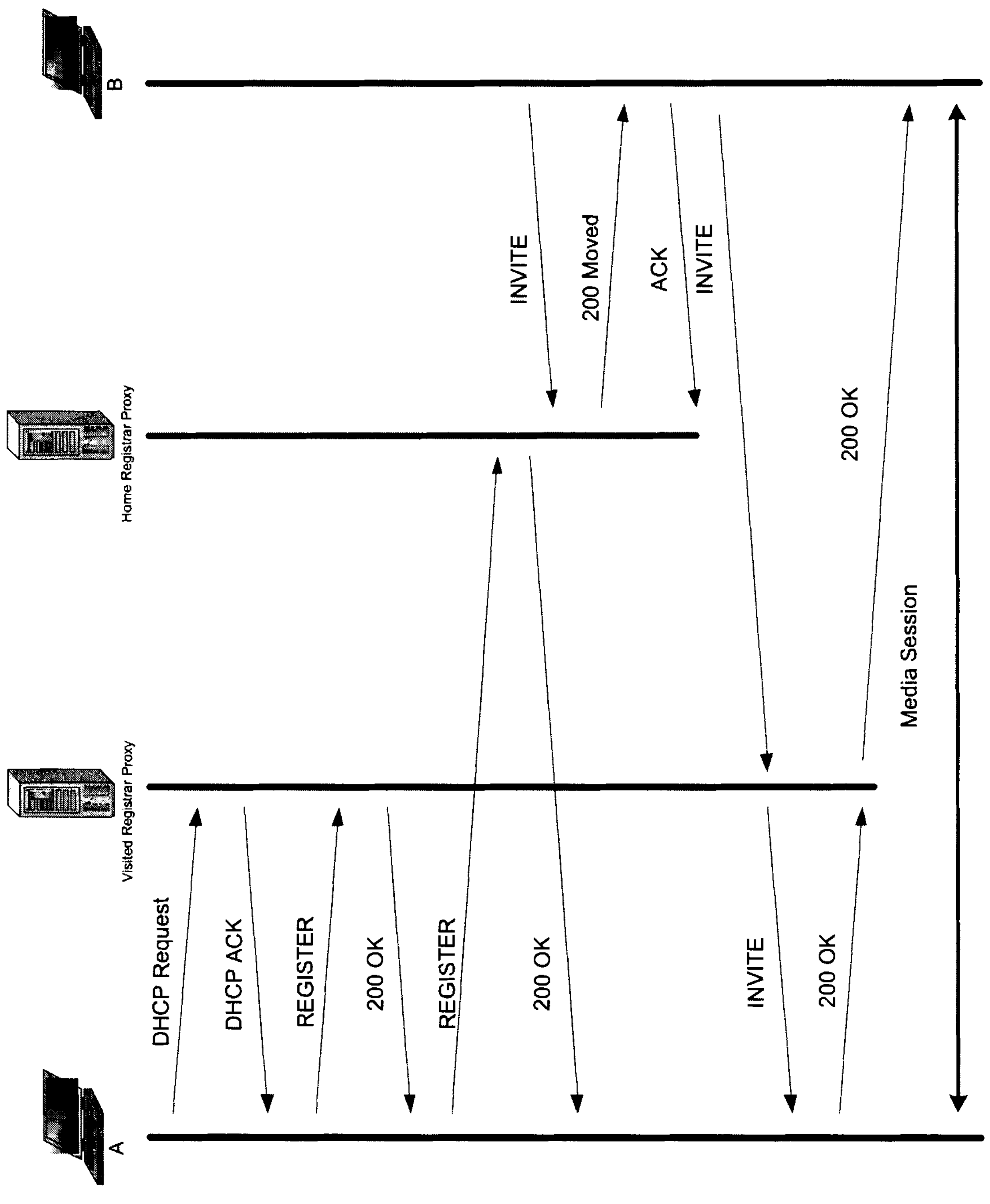


Figure 2.10: Precall Mobility

Request is sent to the Visited Registrar Proxy. After he receives the DHCP ACK acknowledgement from the new service provider, he sends the second REGISTER request to update his location information in his home Registrar Proxy. B receives A's temporary IP address from A's Registrar Proxy after he sends an INVITE request to it. Then, B is able to set up a call with A through A's new service provider.

In Figure 2.11, A moves to another SIP network during the call. He is temporarily assigned a new IP address. He quickly performs an INVITE request to B in order to forward the media flows to his new location. A few RTP packets may be lost during this short period, which leads to a slight interruption to the call. If A can receive the media streams from both service providers, then, it will not effect the call quality.

2.5 H.323 vs. SIP

The ITU-T H.323 is a series of recommendations that defines protocols and procedures for multimedia communications on the Internet [5]. It is part of the H.32x series protocols that describe multimedia communication over ISDN, broadband ATM, PSTN and IP networks. These protocols are listed in Table 2.5.

Table 2.4: ITU H. 32x Family of Standards

Protocol	**Title**
H.320	Narrow-band visual telephone systems and terminal equipment
H.321	Adaptation of H.320 visual telephone terminals to B-ISDN environments
H.322	Visual telephone systems and terminal equipment for local area networks which provide a guaranteed quality of service
H.323	Packet-based multimedia communications systems
H.324	Terminal for low bit-rate multimedia communication

The first version of H.323 was originally created in 1996 to provide a mechanism for

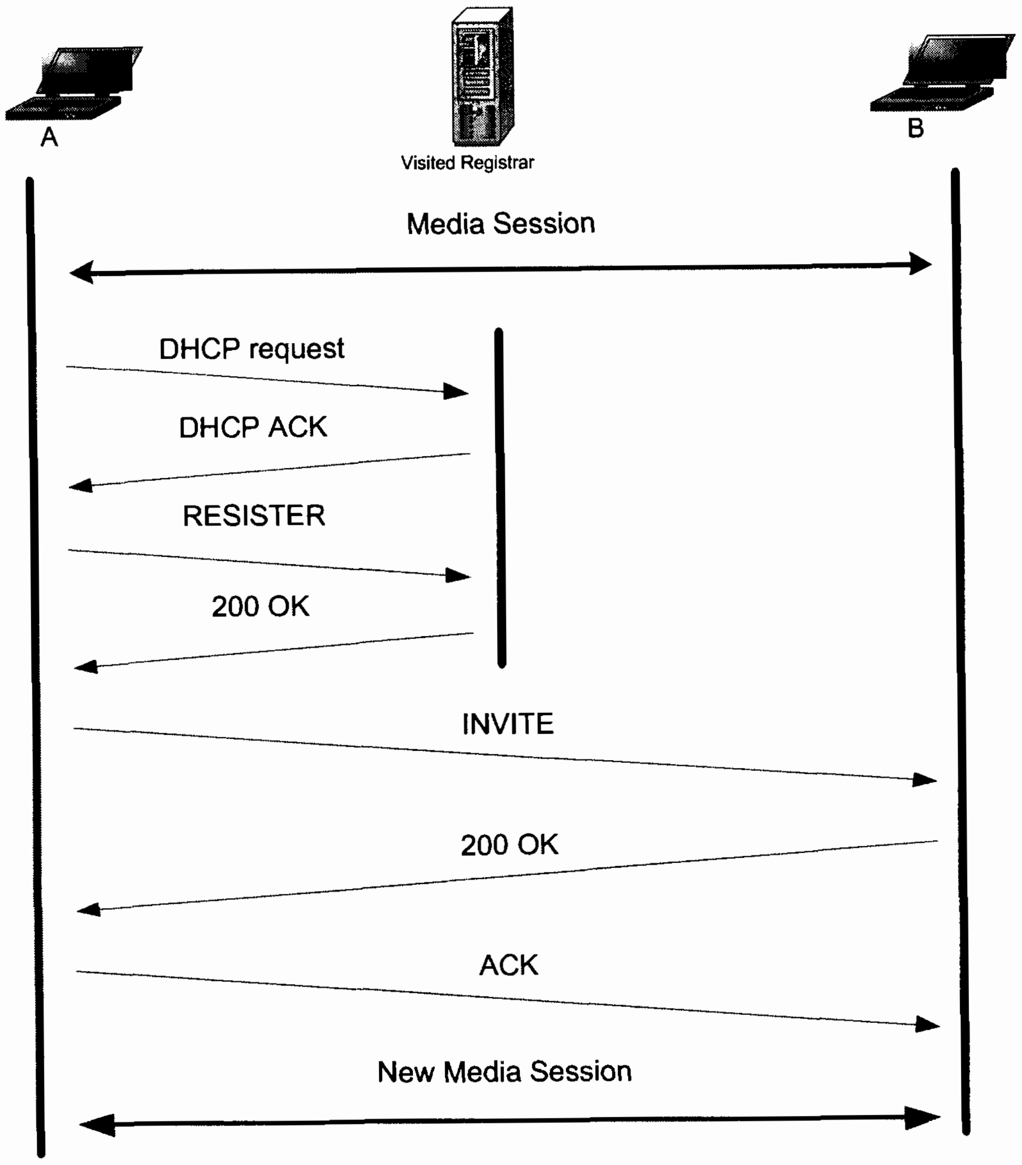


Figure 2.11: Midcall Mobility

transporting multimedia applications over LANs. Most of the existing H.323 implementations are based on the second version of the standard, which was decided in February 1998 to fix some of the problems and limitations in version 1. The latest version, version 5 released in July 2003, but the protocol is still under revision at the IUT-T to address the growing needs of VoIP networks.

H.323 consists of a set of protocols that are responsible for encoding, decoding, and packetizing audio and video signals, for call signaling and control, as well as for capability exchange. It references a number of other ITU-T protocols as shown in Table 2.5.

Table 2.5: Protocols Referenced by H.323

Protocol	**Description**
H.225	Call signalling protocols and media stream packetization for multimedia communication systems
H.235	Security and encryption for H-series (H.323 and other H.245-based) packet-based multimedia terminals
H.239	Role management and additional media channels for H.300-series terminals
H.245	Control protocol for multimedia communication
H.332	For large size conferencing
H.450	Supplementary Services
H.26x	Video Codecs
G.7xx	Audio Codecs
T.120	Multipoint graphic communication

The protocol defines the following main elements for a network-based communication system: the terminal, the gatekeeper, the gateway, and the multipoint control unit (MCU). A typical communication system example is shown in Figure 2.12. All H.323 terminals must support basic G.711 PCM audio codec, and their support of video and data are optional. A gatekeeper is an entity on the network that provides network services for H.323 terminals, gateways, and MCUs. Its services include address translation, network access control, and bandwidth management etc. If a gatekeeper is present, all terminals within that zone must register with it, and the gatekeeper gives permission to make or accept a H.323 call based on a variety of factors. However, a gatekeeper is not a required element for the H.323 networks. A gateway provides the connection path between the packet-switched (IP) networks and another protocol networks, such as PSTN and ISDN. The gateway is not required when there is no interconnection with other protocol networks. An MCU provides conferencing services for terminals. It includes both multipoint controllers, which manage the H.323 terminal functions and capabilities in a multipoint conference, and multipoint processors, which process the audio, video, and data streams between H.323 terminals.

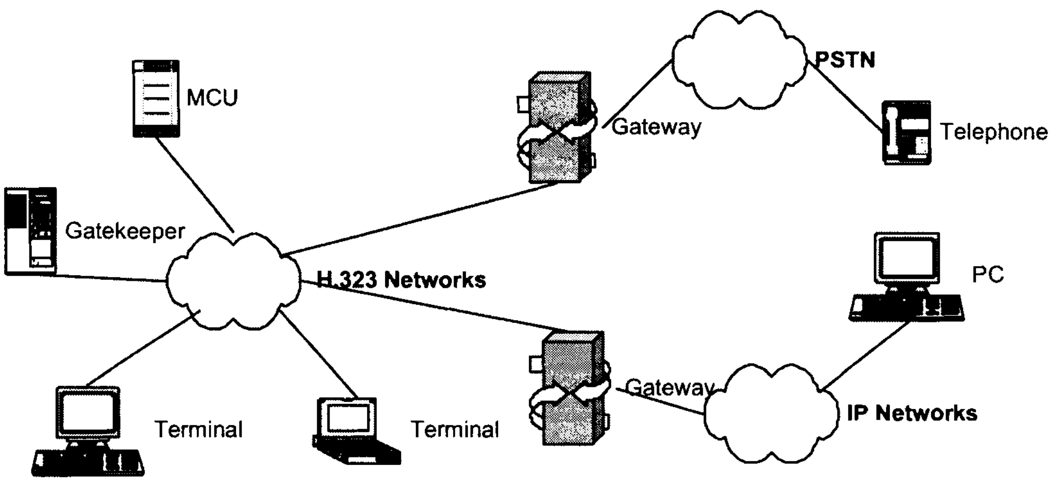


Figure 2.12: Elements of an H.323 Networks

SIP and H.323 were developed for different purposes by standards bodies with very different requirements. H.323 was developed by the ITU. So, it inherits the complexity of other ITU-T protocols, such as PSTN, and it utilizes binary encoding and reuses parts of ISDN signaling. SIP, on the other hand, was developed by the IETF with an Internet perspective, designed to be scalable over the Internet and work in an interdomain way utilizing the full set of Internet utilities and functions. Based on the previous work that compared different versions of H.323 with SIP in various aspects [6], [7], [8] and [9], we only briefly compare these two protocols in terms of complexity, extensibility, scalability, services and wireless standards consideration.

The first key difference is the encoding scheme used by the two protocols. H.323 uses a binary representation for its messages while SIP encodes its messages into text, similar to HTTP and SMTP. The binary encoding of H.323 may result in smaller message sizes but it adds complexity to implementations. The text based messages are simpler and easier to debug and analyze, which makes it more friendly to Internet and Web developers who have used it to develop various other Internet applications. Furthermore, H.323 is a rather complex protocol. It consists of a set of protocols that are responsible for encoding, decoding, and packetizing audio and video signals, for call signaling and control, as well as for capability exchange, while SIP only defines 37 headers, each with a small number of values and parameters. SIP only requires a single request that contains all necessary information, while most of the H.323 services requires interaction between the several protocol components that are included in their standards.

Extensibility is another key metric for measuring an Internet phone signaling protocol. The extensibility mechanisms ensures continuous support of existing as well as new services and features over time. H.323 provides a full backward compatibility with its old versions, however, vendors may chose to support only the last 2 or 3 versions due to the time and budget involved to update the protocol components and parameters. On the other hand, SIP does not have explicit requirements for compatibility among versions. It simply ignores unknown or unsupported headers by default. New features and services can be extended by defining new SIP header information. In this way, new features and services can be easily added without changing existing headers in RFC, which reduces code size and protocol complexity.

As for the vender support, new features and services can be deployed over existing proxy based wireless capable IP environment without any intervention of vendors due to the ASCII nature of SIP. On the other hand, deployment of H.323 based new features and services will likely result waiting for operator's vendors to add the new parameters due to complexity of defining new parameters in H.323.

Scalability is also an important issue as the use of Internet and its services are growing fast. Both SIP and H.323 support different topologies, such as hierarchical and flat. The SIP server or Gatekeeper is the main entity that is responsible for signaling, and they can both make use of DNS, directories, internal translation databases or other location and translation mechanisms to facilitate global deployment. H.323 requires a MCU to processe all conferencing signals, even for small size conferences. Therefore, the MCU tends to be the bottleneck for larger scale conferences. One the other hand, SIP does not require such an entity as the MCU and its conference coordination is fully distributed. This improves scalability and complexity but gives a service provider less control.

Both SIP and H.323 provide services such as Call Hold, Call Transfer, Call Forwarding, Call Waiting, Caller ID, Conferencing and personal mobility, while other new services are continually being added. Compared with H.323, SIP integrates with more Internet services, instant messages and event notification for example, which are now very popular. However, when we analyze them in the aspect of interworking and interoperability with other legacy networks, H.323 has defined more detailed specific standards specifying interoperability with circuit switched networks while SIP, on the other hand, does not provide related specifications to date.

SIP has been adopted by mobile operators as the call signaling and instant message protocol for their third generation networks and it has been under rapid development. Moreover, SIP is also being coupled with the IEEE 802.11 wireless networks for another set of mobile services, such as real time conferencing and online interactive games or learning, which will offer the promise of millions of SIP-enabled wireless devices in the next few years.

Chapter 3

Implementation

3.1 Introduction

In this chapter, we start with presenting a brief introduction to the entire OPNET modeling environment to understand how different components of OPNET work together. Then, we describe our proposed conferencing model and the conferencing entities, including User Agent (UA), Conferencing Server (CSer), SIP Server (SSer) and IP Cloud. Each entity is analyzed and modeled in OPNET simulation software.

3.2 OPNET Environment

OPNET provides an environment that supports modeling of communication networks and distributed systems [14]. It divides the majority of model specification into a set of four hierarchical environments called modeling domains, including Network Domain, Node Domain, Process Domain and External System Domain. These four modeling domains are hierarchically related to each other. Process models and external systems are instantiated in the Node Domain, and node models are instantiated in the Network Domain. Within each domain, objects used to define models may also have hierarchical relationships to each other as well. There is an editor associated with each domain to design and define models under different layers. OPNET also provides tools for all phases of a study, including design, simulation, data collection, and data analysis.

3.2.1 Network Domain

The Network Domain is concerned with the specification of a system in terms of high-level devices called nodes, and communication links between them. The project editor associated with it is used to construct and define the topology of a communication network. A network model may contain any number of communicating entities called nodes and communication links. Nodes are instances of node models, which are developed using the Node Editor. OPNET comes with fixed nodes as well as mobile nodes.

Figure 3.1, which is captured in the Project Editor, illustrates a common network model diagram consisting of two user groups that can access the data from the server located in another group.

3.2.2 Node Domain

The Node Domain is concerned with the modeling of communication devices that can be deployed and interconnected at the network level. In the real world, these devices correspond to various types of computing and communicating equipments such as routers, bridges, workstations, mainframe computers, servers, switches, satellites, firewalls and so on. In OPNET, these devices are called node models.

Node models are developed in the Node Editor and can consist of any number of modules of different types. Some modules offer capabilities that are substantially predefined and can only be configured through a set of built-in parameters. These include various transmitters and receivers that allow a node to be attached to communication links in the network domain. Other modules, called processors, queues, and external systems, are highly programmable, their behavior being prescribed by an assigned process model. Modules interaction with each other using connections such as packet streams and statistic wires in the node domain. The node model interface is defined

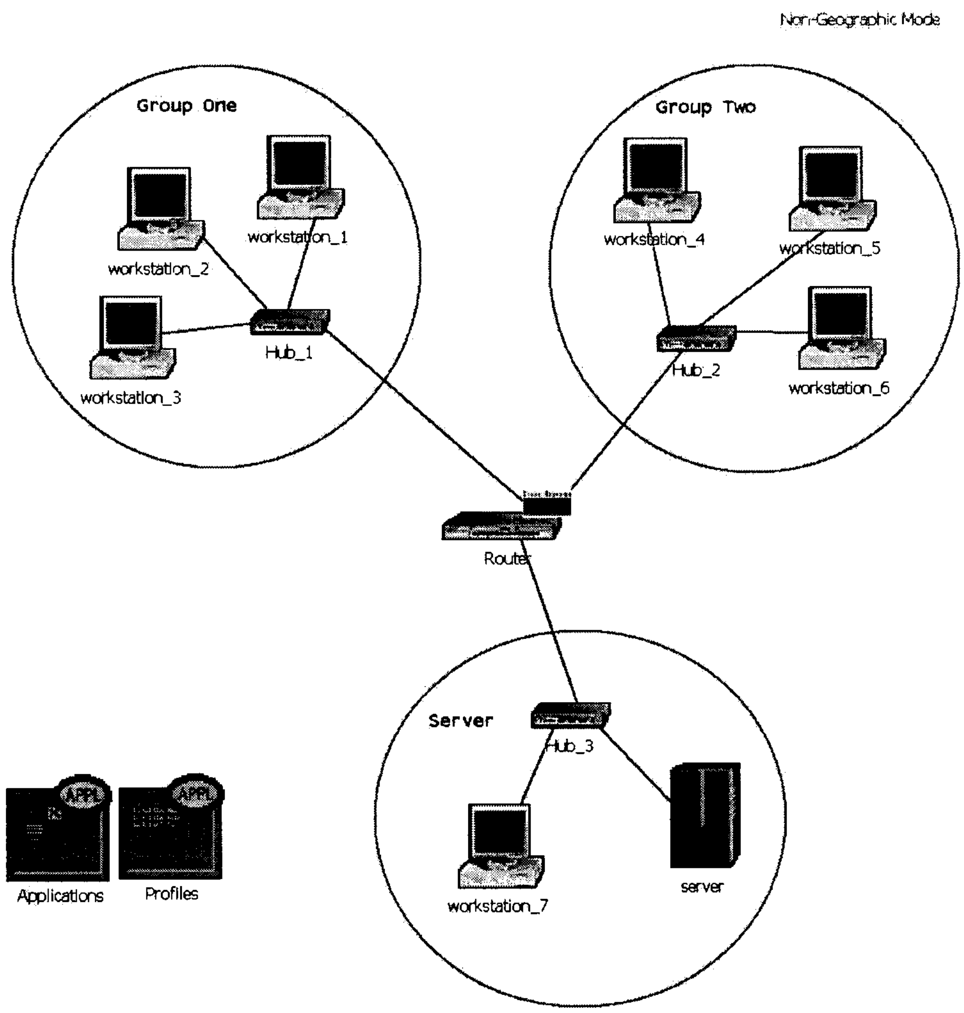


Figure 3.1: Example of Network Model in the Network Domain

in the Node Editor, which determines which parameters of the node model are visible and definable by the user.

Figure 3.2 taken from the Node Domain Editor is an wireless workstation node model that represents the real world mobile workstation with client-server applications running over TCP/IP and UDP/IP, such as a laptop computer.

3.2.3 Process Domain

The Process Domain is mainly concerned with the specification of behavior for the processes that operate within the nodes of the system. Each process that executes in a queue, processor, or esys module is an instance of a particular process model. It is programmable in Process Editor using programming languages called Proto-C or C/C++ to define the protocols and algorithms. In addition, the Process Editor defines the process model interfaces, and declares its child processes as well as external files and packet format that are used by the process.

Processes are designed to respond to interrupts and/or invocations. Interrupts indicate that events of interest have occurred such as the arrival of a message, the allocation of resources, the expiration of a timer, or the change of the state. Only one process can be executed at any time. When a process is interrupted, it takes specific actions in response and then blocks, waiting for the new interrupt. A process that is currently executing can also invoke another child process dynamically during the simulation. When this happen, the invoking process is temporarily suspended to conduct further execution until the invoked process blocks.

Figure 3.5, taken from the Process Editor, shows an example of the State Transition Diagram of the *ip_encap_v4* process model which encapsulates packets coming from the higher layer into IP packets and decapsulates packets arriving from the lower layer in order to forward the user data to the transport layer. This model is designed to be an interface between the higher layers and the IP module. Each process model

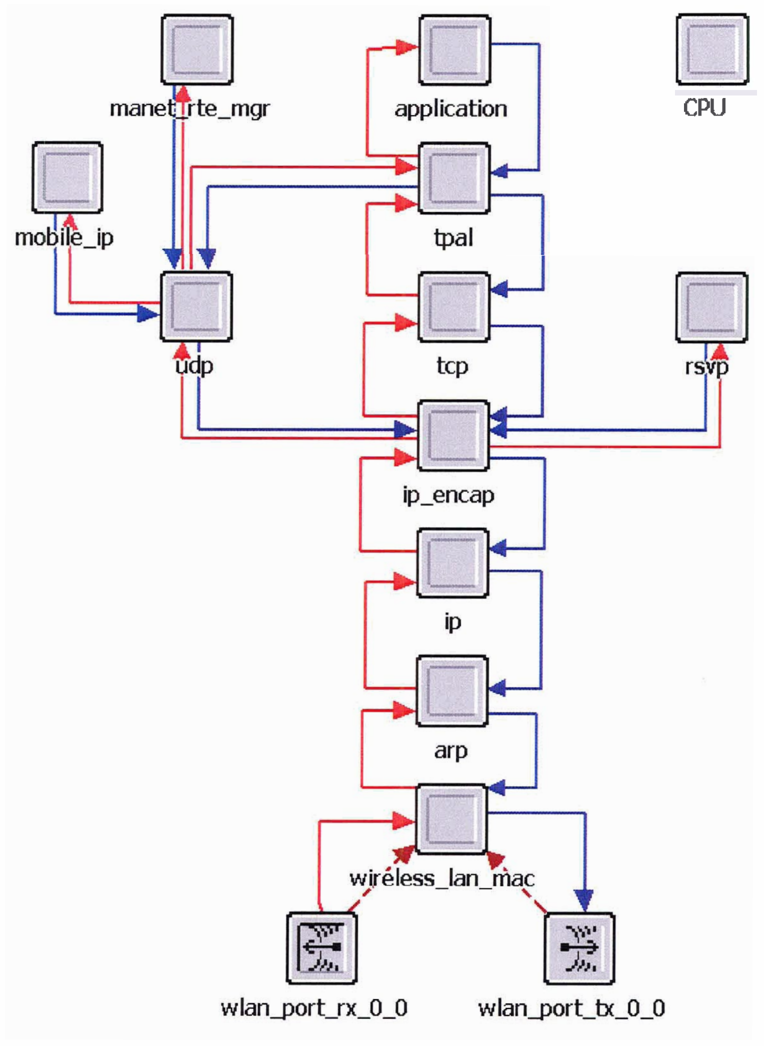


Figure 3.2: Example of Wireless Workstation Node Model in Node Domain

consists of two component types: states and transitions, thus, we call it a state transition diagram (STD). States are further classified into two types: forced and unforced. More detailed information about the states and transitions can be found in OPNET's help documents. The Process Editor includes blocks for the declaration of state variables and temporary variables as well as attributes and functions called by the process KP. Figure 3.4 is a screen shot of the function block of the *ip_encap_v4* process model. The functions are written in a Proto-C language and can be called by the process.

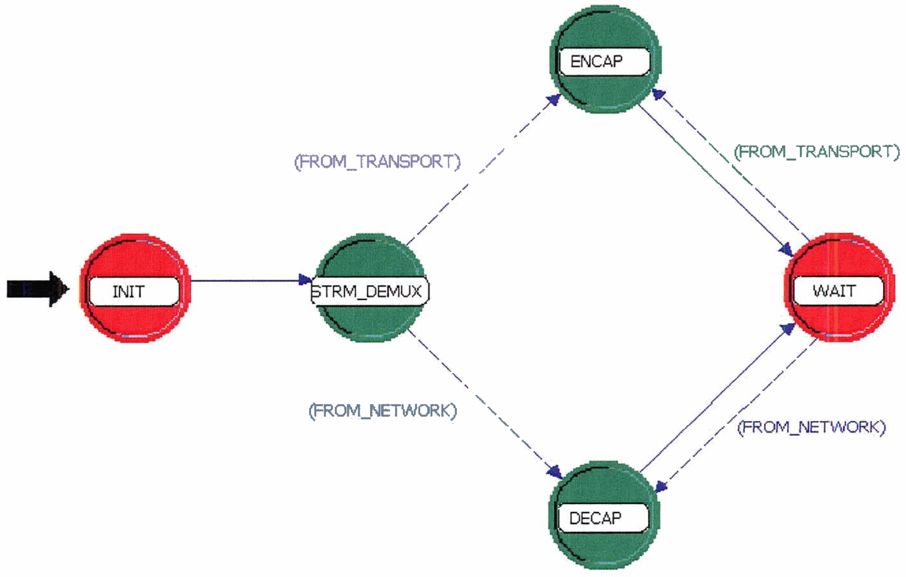
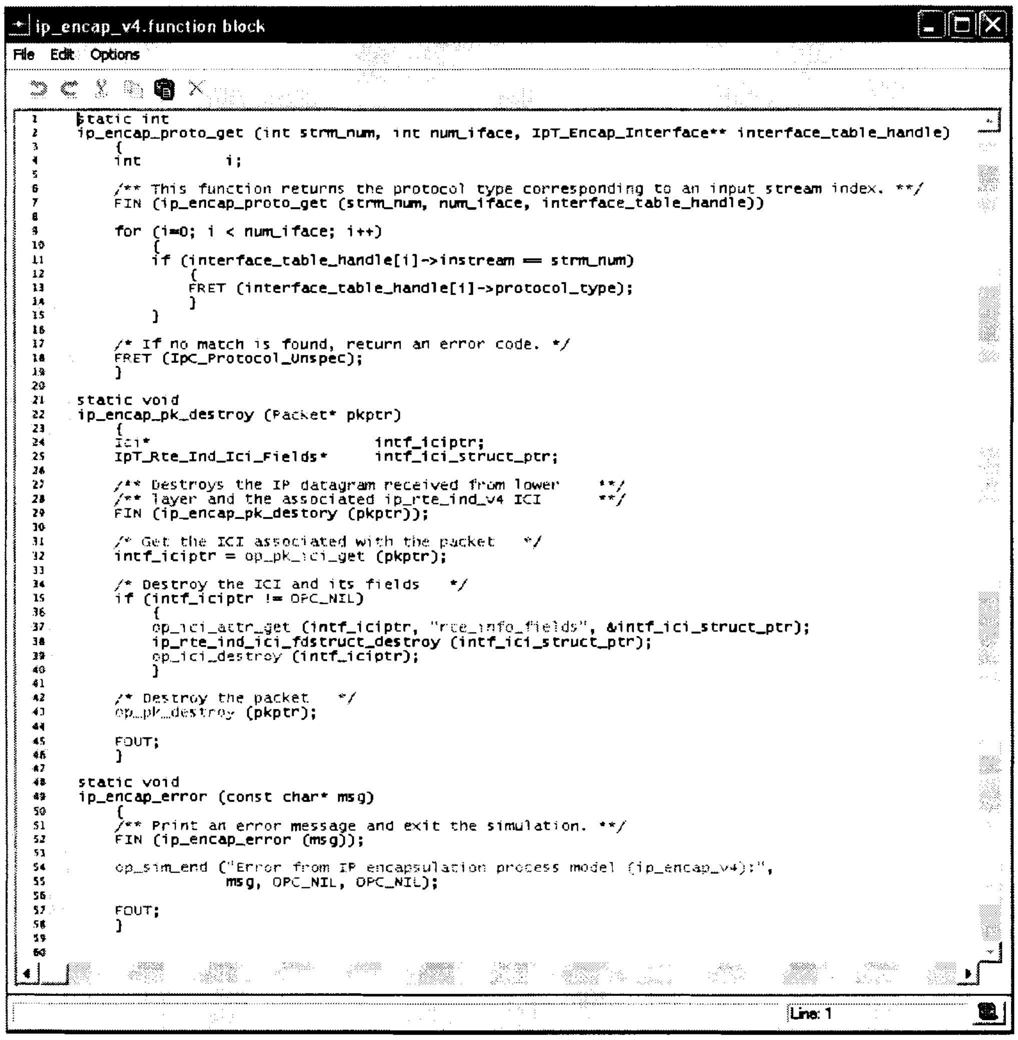


Figure 3.3: Example of State Transition Diagram of IP Encap in Process Domain

3.2.4 External System Domain

The External System Domain is concerned with the specification of communications mechanisms that allow OPNET to include models from external code or systems. During an OPNET simulation, it can accept data from external code, send data to external code, or do both.



```
static int
ip_encap_proto_get (int strm_num, int num_iface, IpT_Encap_Interface** interface_table_handle)
    {
    int         i;

    /** This function returns the protocol type corresponding to an input stream index. **/
    FIN (ip_encap_proto_get (strm_num, num_iface, interface_table_handle))

    for (i=0; i < num_iface; i++)
        {
        if (interface_table_handle[i]->instream == strm_num)
            {
            FRET (interface_table_handle[i]->protocol_type);
            }
        }

    /* If no match is found, return an error code. */
    FRET (IpC_Protocol_Unspec);
    }

static void
ip_encap_pk_destroy (Packet* pkptr)
    {
    Ici*                        intf_iciptr;
    IpT_Rte_Ind_Ici_Fields*     intf_ici_struct_ptr;

    /** Destroys the IP datagram received from lower     **/
    /** layer and the associated ip_rte_ind_v4 ICI       **/
    FIN (ip_encap_pk_destory (pkptr));

    /* Get the ICI associated with the packet   */
    intf_iciptr = op_pk_ici_get (pkptr);

    /* Destroy the ICI and its fields   */
    if (intf_iciptr != OPC_NIL)
        {
        op_ici_attr_get (intf_iciptr, "rte_info_fields", &intf_ici_struct_ptr);
        ip_rte_ind_ici_fdstruct_destroy (intf_ici_struct_ptr);
        op_ici_destroy (intf_iciptr);
        }

    /* Destroy the packet   */
    op_pk_destroy (pkptr);

    FOUT;
    }

static void
ip_encap_error (const char* msg)
    {
    /** Print an error message and exit the simulation. **/
    FIN (ip_encap_error (msg));

    op_sim_end ("Error from IP encapsulation process model (ip_encap_v4):",
                msg, OPC_NIL, OPC_NIL);

    FOUT;
    }
```

Figure 3.4: Function Block Within IP Encap Process Model

Relationship between hierarchical levels in the OPNET environment is shown in Figure 3.1.

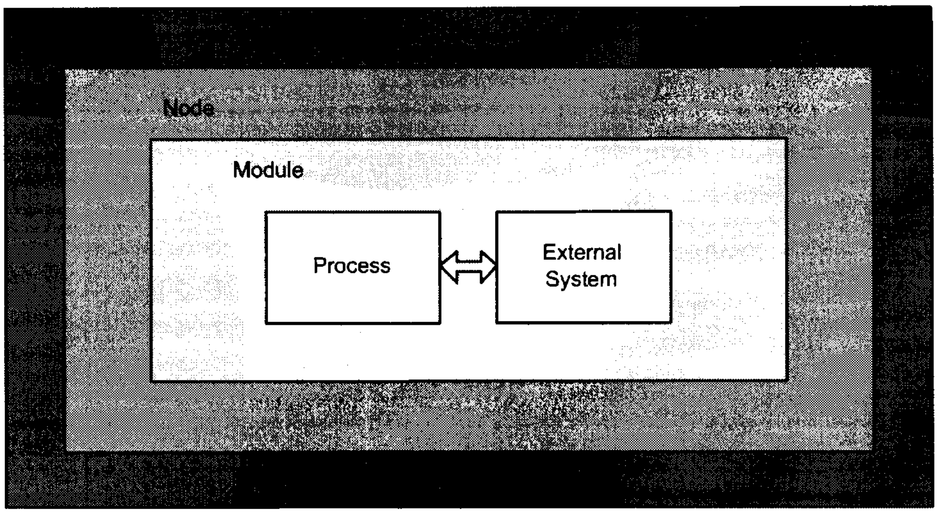


Figure 3.5: Relationship between Hierarchical Levels in the OPNET Environment

3.3 Our Model

When designing a conferencing system, the main goal is to create a scalable and distributed conferencing system while maintaining the required QoS. Furthermore, the limitations of the target users, small handheld devices, and the IEEE 802.11 networks for the conferencing system must also be taken into consideration. Therefore, we propose the following conferencing architecture based on Singh's et al SIP based conferencing model [13] to address those limitations.

We consider a large conference with hundreds of participants that are situated in several subnets distributed geographically and interconnected with each other by the Internet. In Figure 3.6, we demonstrate a conferencing architecture with four subnets which represent four different wireless LAN locations. We model the four subnets in the OPNET environment and each subnet can interconnect with the outside IP networks through the IP cloud as discussed later in this chapter. Figure 3.7 shows a subnet with five user agents. In our proposed model, a single Conferencing Server is present in each subnet. All the User Agents within that subnet communicate with the Conferencing Server in a way similar to the unicast receive and multicast send conferencing model as we discussed in Chapter 2. Then, the Conferencing Server communicates with other Conferencing Servers in a Full Mesh based systerm. The conferencing entities in our conferencing model include the SIP User Agent (UA), the Conferencing Server (CSer), the SIP Server (SSer) and the IP Cloud. We describe each of them in more detail in the following sections.

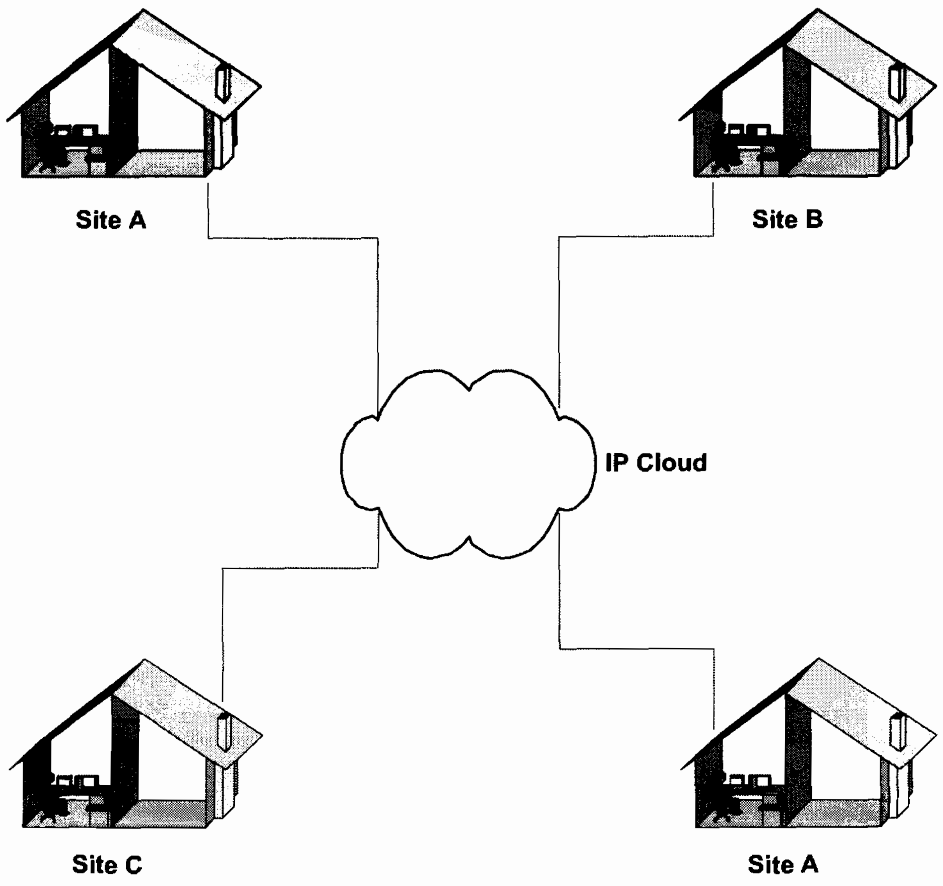


Figure 3.6: Conferencing Architecture with 4 Subnets

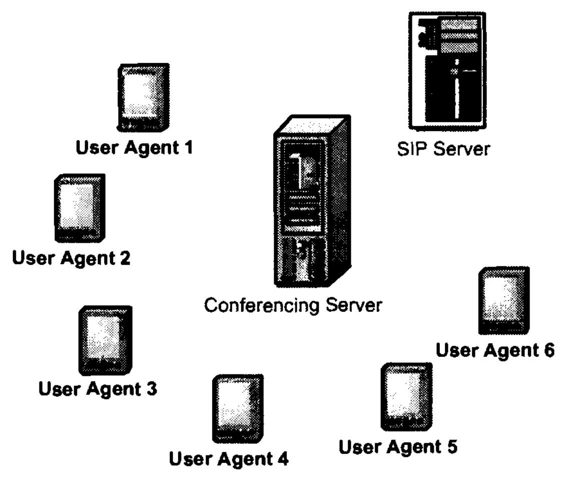


Figure 3.7: Subnet Topology with 6 User Agents

3.3.1 User Agent (UA)

The UA is a SIP enabled end user. It represents the mobile handheld device user, which can travel from Subnet A to Subnet B during a conference as shown in Figure 3.6. A SIP UA contains both a SIP client application and a SIP server application. OPNET models these applications into two parts called User Agent Client (UAC) and User Agent Server (UAS). They are embedded into the node model's application layer. The UAC is responsible for initializing SIP requests while the UAS is used to generate SIP responses. The UA usually operates as both a UAC and a UAS during a session, and it must be capable of establishing a media session with another UA as well. In addition, the UA is able to utilize a voice codec that supports silent suppression, such as G.711, G.723.1, G.726, G.728 and G.729 as we discussed in Chapter 2.

Figure 3.8 shows our UA node model in OPNET. It is derived from the *wlan_wkstn_adv* node model by exchanging its application layer module for our modified process model *gna_clsvr_mgr_client*. We will describe the process model in more details in the following section. In addition, we customize a low CPU processing speed and a small buffer size for the UA node model to represent the handheld device user. The UA can initiate, receive and terminate a call by sending corresponding SIP request messages.

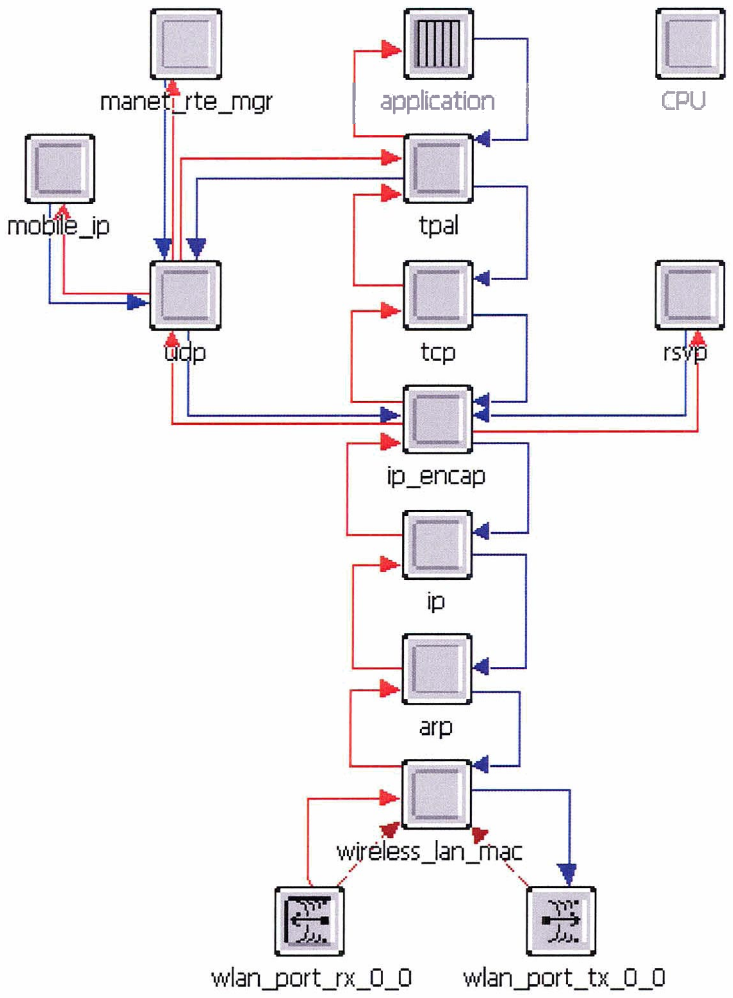


Figure 3.8: The User Agent Node Model

3.3.2 Conferencing Server (CSer)

There is a Conferencing Server (CSer) in each subnet. It is only responsible for handling the media streams using Real Time Protocol (RTP). Each UA, User Agent 1 for example, within its subnet sends its media stream to the CSer, which in turn sends the received media streams to each of the other user agents within its subnet via unicast. For example, the CSer in Figure 3.7 sends the media stream received from User Agent 1 to User Agent 2, 3, 4 and 5. At the same time, the CSer mixes all the received media streams within its subnet and sends the mixed stream to the other subnets via multicast. The CSer also mixes the incoming media streams from all other subnets and sends the result to all the user agents within its subnet, which greatly reduces the data processing requirements for the UA. We can summarize the functions of the CSer in each subnet as follows:

1. Receives media streams from the UAs in its subnet and mixes them before sending to the other UAs within its subnet via unicast.

2. Mixes all the media streams received from its subnet and sends the mixed stream to other subnets using multicast.

3. Receives media streams from other subnets and mixes them before sending the result to the UAs in its subnet using multicast.

A CSer may be serving many conferences and many participants at an instant of time. If the number of users exceeds the maximum number of users that the CSer can handle, we can put another CSer into that subnet. Many CSer units can co-exist in the same domain, and each of them is responsible for different fractions of the users. Furthermore, the CSer stores the information of all other CSers that are involved in the conferencing. It also stores a flag to indicate whether multicast is supported.

Figure 3.9 shows our CSer node model in OPNET. It is not only responsible for mixing the media streams for appropriate destinations, but also acts as the access point for the UAs and routes the data packets to their destinations. It is derived from the *wlan_ethernet_router_adv* node model by adding a mixer in its application layer with our modified process model *gna_clsvr_mgr_server*. We will describe the process model in more details later.

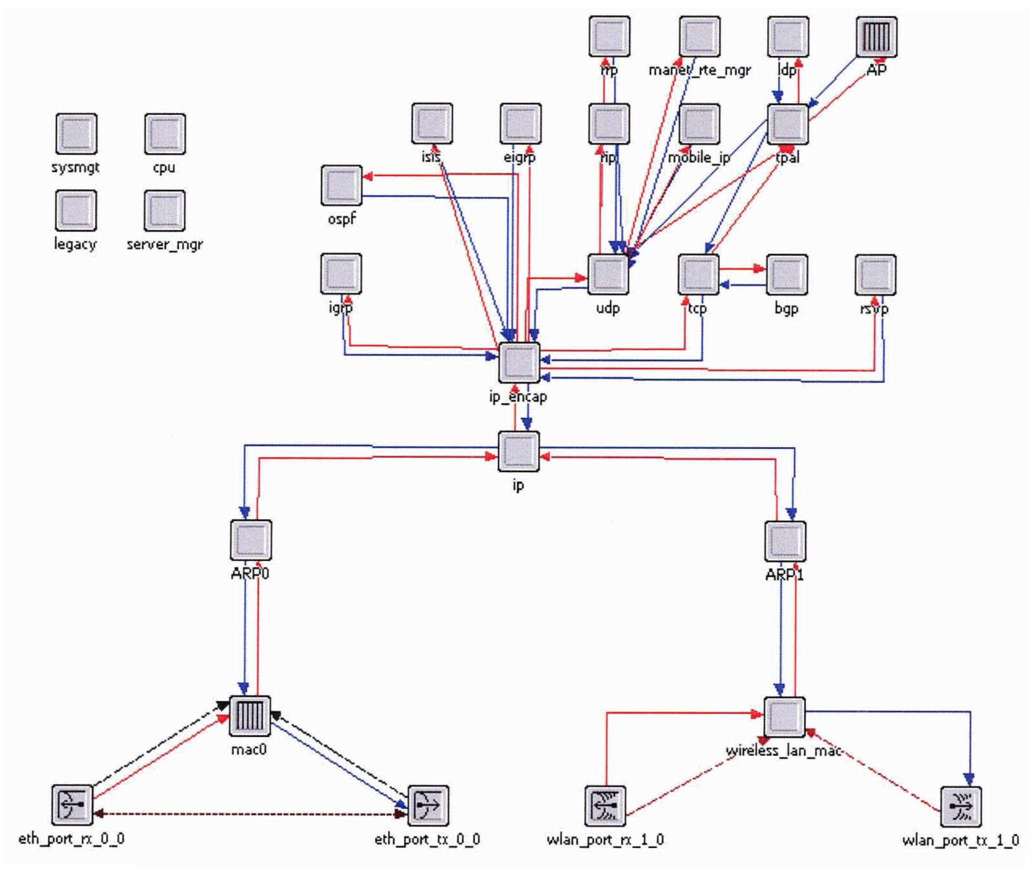


Figure 3.9: The Conferencing Server Node Model

3.3.3 SIP Server (SSer)

The SIP Server (SSer) is an entity that accepts SIP requests and responds to them. It provides service to the UA, but it is different from the UA in that it does not issue requests and only responds to requests from the UA. In addition, the SSer has no media capability. We use the *sip_proxy_server* node model provided by the OPNET node model library in our simulation. It represents the SSer node and supports SIP UAS service. One of the important functions for SSer is its terminal mobility management. As mentioned in Chen's book [31], SIP appears to be the only application layer protocol which supports terminal mobility. It supports pre-call terminal mobility as well as mid-call terminal mobility. The *sip_proxy_server* node model does not provide mobility management capability at this time, so we mainly use the SSer to set up the call session in our simulation and focus on the conference media stream QoS performance at this stage. However, we will study their QoS performance when the UAs are conducting the conference in a mobility environment.

3.3.4 IP Cloud

The IP Cloud is used to represent the outside IP network traffic. During our simulation, subnets communicate with each other through the IP Cloud node model. We use the *p8_cloud_adv* node model in OPENT to model the Internet traffic. The IP packets arriving on any cloud interface are routed to the appropriate output interface based on their destination IP address, and the data transmission brings in packet transmission delay and latency. Furthermore, we can define different packet discard ratios to represent different levels of real world IP network traffic congestion and determine the number of packets dropped out of the packets being transferred.

3.4 Voice Packet Format

The primary information-carrying object in an OPNET simulation is a packet. Each packet has a format that defines which fields it contains, and it carries the actual values that can be assigned and extracted by the processes. Packets can also be generated spontaneously by processes in the process domain. Then, they are forwarded to other objects. When a packet is received by a process, the information can be analyzed and used as the basis for making decisions, including copying, modifying, destroying, or relaying the packet, as well as changing the system state.

We define a voice packet format to be used in our simulation, which is shown in Figure 3.10. The voice packet format is derived from the gna packet format which is predefined in OPNET. We add three fields to the packet format, including the subnet Object ID, node Object ID and the original node Object ID as shown in the last row. When generating a packet, each field will be initialized and assigned a value by the node model that generates it. The three items of Object ID information stored in the packet fields will be used to identify the data source to determine the next executive.

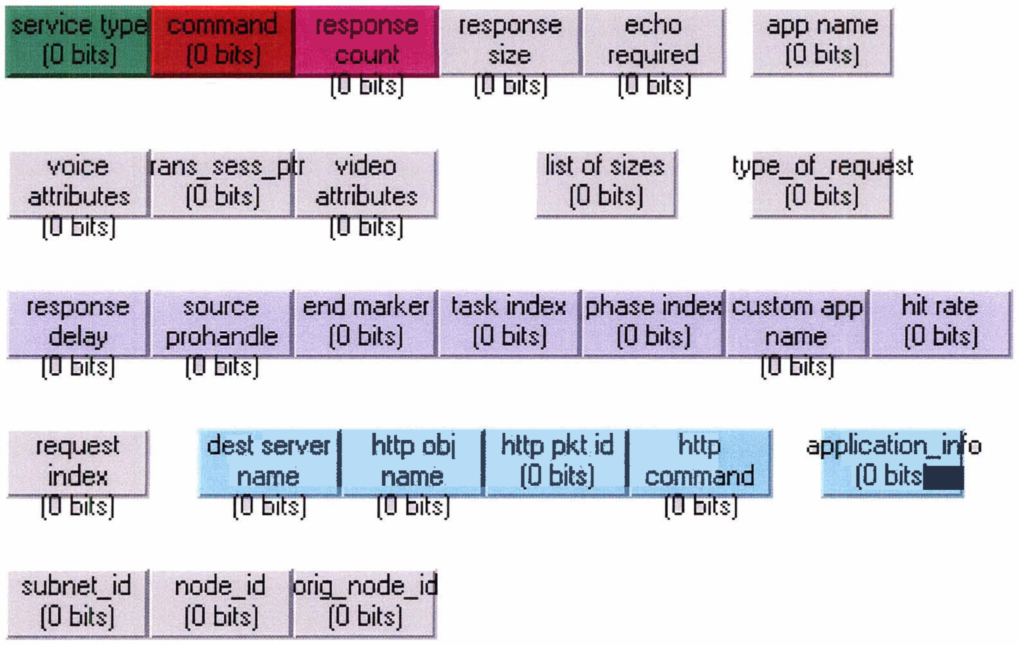


Figure 3.10: Voice Packet Format

In order to illustrate the big picture of how these three packet fields are used to determine process executives, we present the following example in Figure 3.11 to illustrate the process in point to point voice communication.

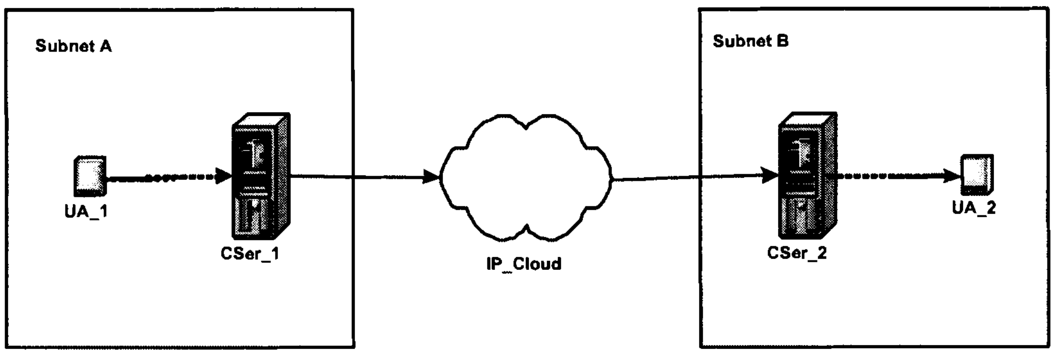


Figure 3.11: Data Flow Example

Step 1. *UA_1* in subnet A generates a voice packet and sends it to the *CSer_1*.
Step 2. *CSer_1* receives the voice packet and extracts the subnet_id specific information of the packet. Then, it sends the received packets to the other conferencing entities based on the following rules as shown in Figure 3.12.
Step 3. The *IP Cloud* receives the voice packet from *CSer_1*, then it sends it to other CSers participating in the conference.
Step 4. *CSer_2* receives the voice packet from the *IP Cloud* and extracts the subnet_id specific information of the packet. Then, it sends the packets to the UAs based on the following rules as shown in Figure 3.12.

Step 4. *UA_2* receives the voice packet from *CSer_2*.

3.5 Child Processes

The *gna_clsvr_mgr_client* process model used in the UA node model and the *gna_clsvr_mgr_server* process model used in the CSer node model are customized and tailored to the needs of our voice conferencing model. They are implemented in the application layer and are derived from the process model *gna_clsvr_mgr* provided by OPNET's process model library. We created a *voice_client* child process for the *gna_clsvr_mgr_client* process model and a *voice_server* child process for the *gna_clsvr_mgr_server* process model, respectively. These two child processes are implemented in the application layer using the state transition diagrams, shown in Figure

3.13 and Figure 3.14. The child process and packet format declaration can be accessed via the menu in the Process Editor.

Figure 3.15 shows the send state flowchart for the *voice_client* child process state transition diagram while Figure 3.16 shows the receive state flowchart for the *voice_server* child process state transition diagram.

```
if (subnet_id == its own subnet_id)
{
   send the packet to the IP Cloud;
   extract the orig_node_id specific information
   of the packet;

   switch (orignal_node_id)
   {
      case UA_1:
        send packet to other UAs within the same subnet
        except UA_1;

      case UA_2:
        send packet to other UAs within the same subnet
        except UA_2;

      etc.
   }

   change the node_id field to CSer_1's node_id;
 }

 else if (subnet_id != its own subnet_id)
     {
        extracts the node_id specific information
        of the packet;

         if (node_id != its own node_id)
            send the packet to all the UAs within
            its subnet;
          else
            destroys the packet;
     }
```

Figure 3.12: CSer packet routing algorithm

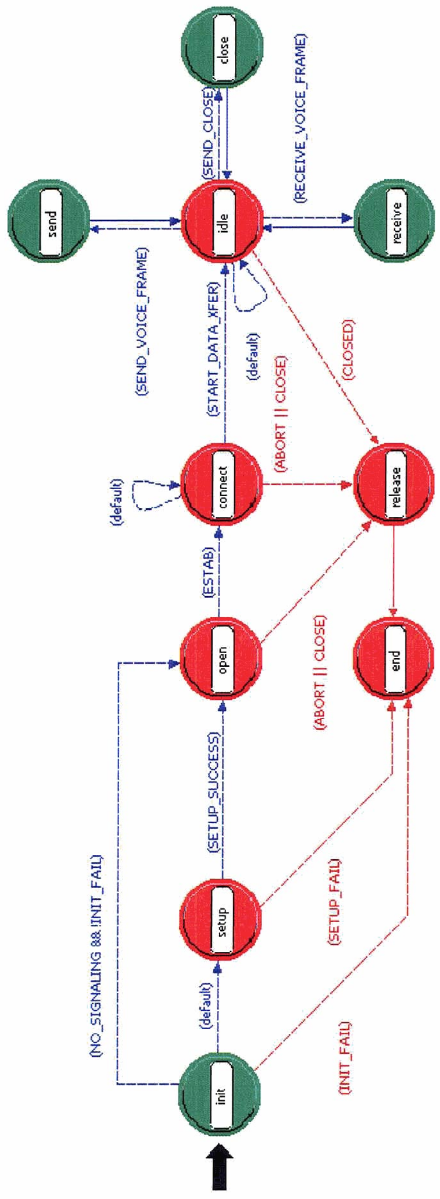


Figure 3.13: State Transition Diagram of the voice_client Process Node

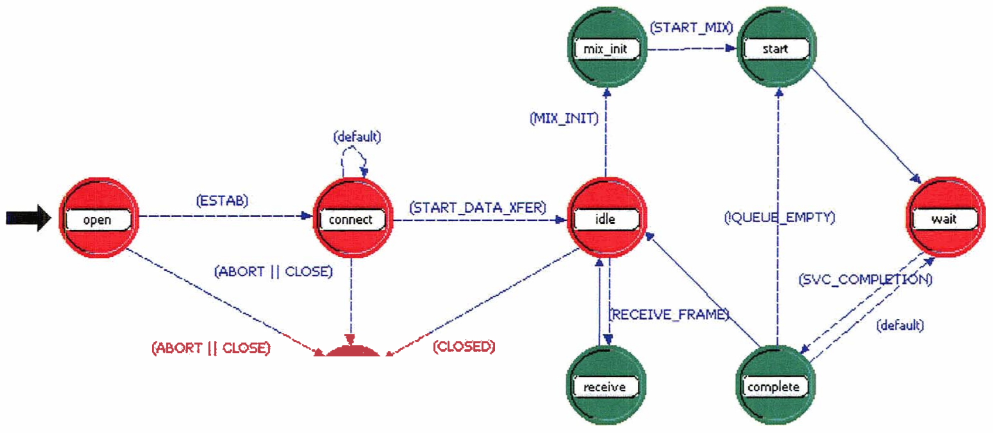


Figure 3.14: State Transition Diagram of the voice_server Process Node

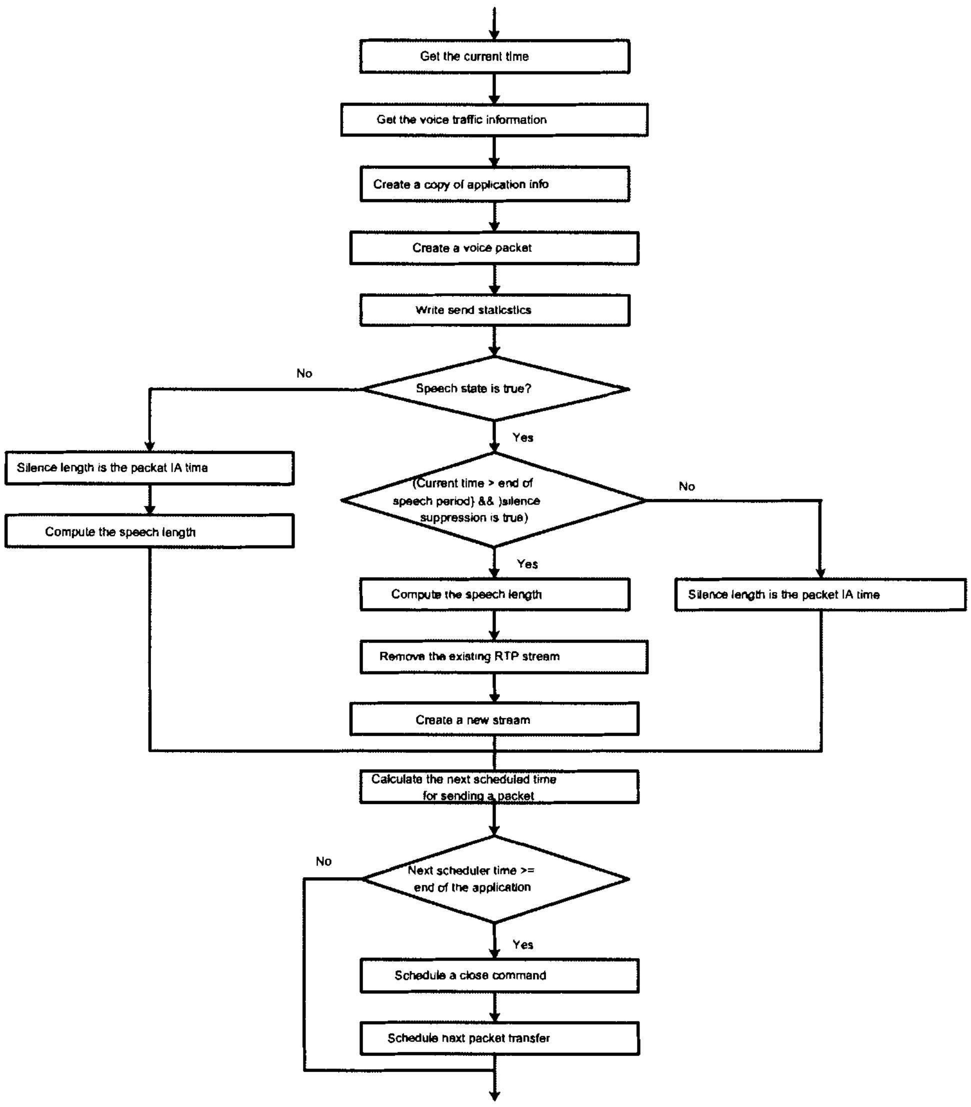

Get the current time
Get the voice traffic information
Create a copy of application info
Create a voice packet
Write send statistics
Speech state is true?
No
Yes
Silence length is the packet IA time
Compute the speech length
(Current time > end of speech period) && (silence suppression is true)
No
Yes
Compute the speech length
Silence length is the packet IA time
Remove the existing RTP stream
Create a new stream
Calculate the next scheduled time for sending a packet
Next scheduler time >= end of the application
No
Yes
Schedule a close command
Schedule next packet transfer

Figure 3.15: Send State Flowchart of voice_client Process State Transition Diagram

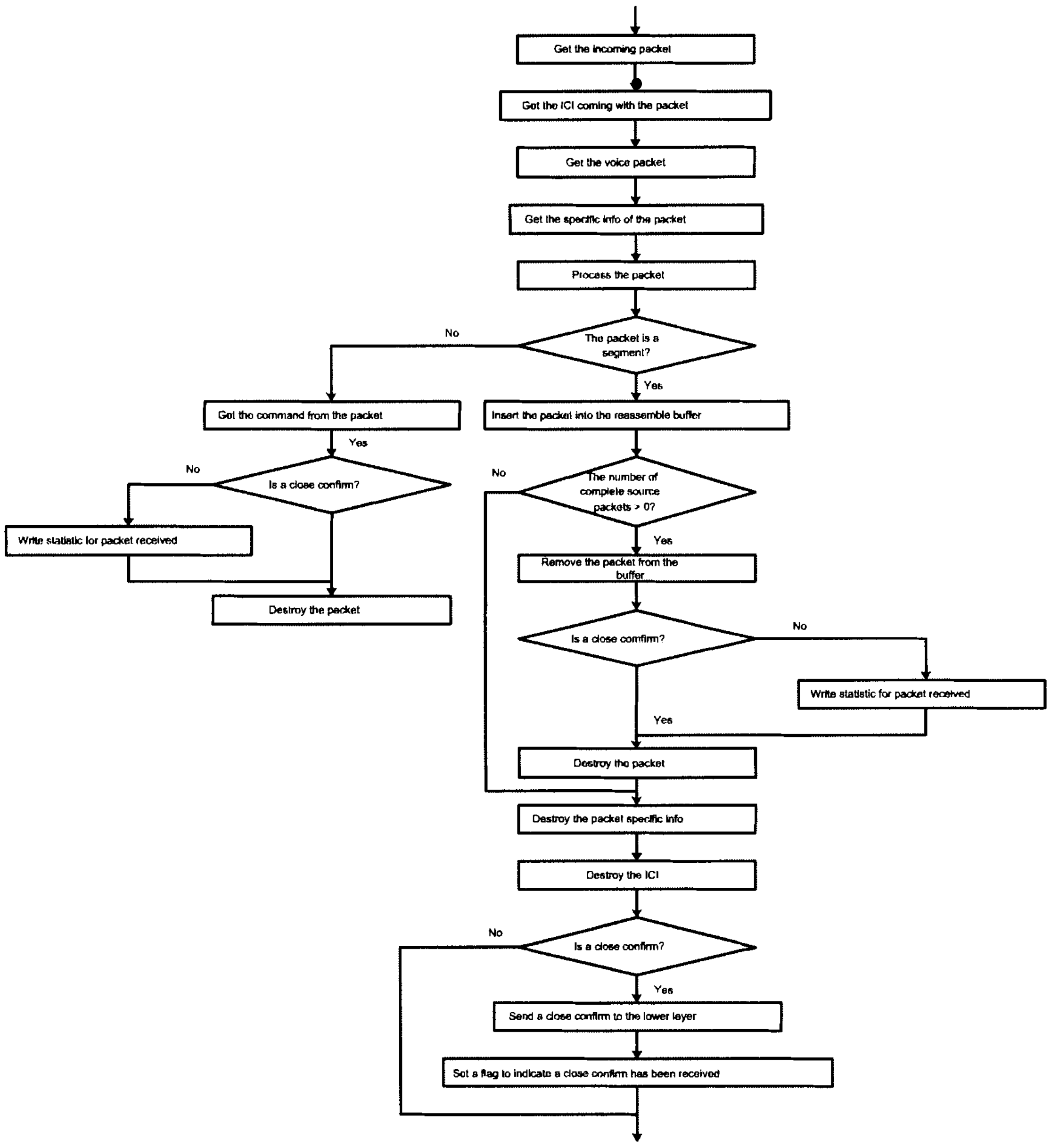

Get the incoming packet
Get the ICI coming with the packet
Get the voice packet
Get the specific info of the packet
Process the packet
No
The packet is a segment?
Yes
Get the command from the packet
Yes
No
Is a close confirm?
Write statistic for packet received
Destroy the packet
Insert the packet into the reassemble buffer
No
The number of complete source packets > 0?
Yes
Remove the packet from the buffer
Is a close comfirm?
No
Write statistic for packet received
Yes
Destroy the packet
Destroy the packet specific info
Destroy the ICI
No
Is a close confirm?
Yes
Send a close confirm to the lower layer
Set a flag to indicate a close confirm has been received

Figure 3.16: Receive State Flowchart of voice_server Process State Transition Diagram

Chapter 4

Simulation Results

4.1 Introduction

In this section, we describe the simulations of a series of scenarios obtained by running the OPNET simulation tool to analyze the performance of the general types of conferencing models as well as our proposed model. The network topologies used for simulations are shown in Figure 4.1, Figure 4.2 and Figure 4.3. Our proposed model simulation network topology is shown in Figure 4.4.

We simulated three scenarios to analyze and compare the performance of various QoS aspects. The simulation parameters are shown in Table 4.1. We customized the UA CPU processing speed and the buffer size based on the HP iPAQ 64MB Pocket PC (HP2490) model. Meanwhile, we defined the CSer CPU processing speed and buffer size based on the Cisco 7800 Series Media Convergence Servers 7835-11 model. In order to illustrate the bandwidth efficiency of using a silence suppression voice codec, we selected a G.729 voice codec in the first scenario and a G.729 (silence) voice codec in scenario two. In the first scenario, we compared the voice conference QoS performance under a small number of users for all the conferencing models while in scenario two, we increased the number of users for the UIMO conferencing model as well as our proposed conferencing model using G.729 (silence) voice codec to evaluate their performance. After that, we compared the performance of static users with mobile users using our proposed conferencing model in scenario three.

We kept track of the node statistics as well as the global wireless statistics such as the end to end delay, jitter, packet loss, and the amount of traffic sent or received by the UA and the CSer in different simulation network topologies. The statistics that were collected and analyzed are listed in Table 4.1.

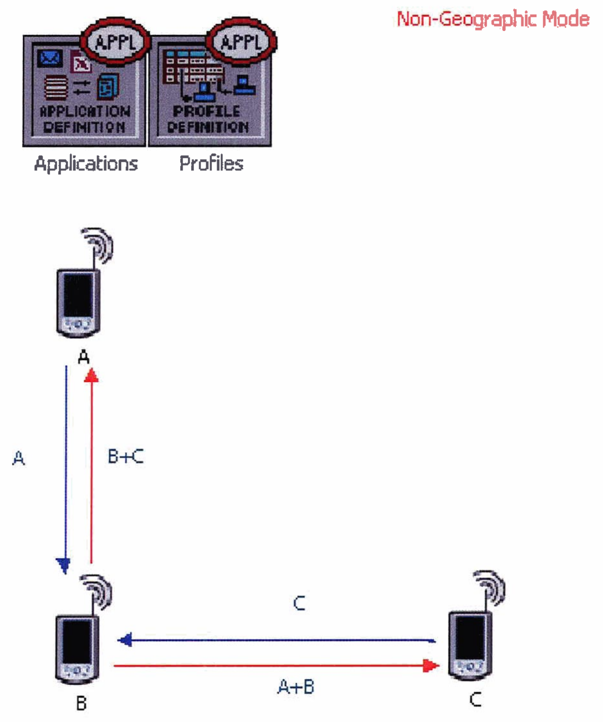


Figure 4.1: End Mixing Conferencing Model Simulation Network Topology

The simulation network topology shown in Figure 4.1 represents the end mixing conferencing model. User B acts as the media stream mixer. It receives voice packets sent by user A and user C, then mixes the received streams with its own media stream before sending out to user A and user C, respectively. We simulated this conferencing model only in scenario one.

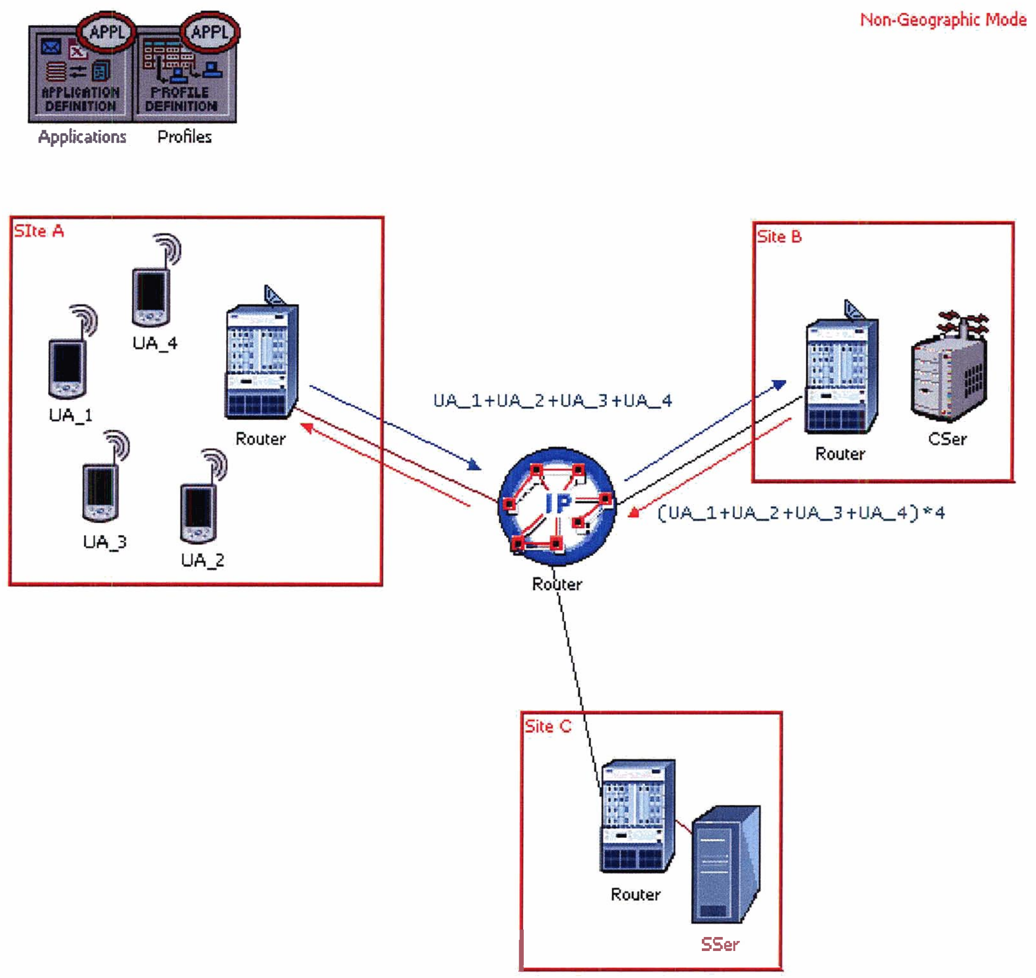


Figure 4.2: Unicast Receive and Unicast Send Conferencing Model Simulation Network Topology

The simulation network topology shown in Figure 4.2 represents the unicast receive and unicast send conferencing model. During the simulation, UAs are in the same subnet. They send their requests to the SIP server located in Site C to establish a conference connection. After they received the Success signaling response from the SIP server, they start to send their media streams to the conferencing server located in Site B via the intermediate routers. The IP cloud is responsible for routing the incoming packets to the appropriate destination based on the packet header information. The conferencing server receives the voice packets and then sends them to all the UAs that are participating in the conference using unicast routing protocol.

The simulation network topology shown in Figure 4.3 represents the unicast receive and multicast send conferencing model. This network topology is similar to Figure 4.2. The main difference is that, the conferencing server sends the received voice packets to all UAs using a multicast routing protocol. In addition, the routers are required to support the multicast routing protocol in this conferencing model. In scenario two, we increased the conference user scale simply by adding more subnets and adding more users in each subnet in a similar way as shown in Figure 4.4.

In our proposed conferencing model, there is a conferencing server in each subnet, which is responsible for handling the media streams as we discussed in Chapter 2. In the first scenario, all the UAs are in the same subnet. Then we expanded the conference by adding more subnets and more users in each subnet in scenario two. Furthermore, we compared the conferencing performance for static users and mobile users in scenario three using this model.

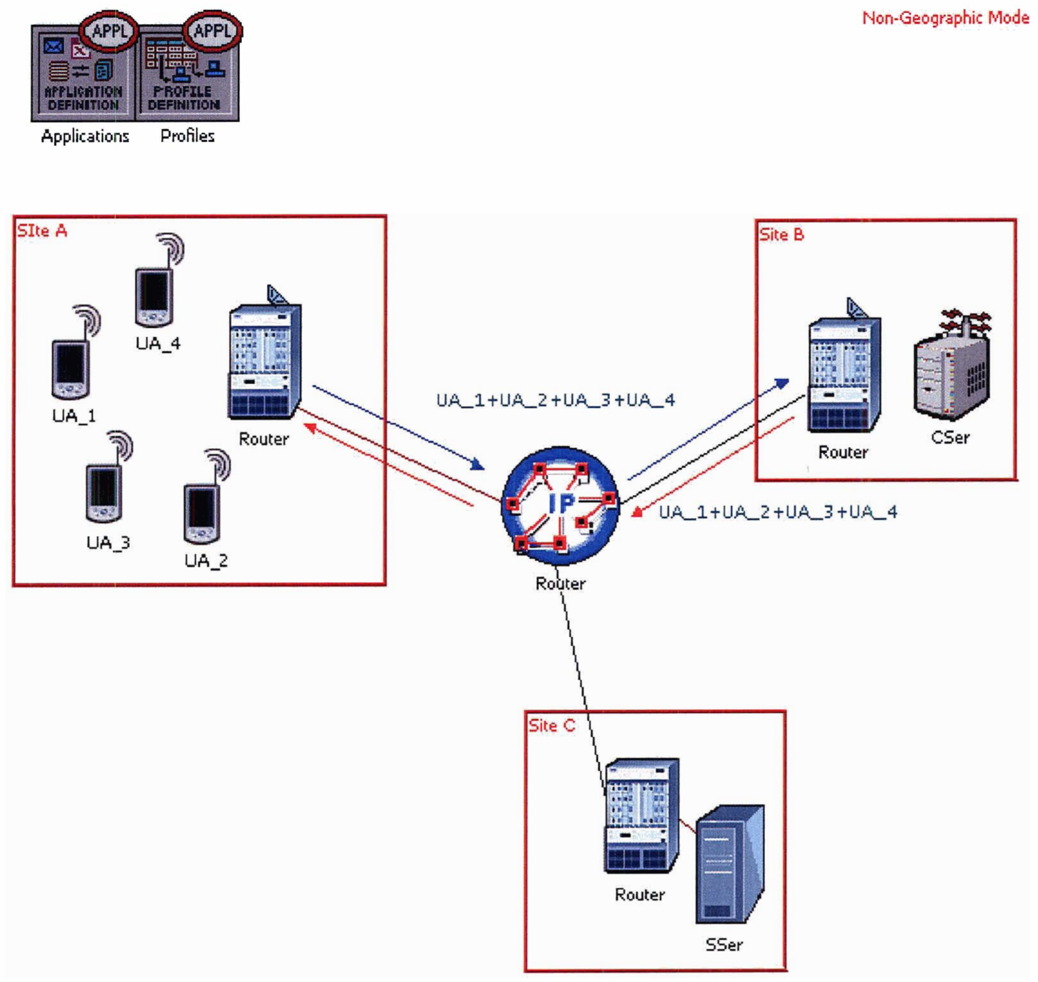


Figure 4.3: Unicast Receive and Multicast Send Conferencing Model Simulation Network Topology

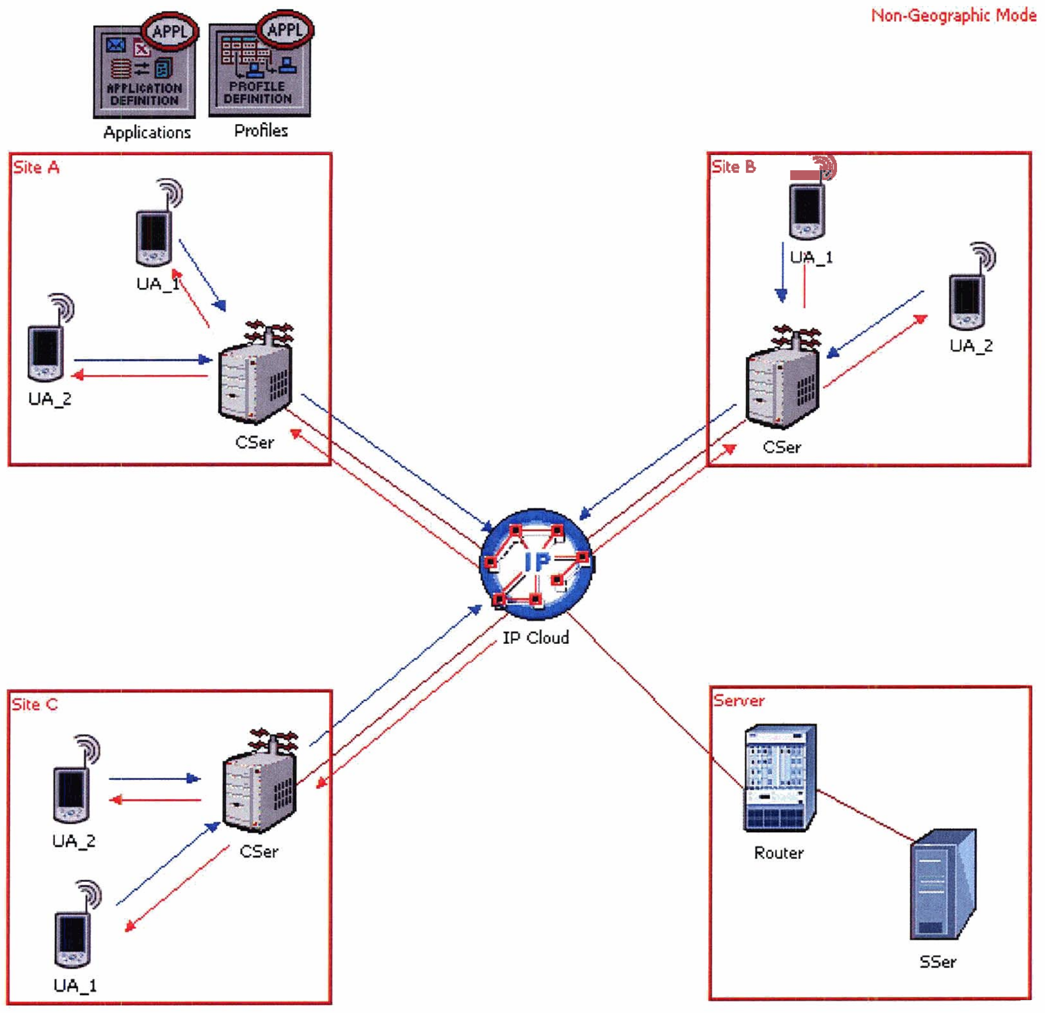


Figure 4.4: Our Proposed Conferencing Model Simulation Network Topology

Table 4.1: Simulation Parameters

Model	Parameter	Value
Application Configuration	Incoming Silence Length (seconds)	exponential(0.65)
	Outgoing Silence Length (seconds)	exponential(0.65)
	Incoming Talk Spurt Length (seconds)	exponential(0.352)
	Outgoing Talk Spurt Length (seconds)	exponential(0.352)
	Voice Encoder Schemes	G.729 (silence)
	Voice Encoder Schemes	G.729
	Voice Frames per Packet	1
	Compression Delay (seconds)	0.02
	Decompression Delay (seconds)	0.02
	Signaling	SIP
Profile Configuration	Start Time Offset(Seconds)	constant (0)
	Operation Mode	Serial(Ordered)
	Start Time (Seconds)	uniform(20,30)
	Duration (Seconds)	End of Simulation
UA	UA CPU Processing Speed Multiplier	0.5
	Application Supported Profiles	G729
	Application Supported Profiles	G729SS
	WLAN Data Rate	11 Mbps
	WLAN Physical Characteristics	DSSS
	WLAN Buffer Sizes (bits)	64000 bits
	SIP UAC Service	Enable
CSer	CSer CPU Processing Speed Multiplier	3.4
	Application Supported Services	All
	WLAN Data Rate	11 Mbps
	WLAN Physical Characteristics	DSSS
	WLAN AP Functionality	Enable
	WLAN Buffer Sizes (bits)	2048000 bits
	SIP UAC Service	Enable
IP Cloud	Packet Discard Ratio	1.00

Table 4.2: Simulation Objectives

Objectives	Statistics to Acquire
Global Wireless LAN Statistics	Global Wireless LAN Data Dropped(bits/sec) Wireless LAN Media Access Delay(sec) Wireless LAN Throughput(bits/sec)
UA Node Statistics	UA Voice Application Jitter(sec) UA Voice Application Packet End-to-End Delay(sec) UA Voice Application Traffic Received(bytes/sec) UA Voice Application Traffic Sent (bytes/sec)
CSer Node Statistics	CSer Voice Application Jitter(sec) CSer Voice Application Packet End-to-End Delay(sec) CSer Voice Application Traffic Received(bytes/sec) CSer Voice Application Traffic Sent(bytes/sec)

4.2 Scenario One

In the first scenario, we selected G.729 as the voice codec. We started our simulation from a small number of users, two UAs for the end mixing conferencing model and four UAs for the other three conferencing models. The reason was that the end mixing conferencing model can only handle a small number of users, so we compared this conferencing model with others using a similar number of users.

We defined the same traffic parameters for each UA. Therefore, each UA sends out the same amount of traffic no matter what the conferencing model is. Figure 4.5 shows the voice application traffic sent by the UA in packets per second. Similar to this, the amount of traffic received by the CSer is predictable, which is twice as much as sent by each UA in the end mixing conferencing model, and four times as much as sent by each UA in the other three conferencing models. This is an ideal scenario in which the number of users is quite small, so the traffic will not cause network congestion and will not cause packet dropped during the transmission. Figure 4.6 shows the voice application traffic received by the CSer in packets per second.

On the other hand, the traffic sent by the CSer and received by the UA varies from each other depending on the simulation network topology and the total number of users that are participating in the conference. Figure 4.7 shows the voice application traffic received by each UA in packets per second, and Figure 4.8 shows the total amount of voice application traffic sent by the CSer in packets per second. As we can see from the results, there are some voice packets lost before they reach the UA in the UIUO conferencing model. This may be due to the duplicated media streams which cause network congestion and data loss during the transmission.

Figure 4.9 and Figure 4.10 show the voice application jitter statistics in seconds for the UA and CSer, respectively. In voice over IP applications, the jitter is defined to be the variation in the time between packets arriving. It is usually caused by network congestion, timing drift, or route changes. The results indicate that the end mixing

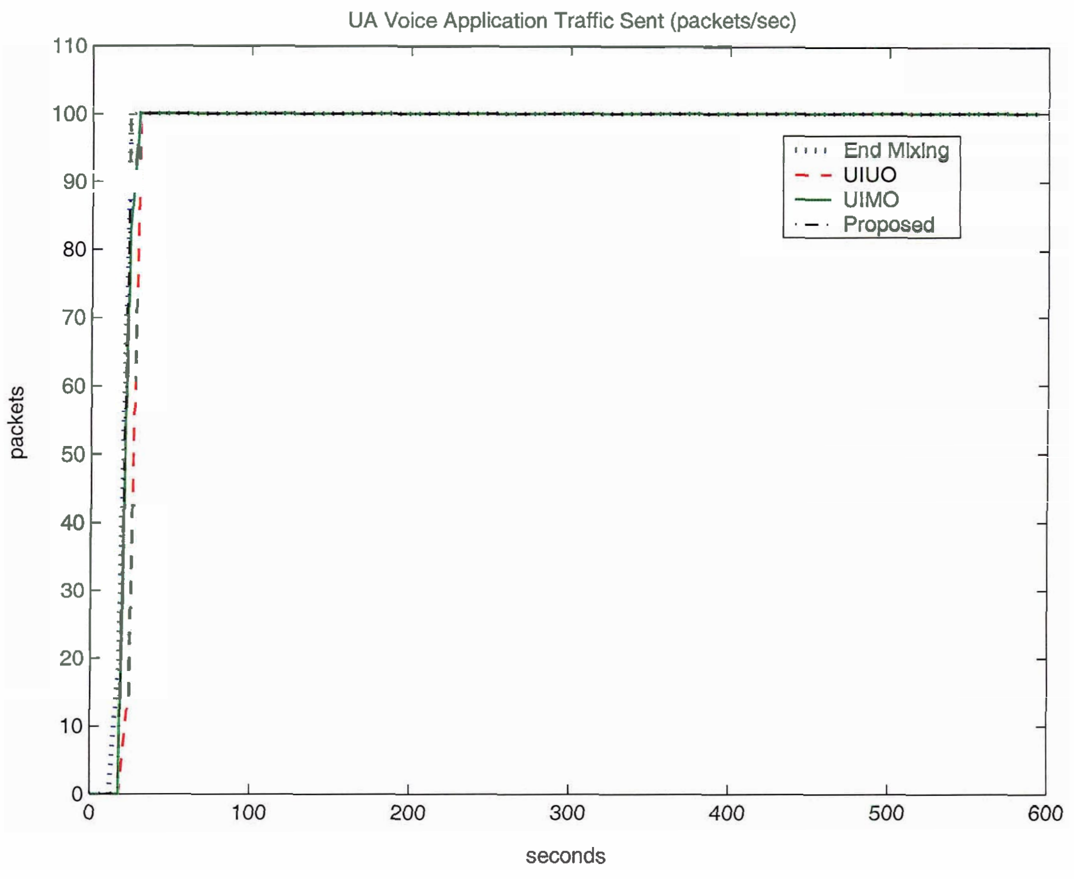


Figure 4.5: UA Voice Application Traffic Sent (packets/sec)

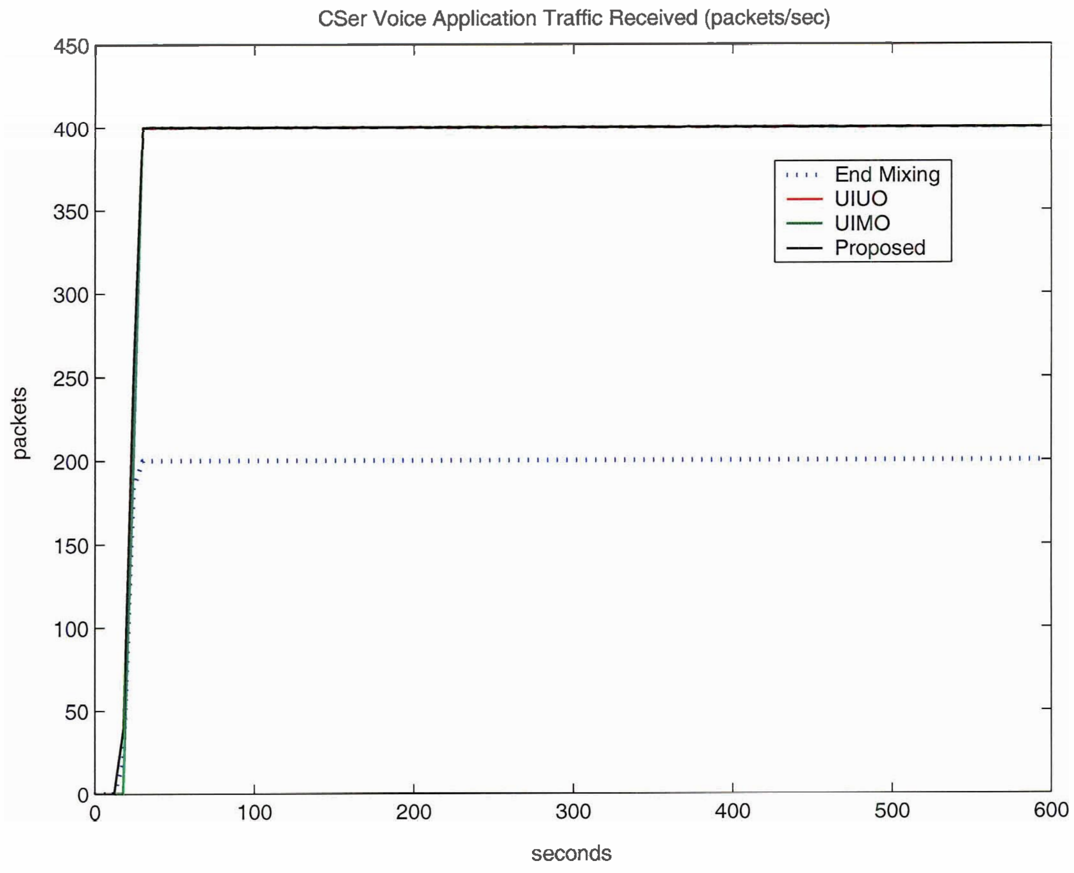


Figure 4.6: CSer Voice Application Traffic Received (packets/sec)

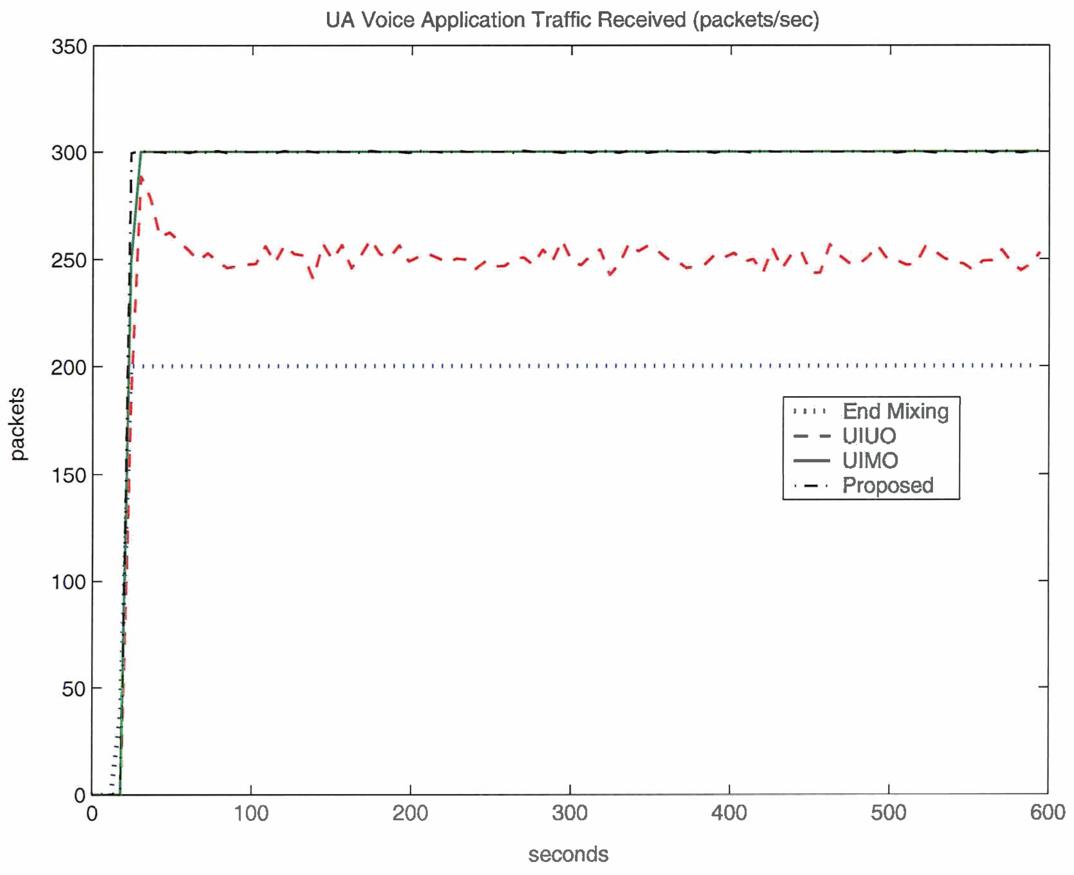


Figure 4.7: UA Voice Application Traffic Received (packets/sec)

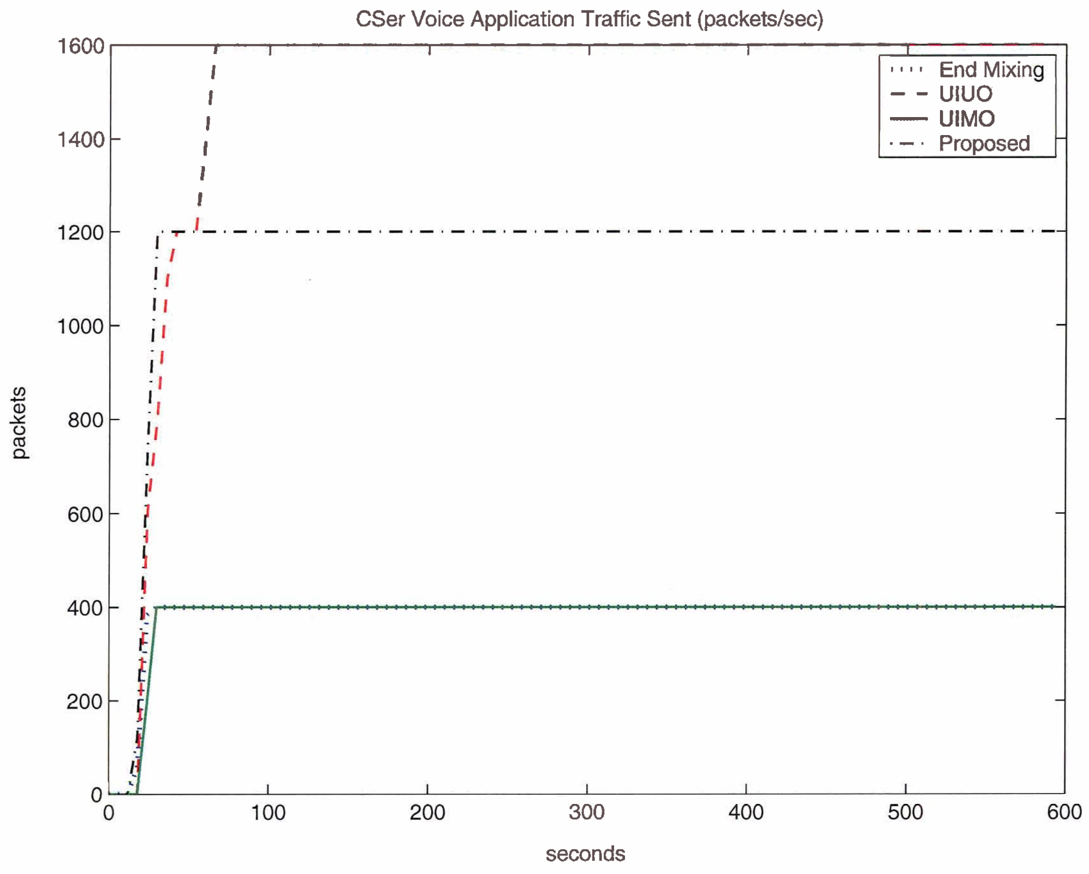


Figure 4.8: CSer Voice Application Traffic Sent (packets/sec)

conferencing model has no or less voice application jitter. This might be because there is no intermediate node in this simple conferencing model, which will not cause route changes or network congestion.

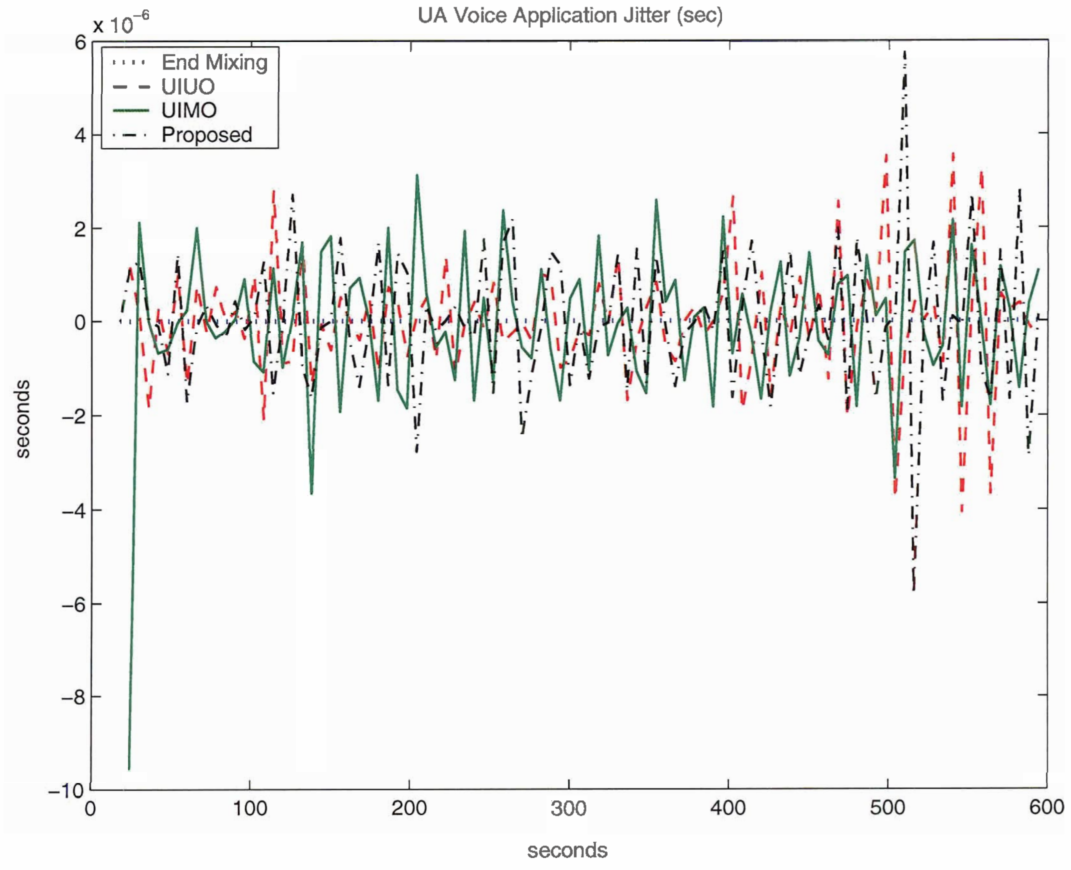


Figure 4.9: UA Voice Application Jitter(sec)

Figure 4.11 and Figure 4.12 show the voice application packet end-to-end delay in seconds for the UA and CSer, respectively. The end-to-end delay is the time that is required for a packet to be transmitted from source to destination. In our simulations, we use the time that the packet reaches the receiver minus the time that the packet was generated in the sender to obtain the end-to-end delay. It can be seen from the results that there is a much longer end-to-end delay in the UIUO conferencing model. This is because it takes a longer time for a packet to transfer from the source to the

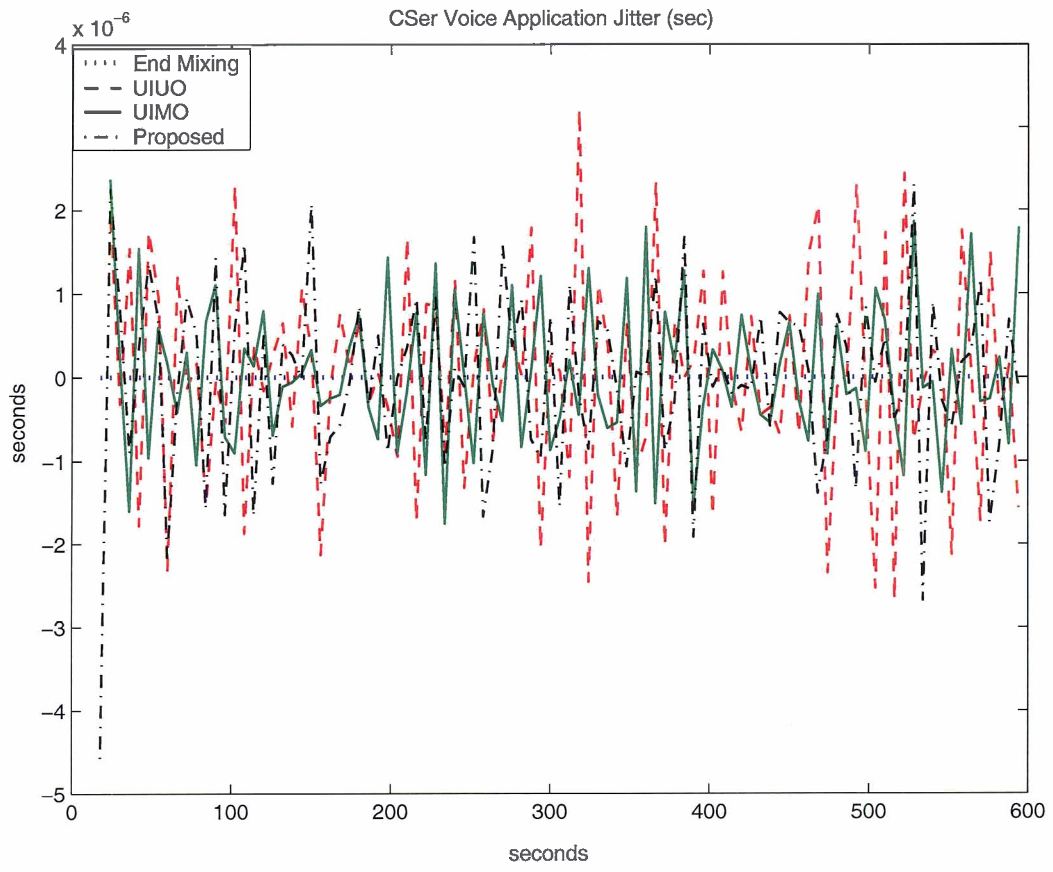


Figure 4.10: CSer Voice Application Jitter(sec)

destination in a congested network.

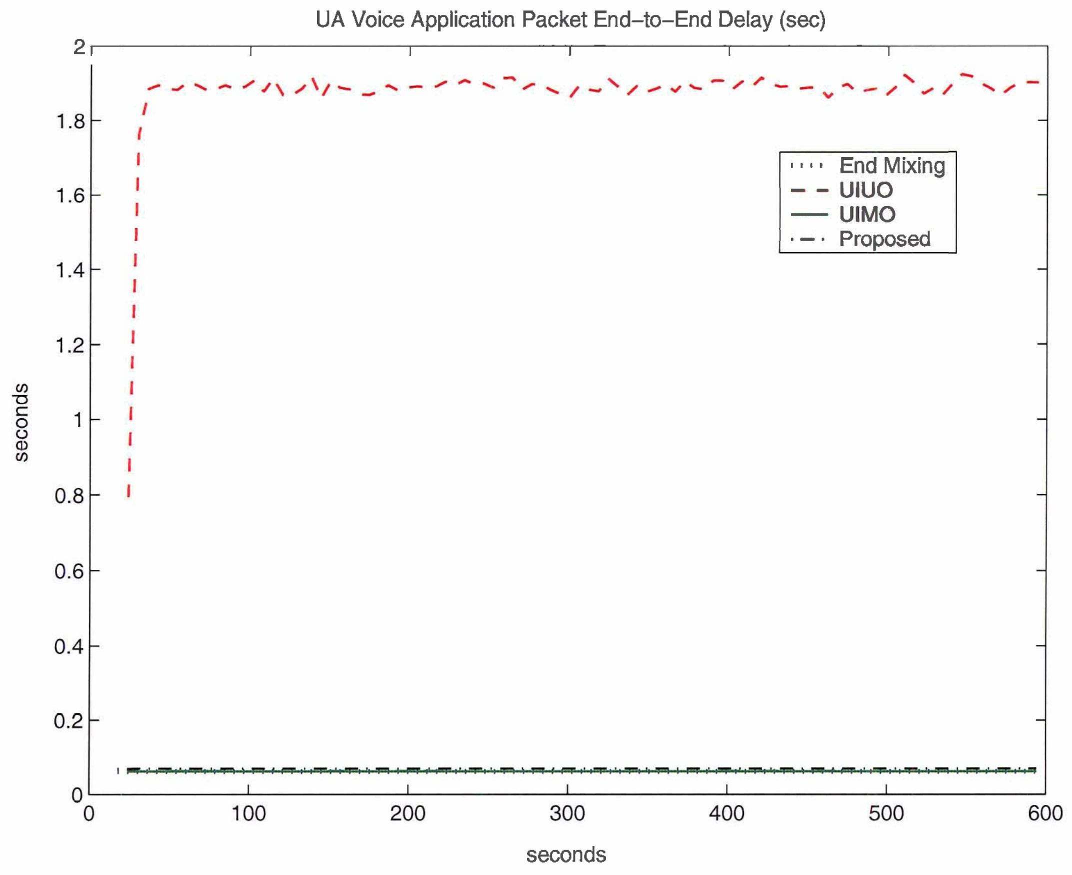


Figure 4.11: UA Voice Application Packet End-to-End Delay

Figure 4.13 shows the global WLAN throughput statistic. The throughput is the speed at which a computer or network sends or receives data. It has been a measure of the channel capacity of a communications link. In our simulations, the global WLAN throughput statistic refers to the amount of data transfered in a specific time in all the wireless interfaces, including the UAs, the CSers and the routers. It is measured in bits per second. As we can see from the results, the UIUO conferencing model has the highest global WLAN throughput. This is simply because that the CSer in this model duplicates unnecessary copies of media streams. As a result, it occupies more wireless bandwidth resources, which are limited in the wireless environment.

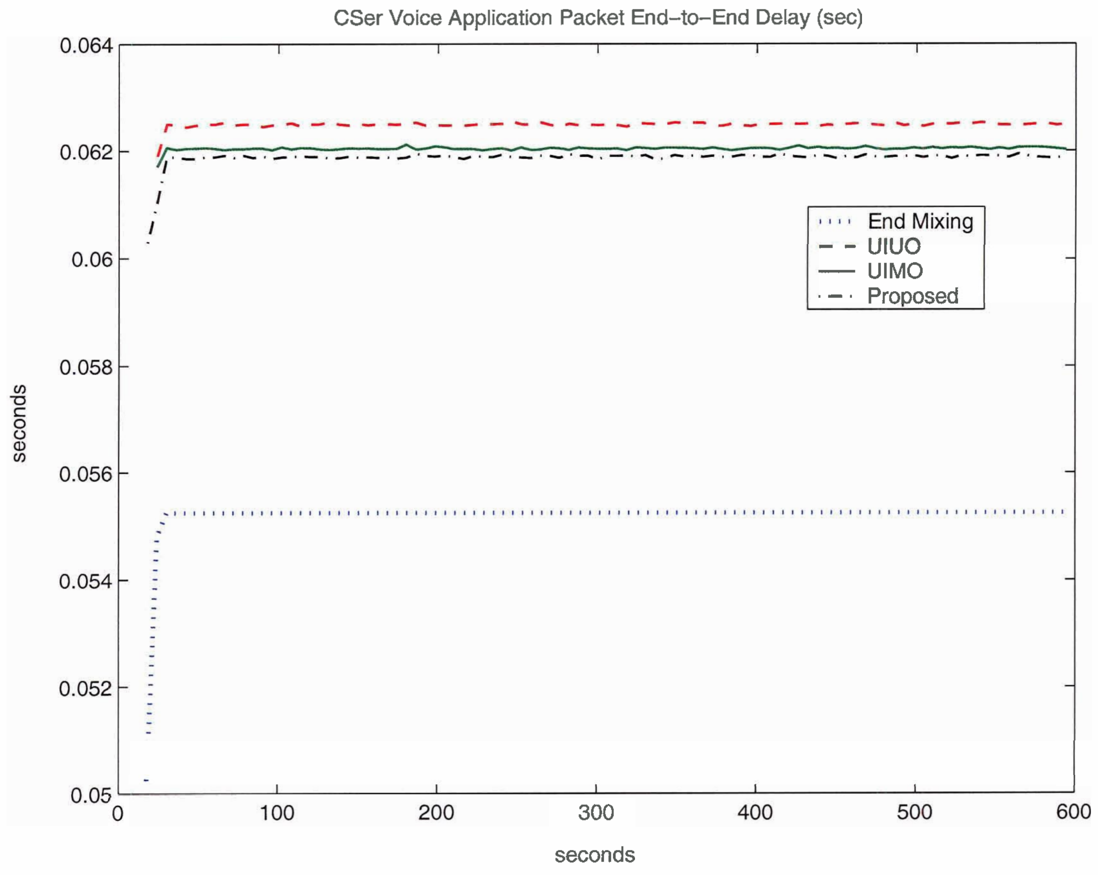


Figure 4.12: CSer Voice Application Packet End-to-End Delay

Furthermore, some packets are dropped due to the network congestion. The global WLAN data dropped statistic in bits per second is shown in Figure 4.14.

Figure 4.15 shows the global WLAN MAC delay statistic in seconds. In our simulation, the WLAN MAC delay is obtained using the current simulation time minuses the arrival time of the packet that is currently handled by the IEEE 802.11 Distributed Coordination Function (DCF).

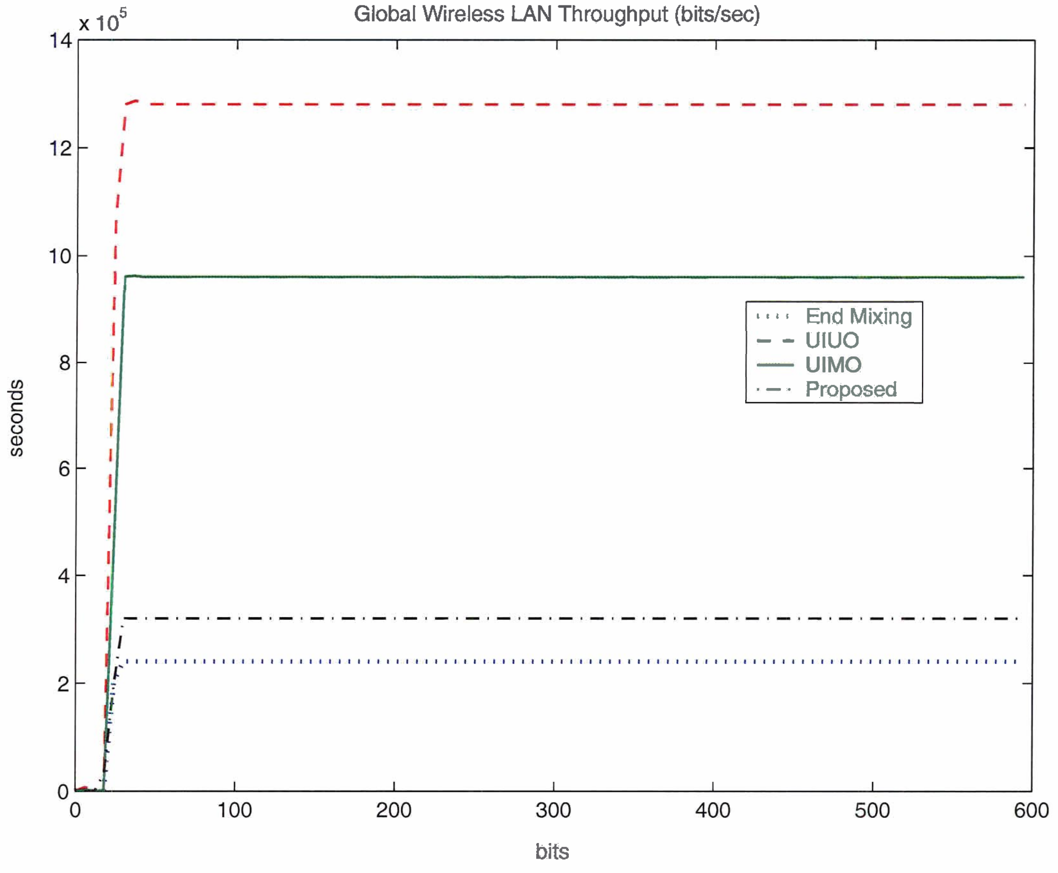


Figure 4.13: Global Wireless LAN Throughput(bits/sec)

Overall, the end mixing conferencing model has the best performance in small number of users. On the other hand, the conferencing server in the unicast receive and multicast send conferencing model duplicates unnecessary copies of media streams to

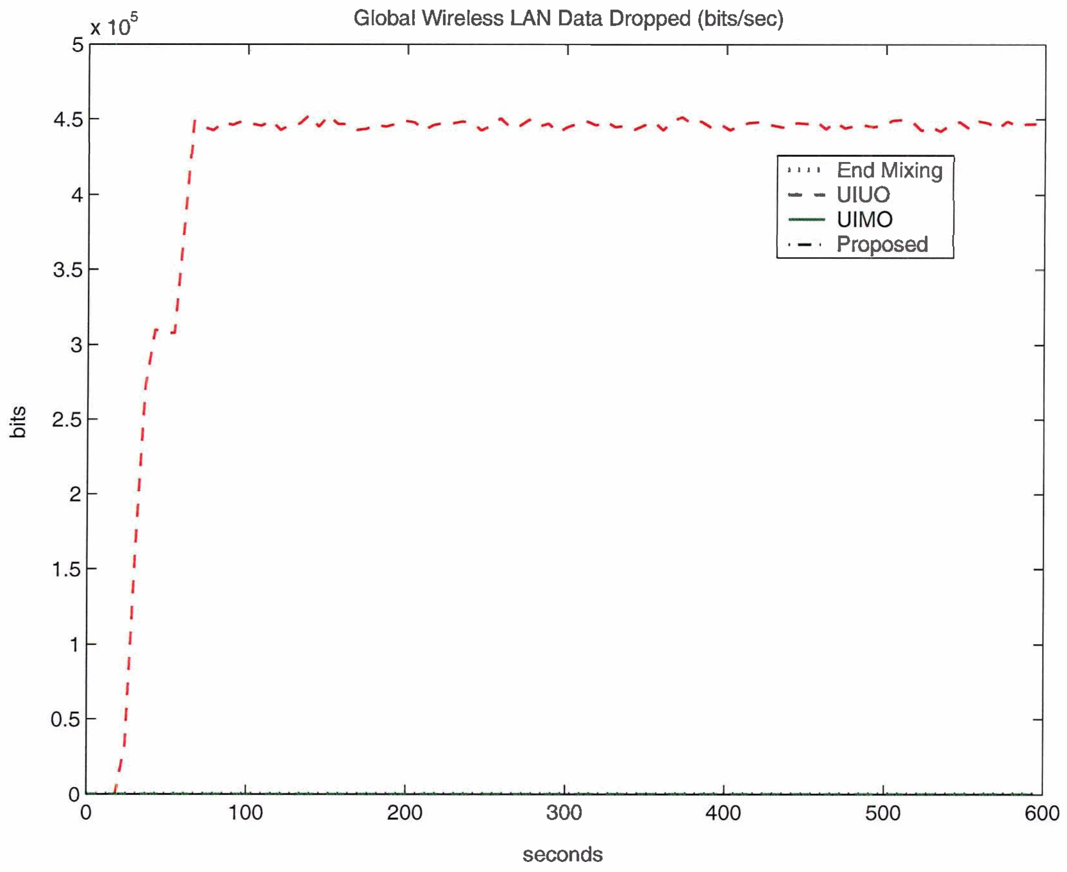


Figure 4.14: Global Wireless LAN Data Dropped(bits/sec)

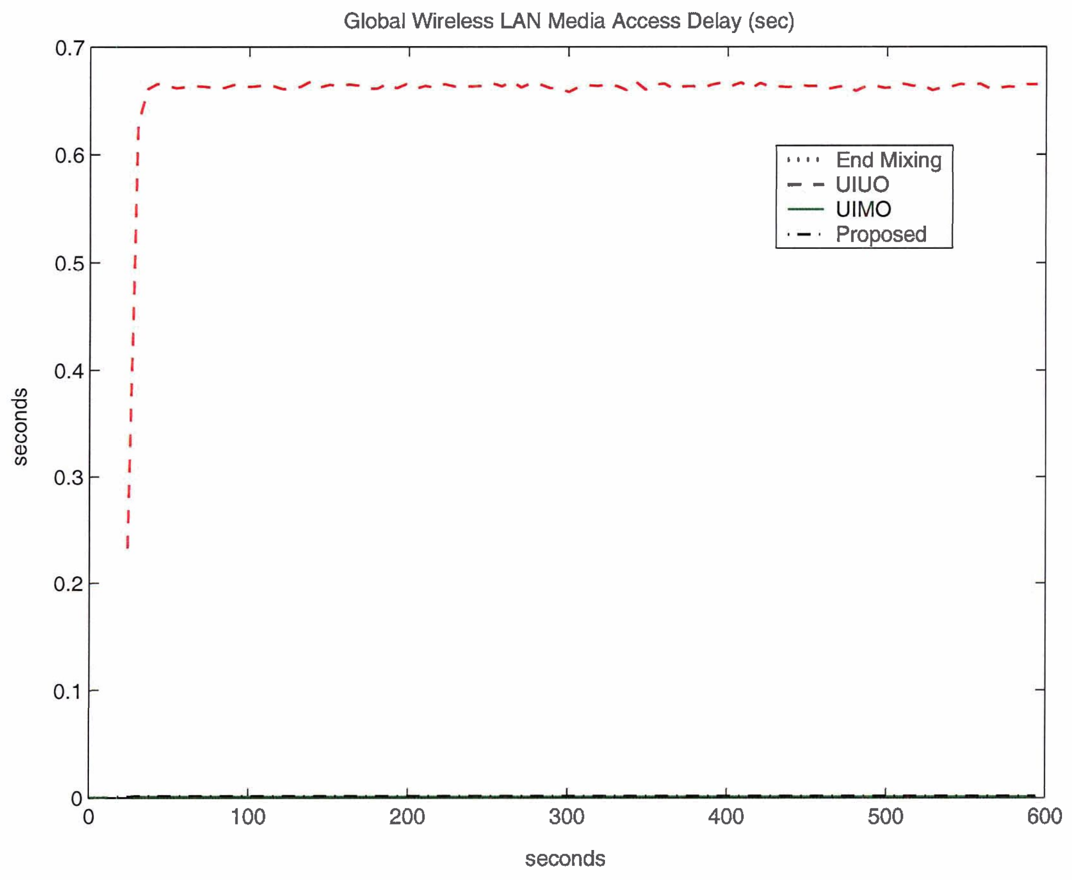


Figure 4.15: Global Wireless LAN Media Access Delay(sec)

occupy more wireless bandwidth. This model tends to cause network congestion and degrade the QoS performance by having longer end-to-end delay and more packets dropped. So, we will only consider to compare our proposed model with the UIMO conferencing model in the next scenario.

4.3 Scenario Two

In scenario two, we selected a G.729 (silence) voice codec to demonstrate the bandwidth efficiency obtained by utilizing a silence suppression voice codec. We simulated this scenario in both the UIMO conferencing model as well as our proposed conferencing model. The initial number of users is ten, then it increases to twenty, then forty and finally reaches eighty. The statistics that are listed in Table 4.1 for both conferencing models were collected and selected statistics were analyzed in this section.

Figure 4.16 and Figure 4.18 show the voice application jitter for the UA and CSer in UIMO conferencing model. Figure 4.17 and Figure 4.19 show the voice application jitter for the UA and CSer in our proposed conferencing model. The results show that the jitter for both the UA and CSer in our proposed model are less that those in UIMO conferencing model.

Figure 4.20 and Figure 4.21 show the amount of voice application traffic sent by the UA in UIMO conferencing model and our proposed conferencing model. Compared with Figure 4.5, utilizing a silence suppression voice codec has a higher bandwidth efficiency. It saves a significant amount of bandwidth, more than 50% in our case.

Figure 4.22 and Figure 4.23 show the amount of voice application traffic received by the UA in UIMO conferencing model and our proposed conferencing model. It can be indicated from the results that as the number of users increases, a significant number of packets are lost before they reach the UA in UIMO conferencing mode. In this case, the voice quality will not be acceptable. On the other hand, our conferencing model performance is quite stable even with 80 users.

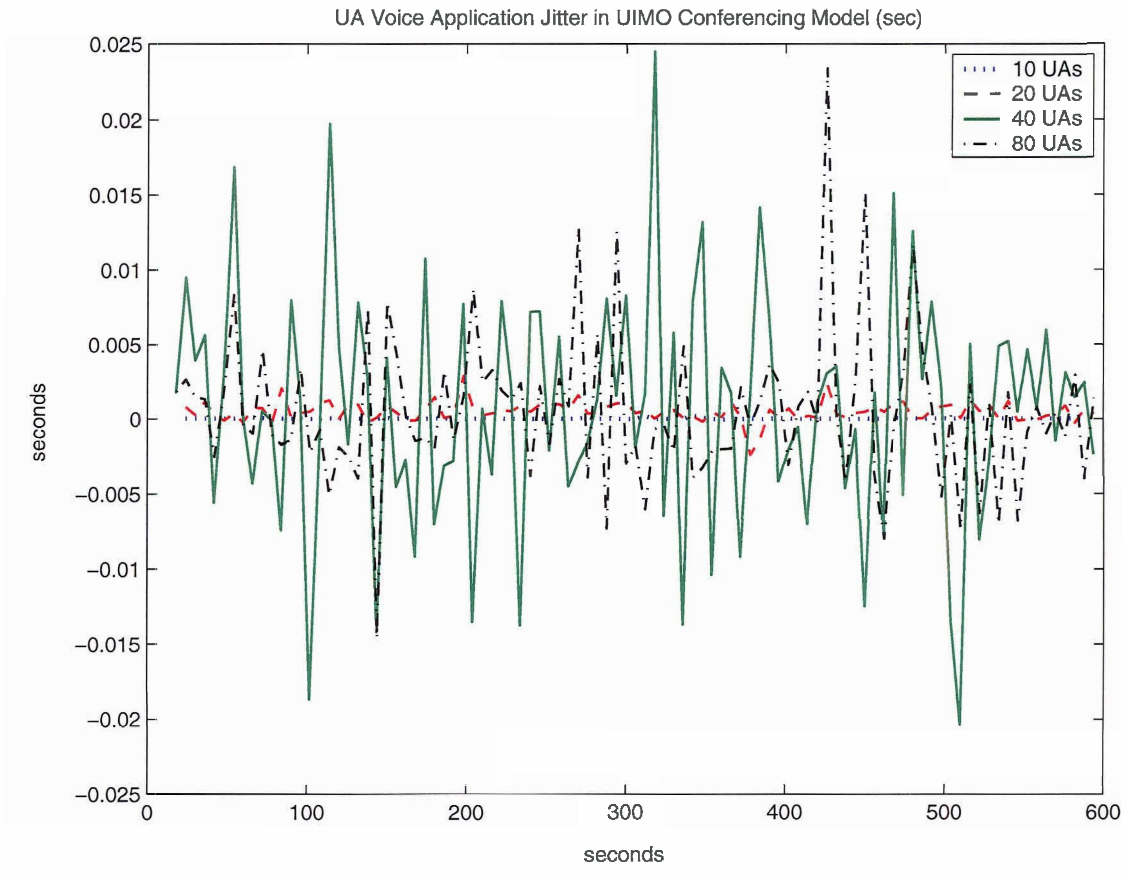


Figure 4.16: UA Voice Application Jitter in UIMO Conferencing Model (sec)

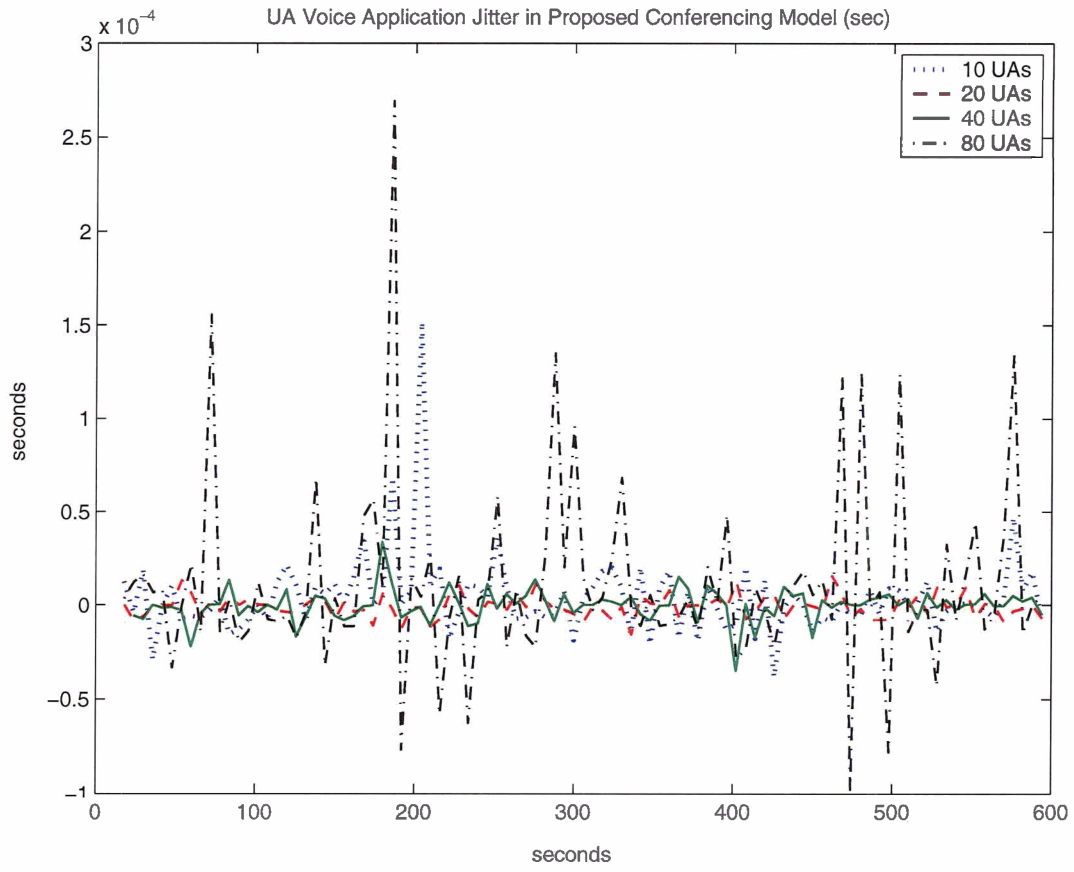


Figure 4.17: UA Voice Application Jitter in Proposed Conferencing Model (sec)

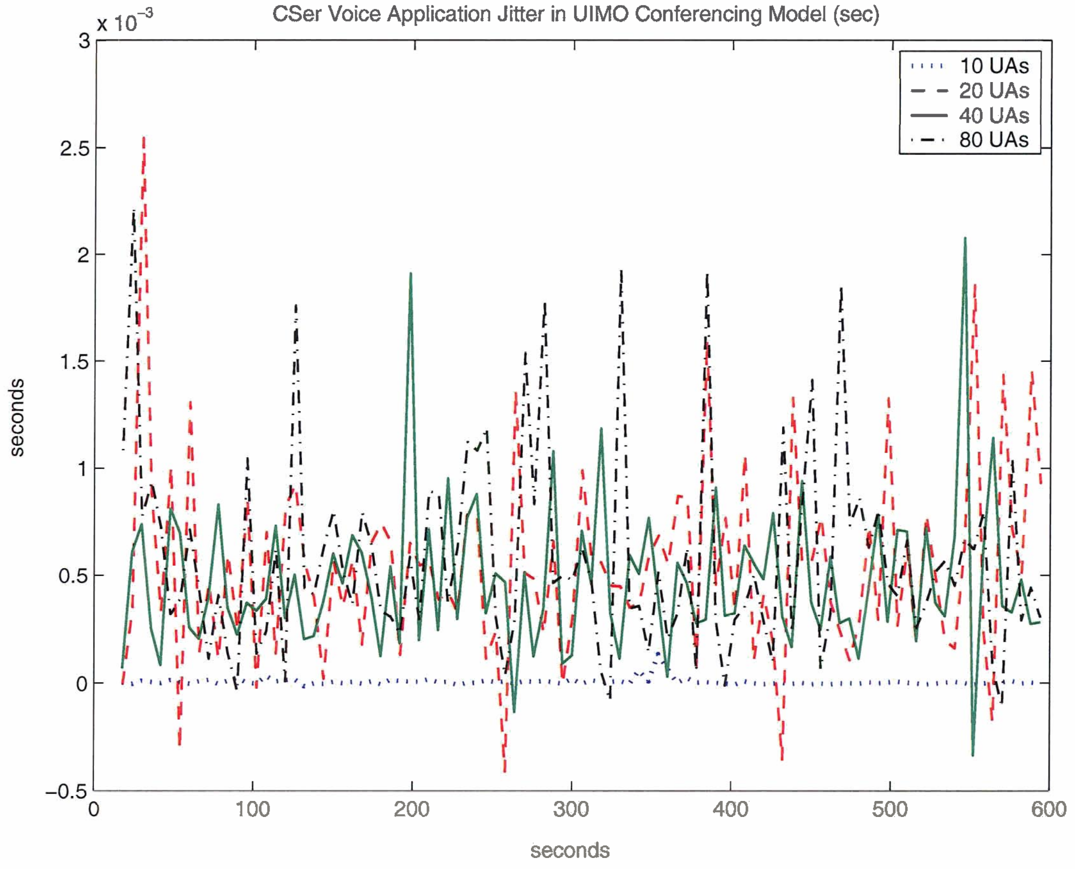


Figure 4.18: CSer Voice Application Jitter in UIMO Conferencing Model (sec)

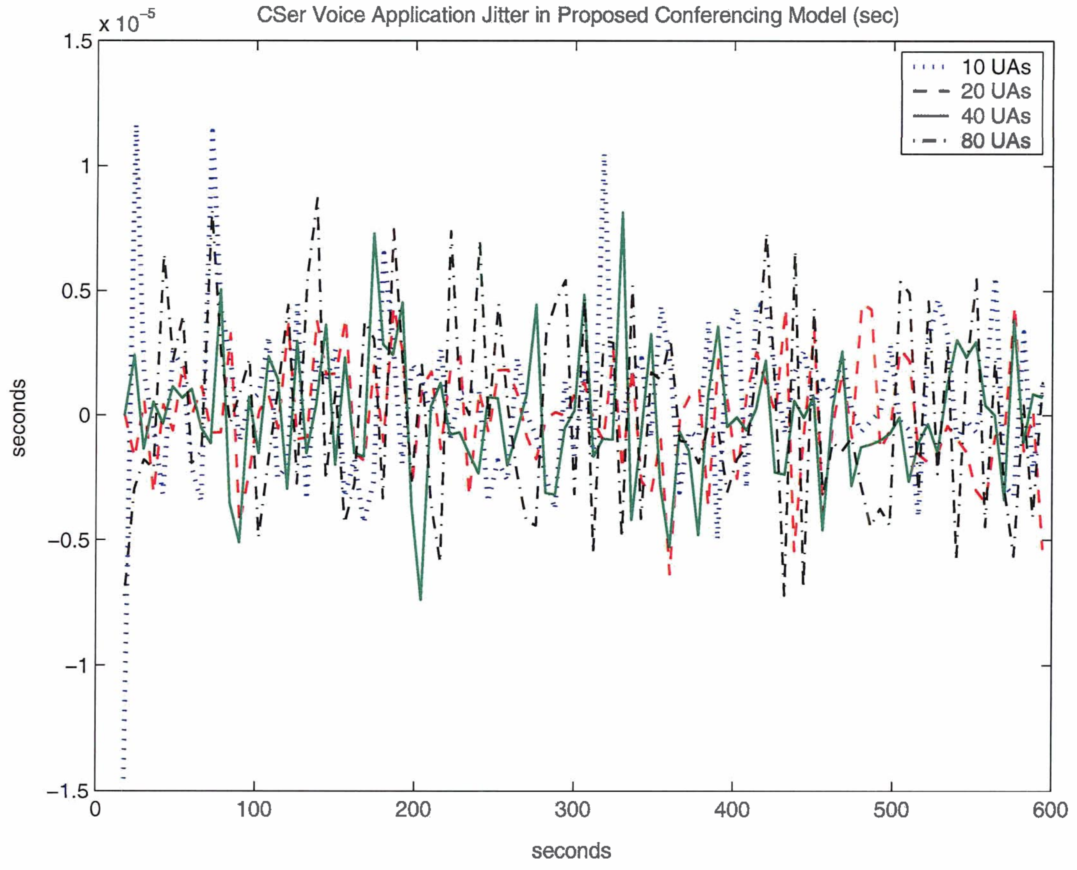


Figure 4.19: CSer Voice Application Jitter in Proposed Conferencing Model (sec)

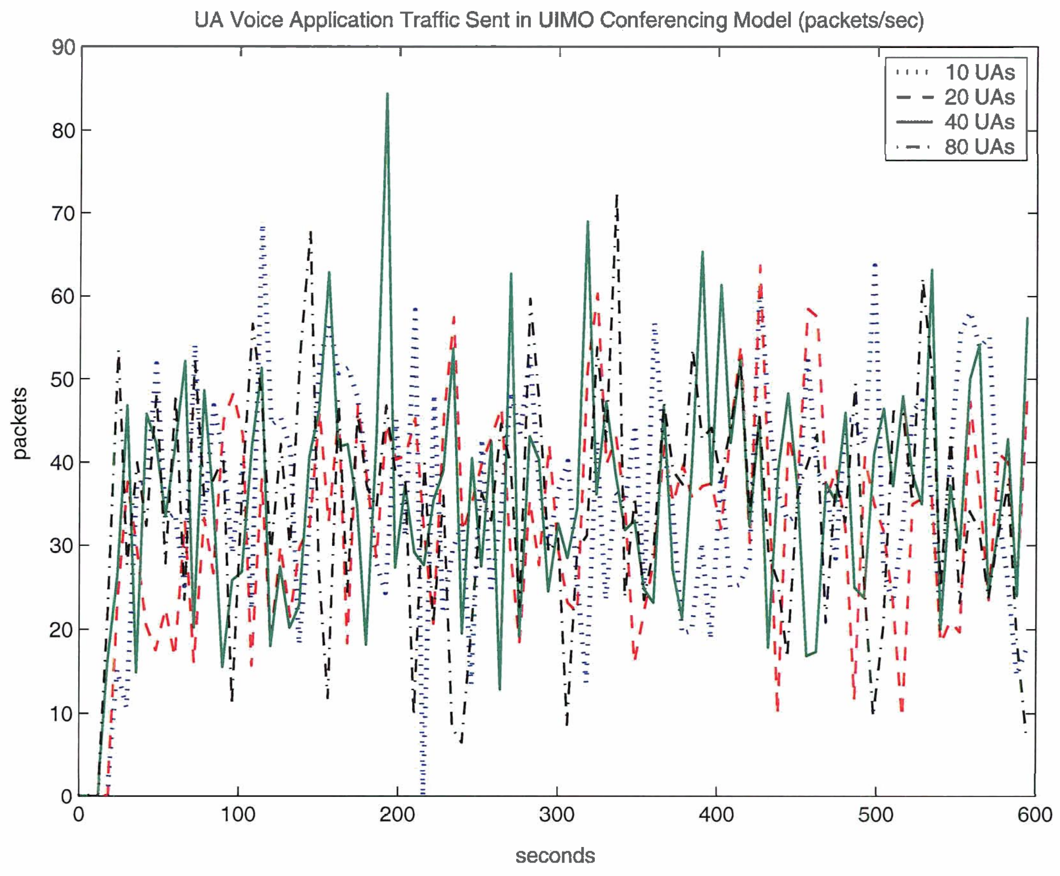


Figure 4.20: UA Voice Application Traffic Sent in UIMO Conferencing Model (packets/sec)

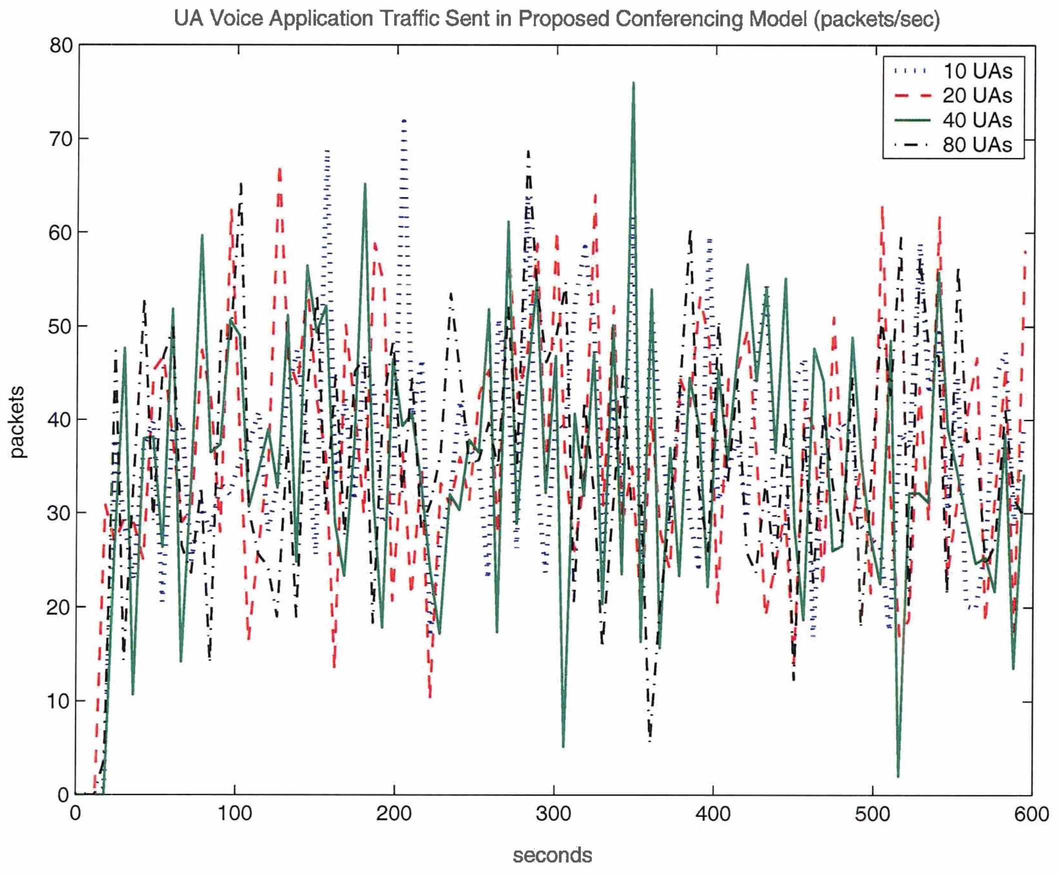


Figure 4.21: UA Voice Application Traffic Sent in Proposed Conferencing Model (packets/sec)

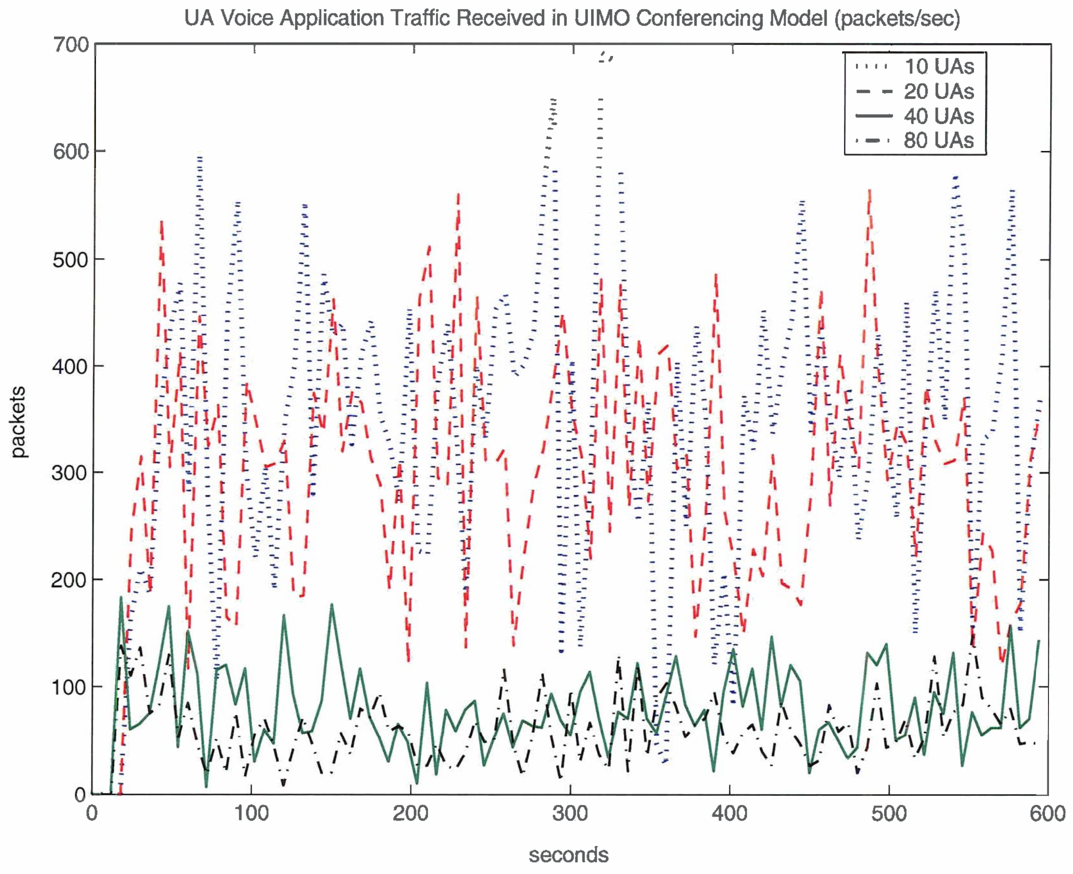


Figure 4.22: UA Voice Application Traffic Received in UIMO Conferencing Model (packets/sec)

Figure 4.24 and Figure 4.25 show the voice application packet end-to-end delay for the UA in the UIMO conferencing model and our proposed conferencing model. Figure 4.26 and Figure 4.27 illustrate the results of voice application packet end-to-end delay in seconds for the CSer for both conferencing models respectively. As shown in Figure 4.24, the voice application packet end-to-end delay for the UA is about 0.6 second when the number of users increase to forty, which is not acceptable for real-time voice communication.

Figure 4.28 and Figure 4.29 show the global WLAN data dropped statistics in bits

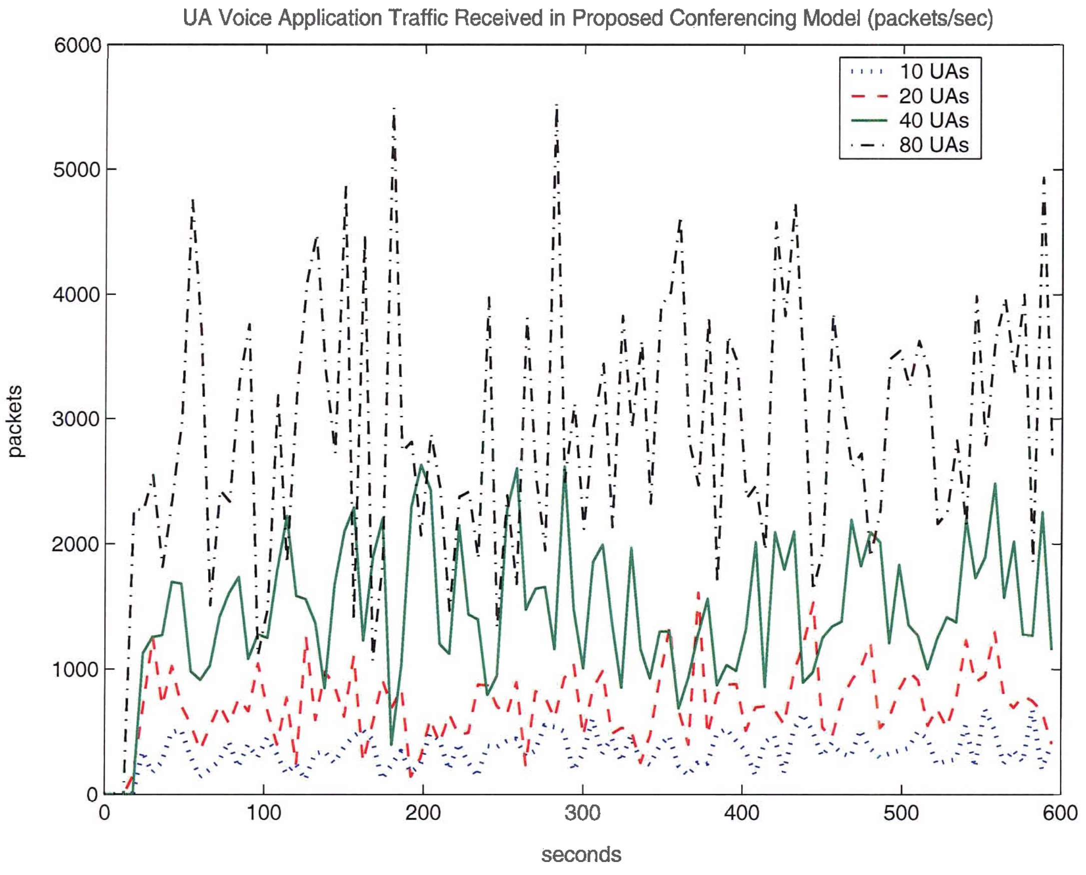


Figure 4.23: UA Voice Application Traffic Received in Proposed Conferencing Model (packets/sec)

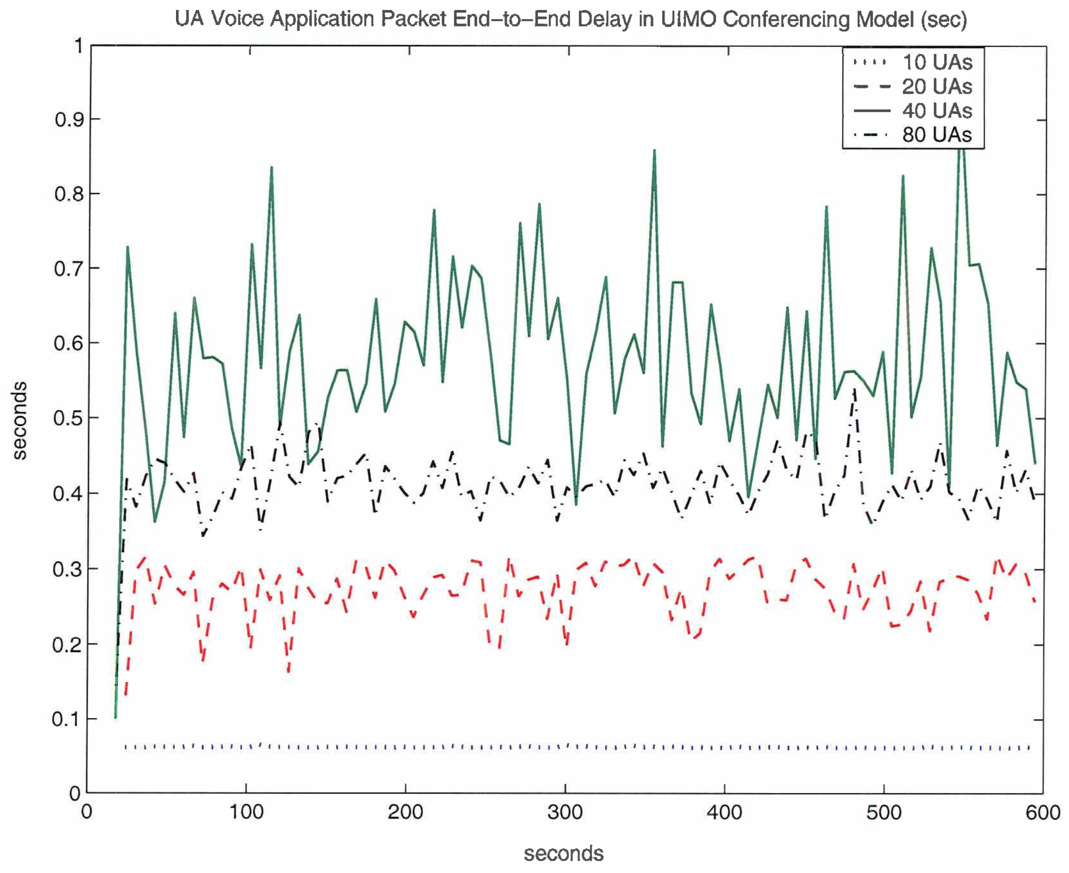


Figure 4.24: UA Voice Application Packet End-to-End Delay in UIMO Conferencing Model (sec)

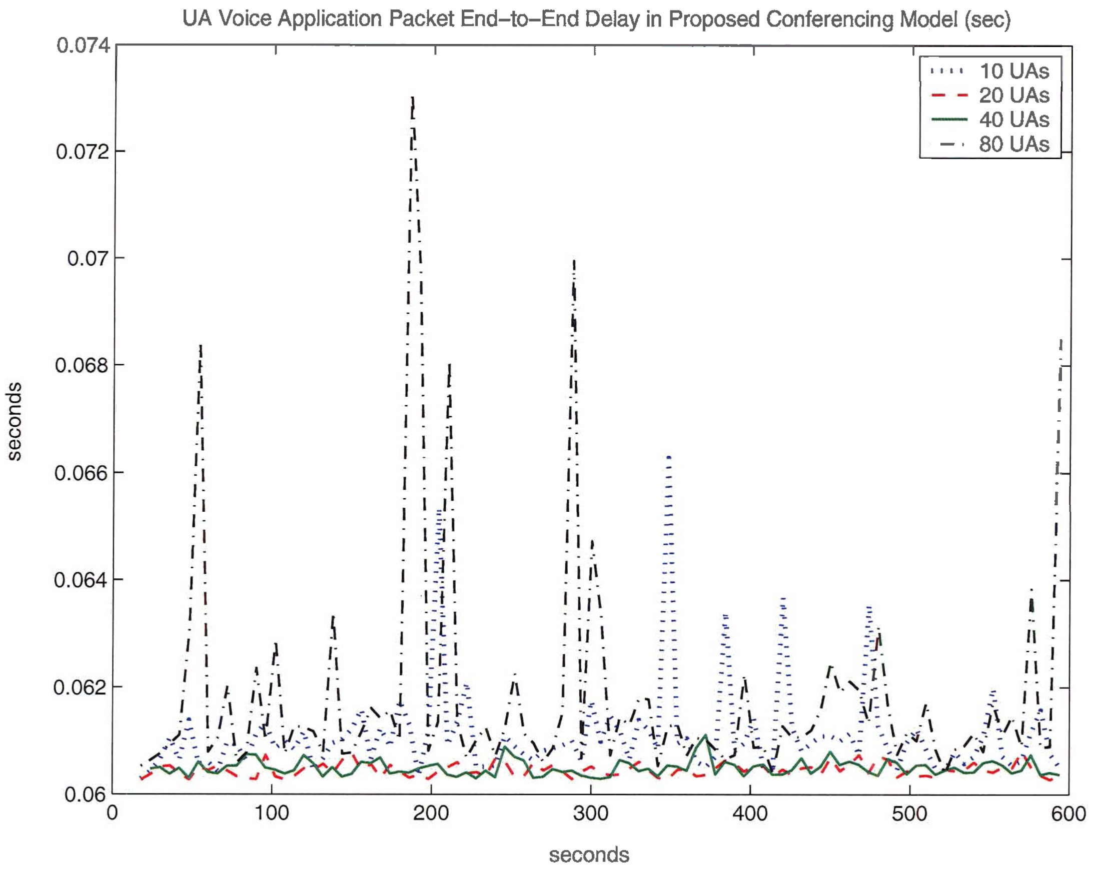


Figure 4.25: UA Voice Application Packet End-to-End Delay in Proposed Conferencing Model (sec)

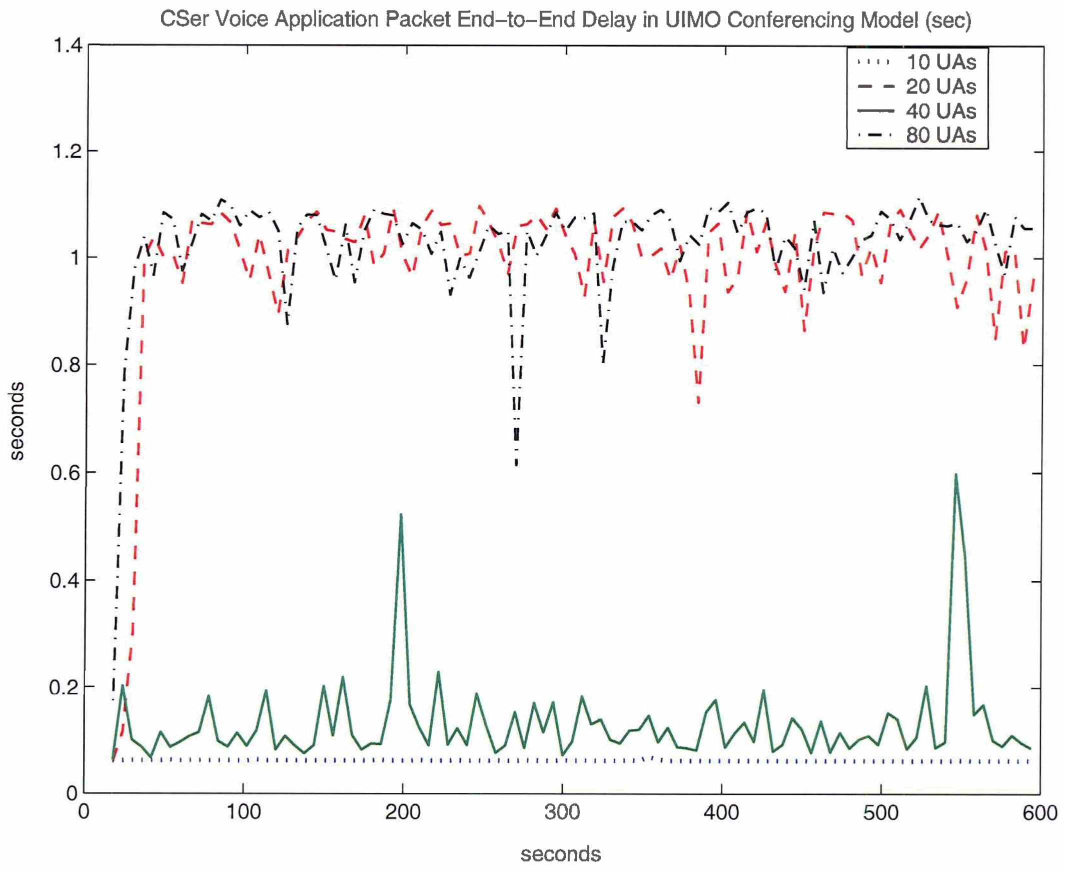


Figure 4.26: CSer Voice Application Packet End-to-End Delay in UIMO Conferencing Model (sec)

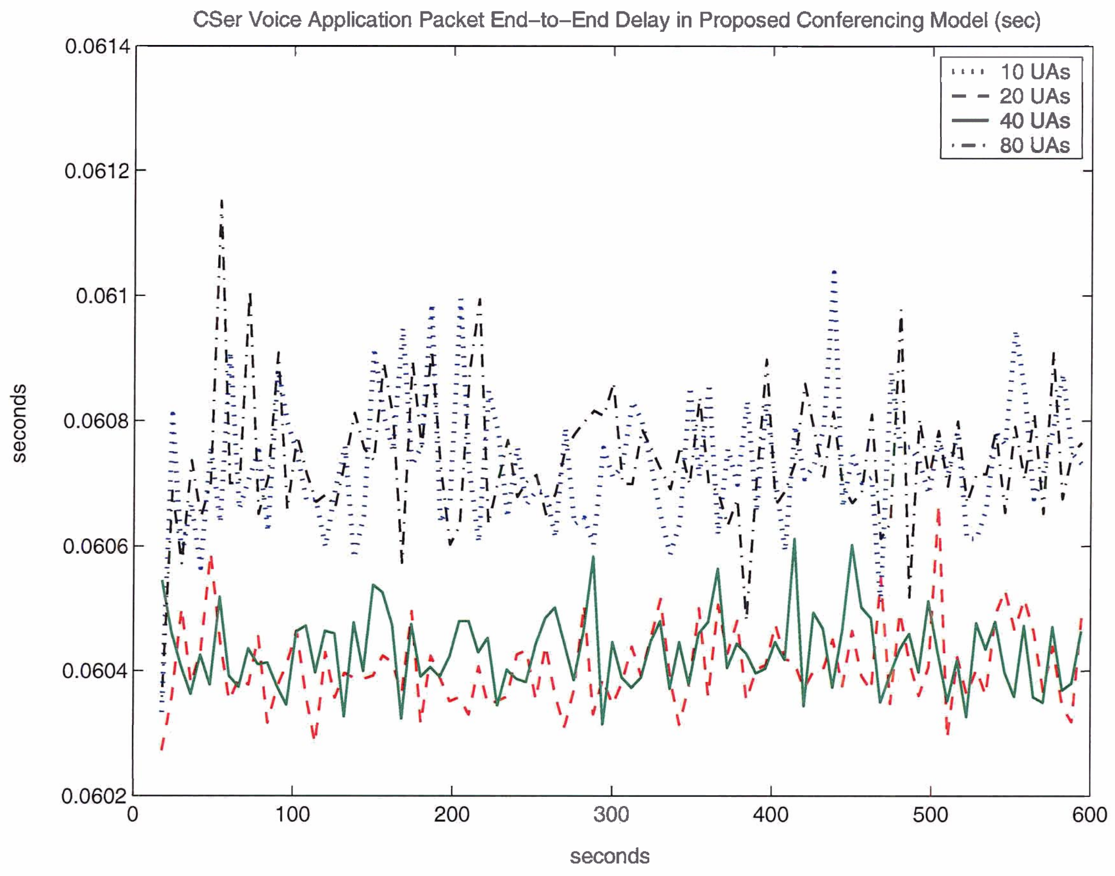


Figure 4.27: CSer Voice Application Packet End-to-End Delay in Proposed Conferencing Model (sec)

per second for UIMO conferencing model and our proposed conferencing model. The WLAN MAC delay statistics for UIMO conferencing model and our proposed conferencing model are shown in Figure 4.30 and Figure 4.31 respectively.

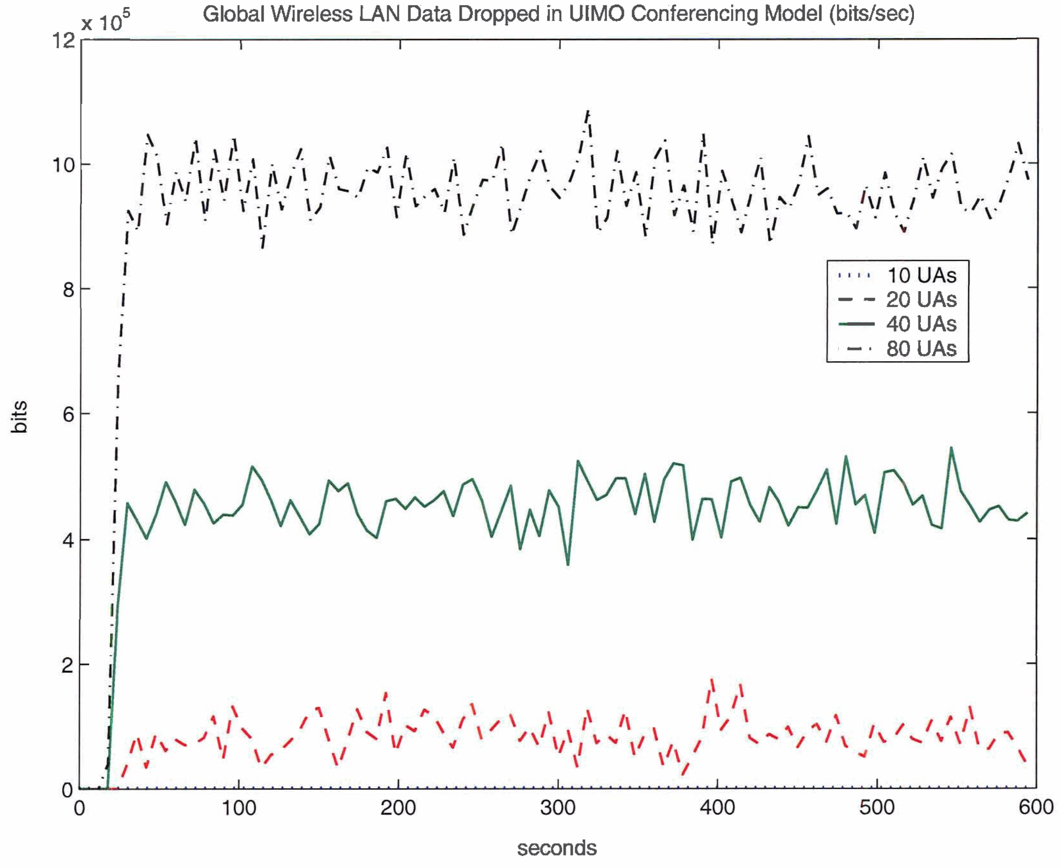


Figure 4.28: Global Wireless LAN Data Dropped in UIMO Conferencing Model (bits/sec)

Overall, our proposed conferencing model has similar QoS performance to the UIMO conferencing model with a small number of users, such as ten. However, the UIMO conferencing model is incapable of providing guaranteed QoS as the number of users increases. On the other hand, our proposed model provides more stable services even with 80 users. In addition, the conferencing scale can be increased simply by adding

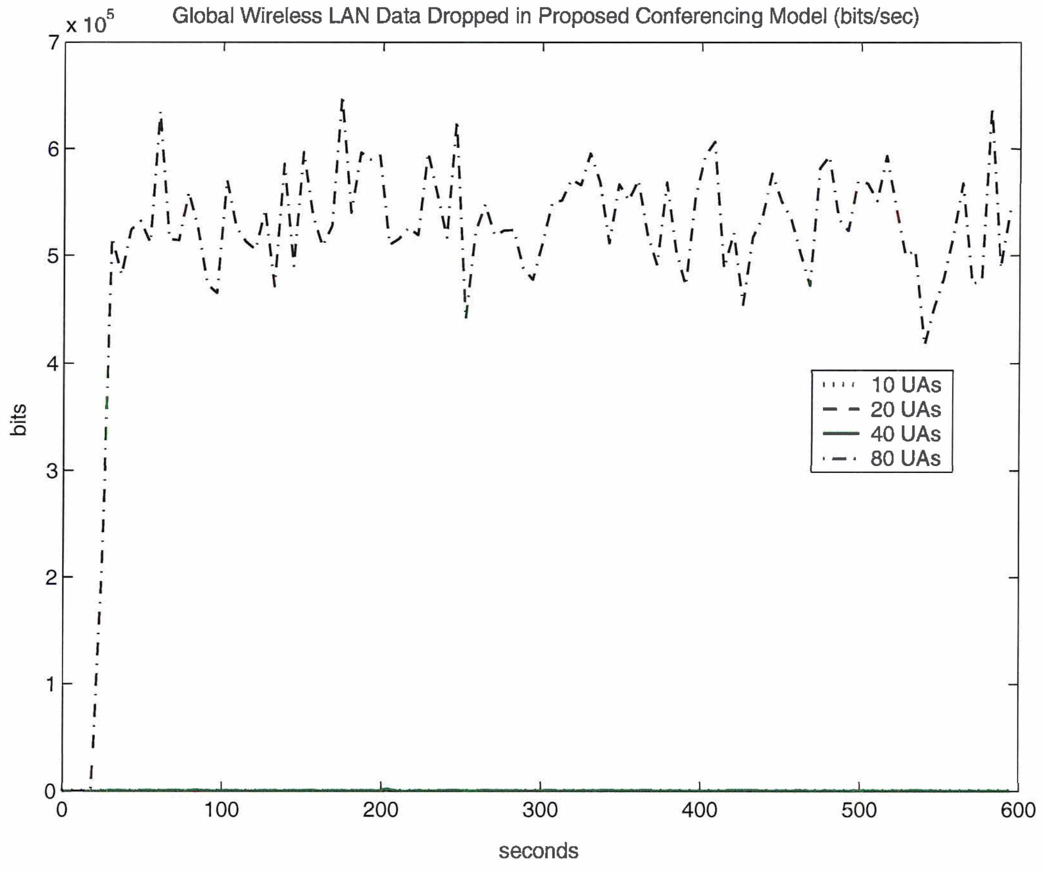


Figure 4.29: Global Wireless LAN Data Dropped in Proposed Conferencing Model (bits/sec)

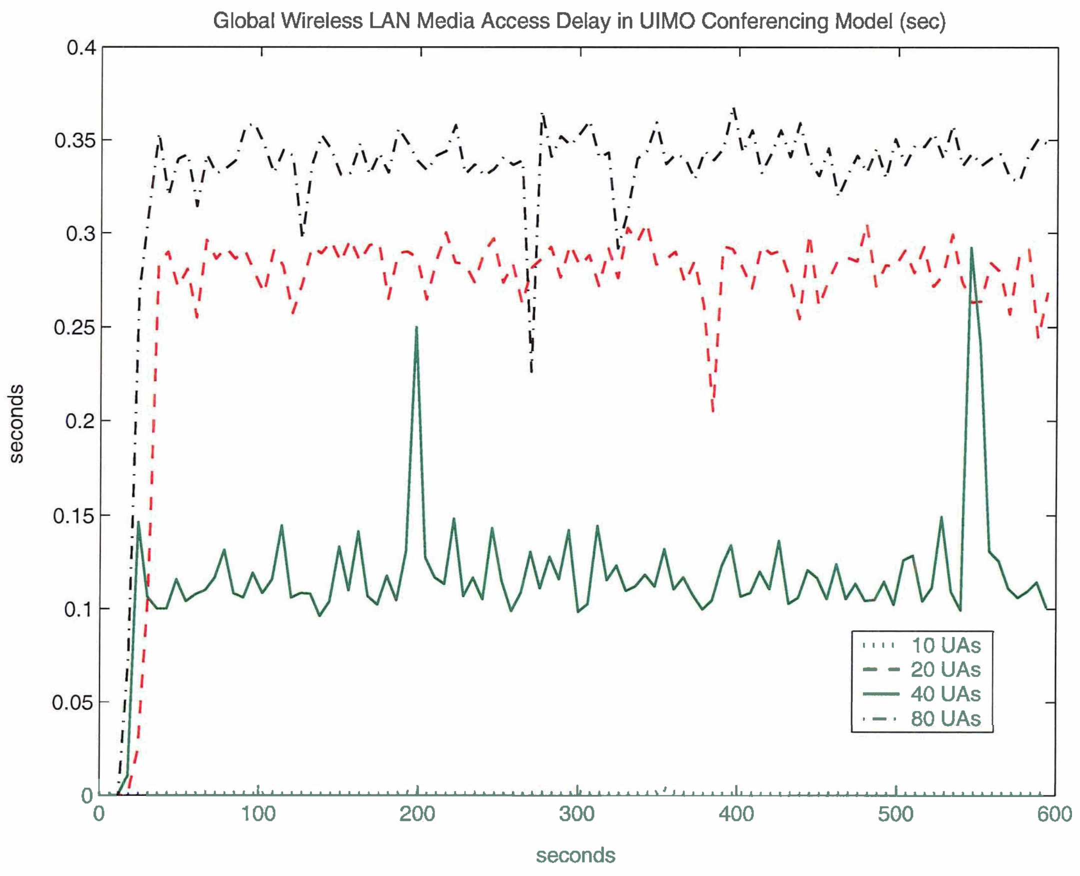


Figure 4.30: Global Wireless LAN Media Access Delay in UIMO Conferencing Model (sec)

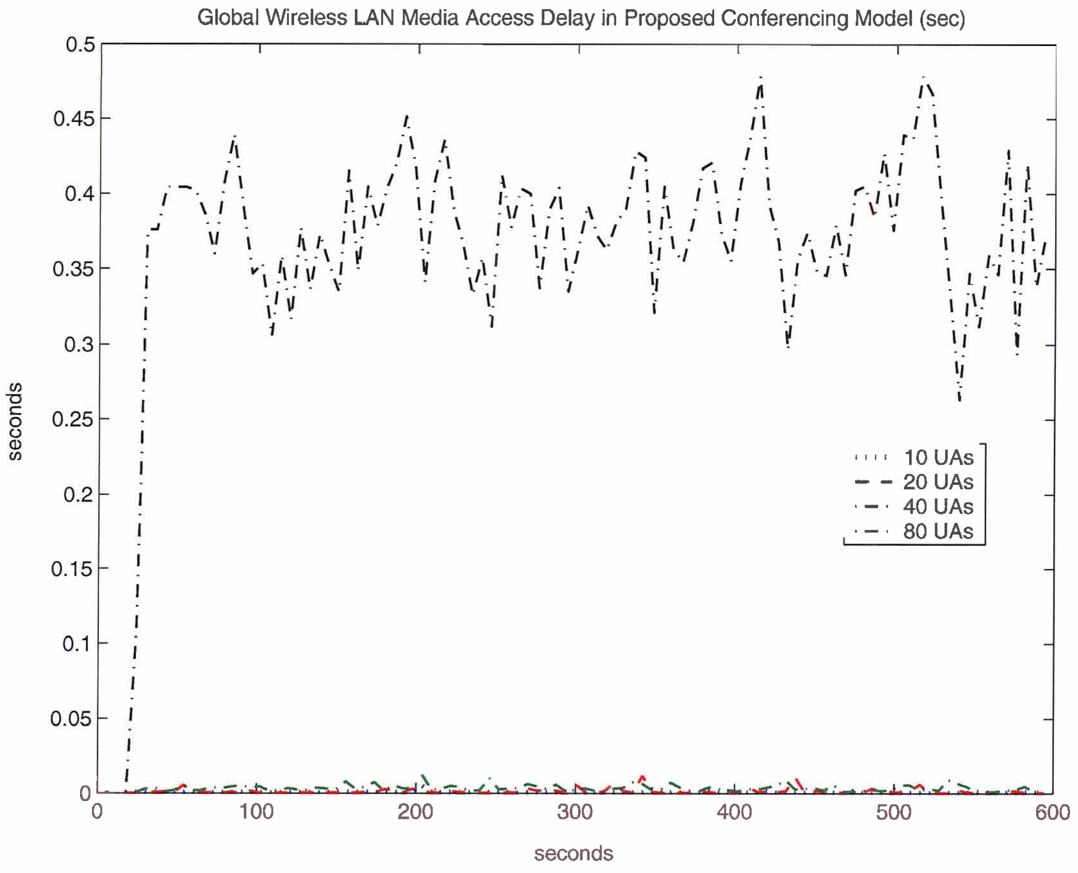


Figure 4.31: Global Wireless LAN Media Access Delay in Proposed Conferencing Model (sec)

additional conferencing servers, which is very flexible. The drawback is that, it requires additional conferencing control such as maintaining a list of active conferencing servers as well as the participating users.

4.4 Scenario Three

The purpose of this scenario is to compare the voice conferencing performance for static users vs. mobile users. In this scenario, we selected the G.729 (silence) voice codec and defined a small scale conference as ten users. We assigned each UA a trajectory, which specifies the times and locations that a mobile node will pass through as the simulation progresses. The simulated conferencing duration is ten minutes.

Figure 4.32 shows the voice application jitter statistic in seconds for both static UAs and mobile UAs.

Figure 4.33 shows the voice application traffic sent in packets per second for both static UAs and mobile UAs. The voice application traffic received statistics in packets per second for both static UAs and mobile UAs are shown in Figure 4.34. The results show that, as long as the UA has the same traffic parameters, it sends out similar traffic distribution, no matter if the UA is static or mobile.

Figure 4.35 shows the voice application end-to-end delay in seconds for both static UAs and mobile UAs. Figure 4.36 shows the global WLAN data dropped in bits per second for both static UAs and mobile UAs. Figure 4.37 shows the global WLAN delay in seconds for both static UAs and mobile UAs. The results indicates that there is a longer delay and more packets are dropped when the UA is in a mobile environment.

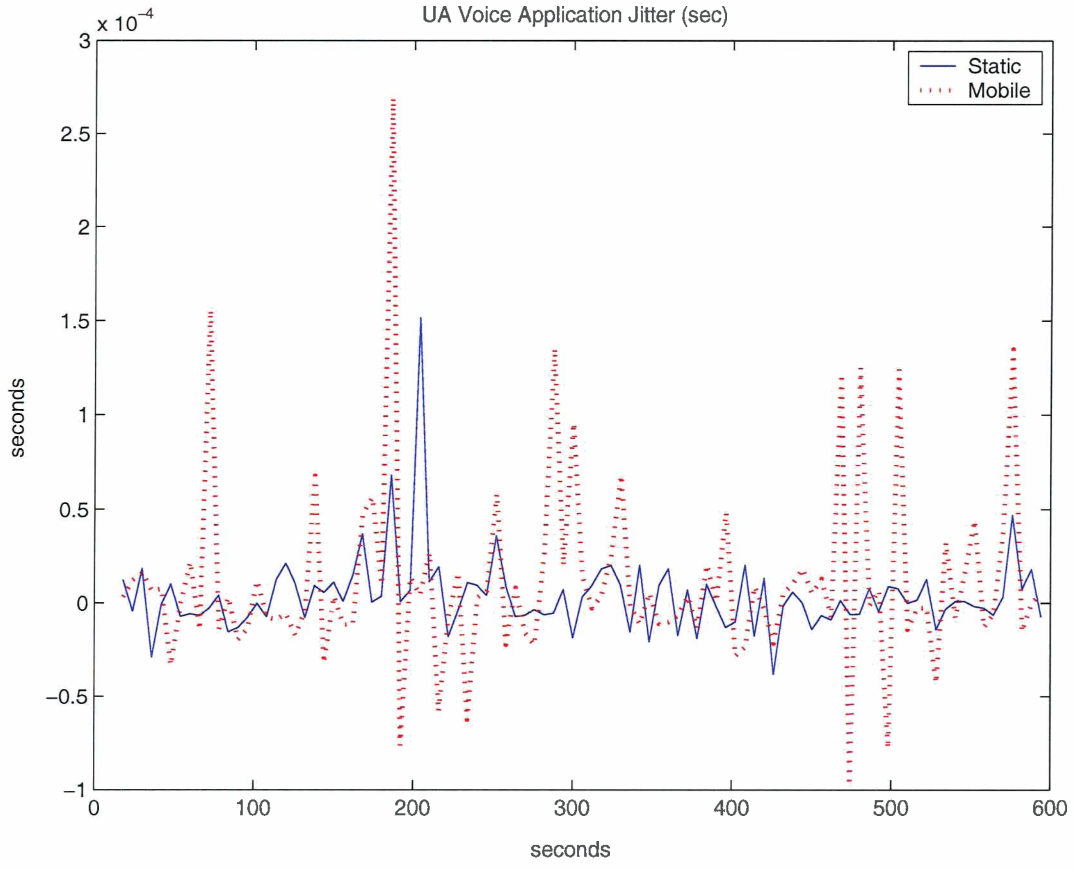


Figure 4.32: UA Voice Application Jitter (sec)

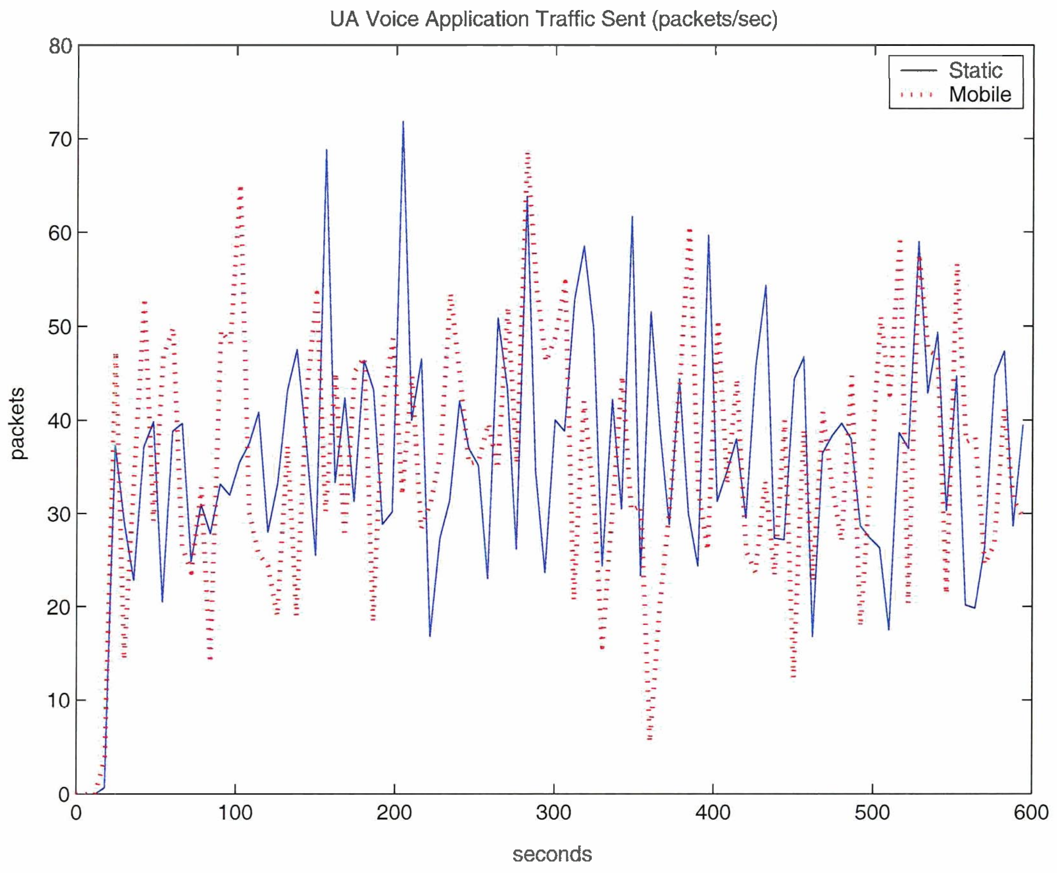


Figure 4.33: UA Voice Application Traffic Sent (packets/sec)

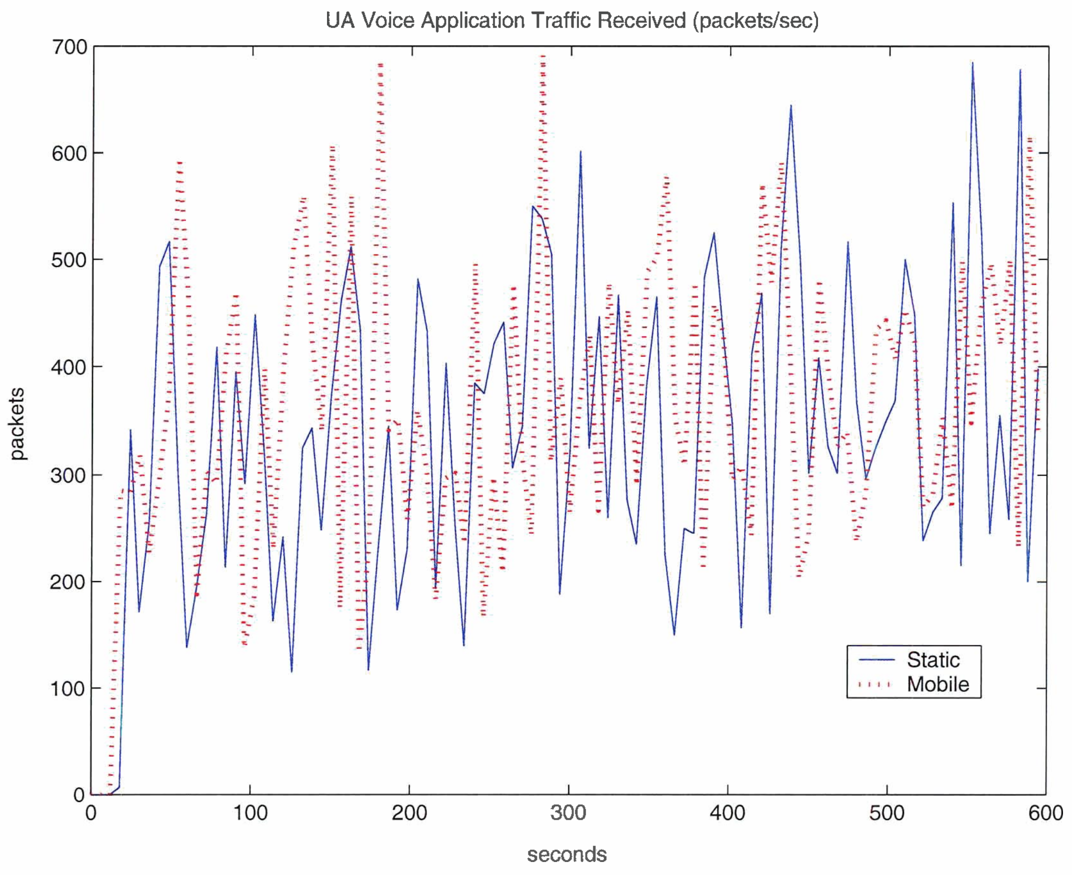


Figure 4.34: UA Voice Application Traffic Received (packets/sec)

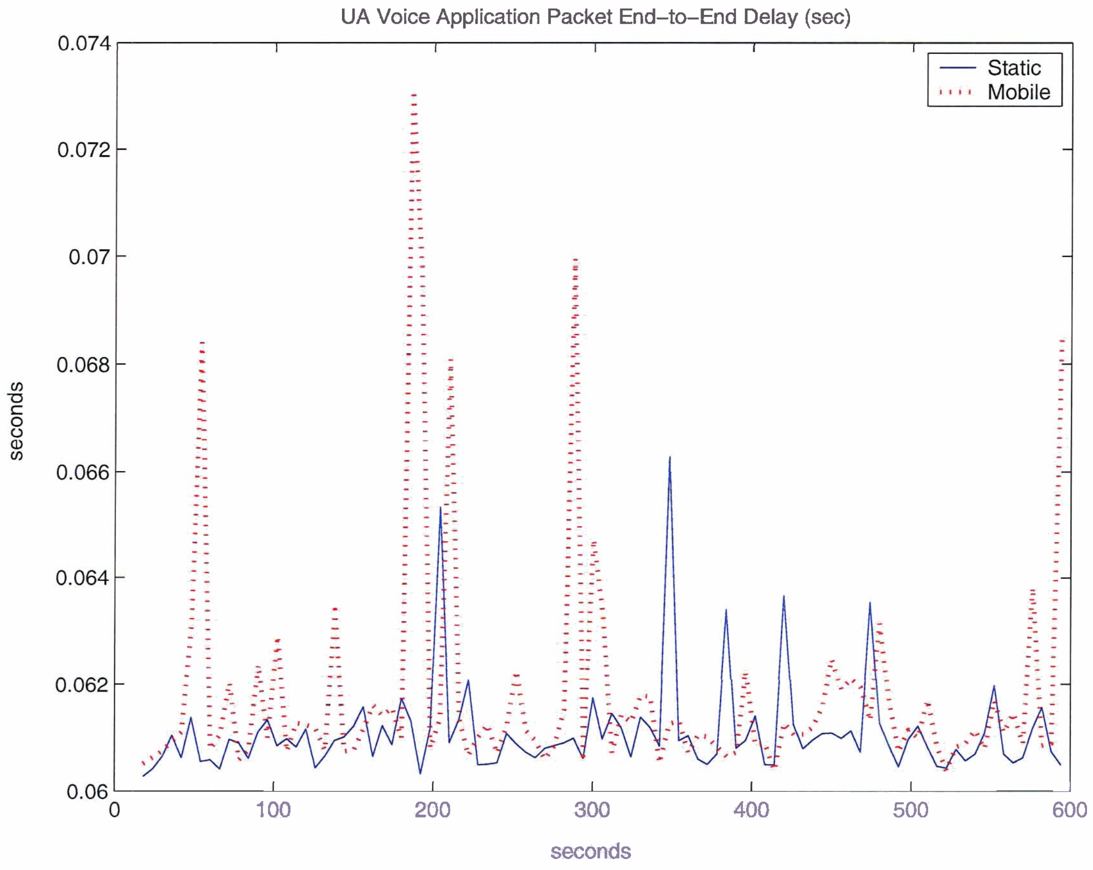


Figure 4.35: UA Voice Application Packet End-to-End Delay (sec)

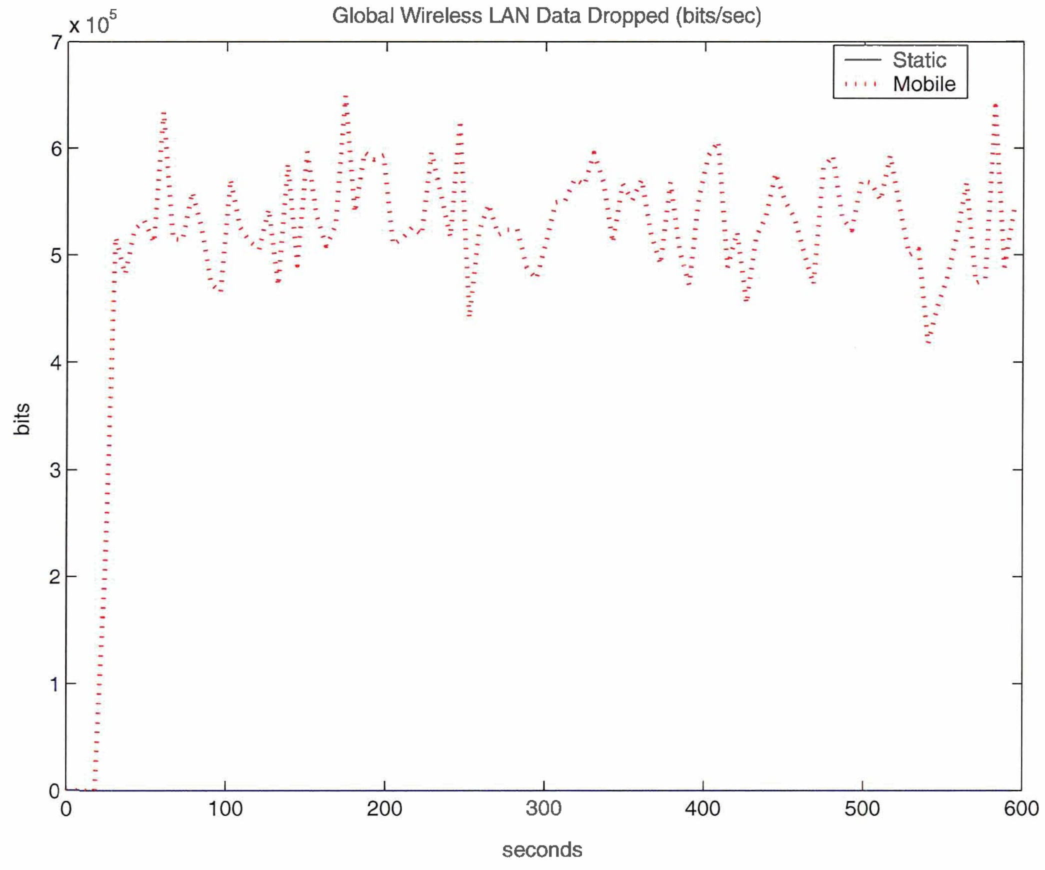


Figure 4.36: Global Wireless LAN Data Dropped (bits/sec)

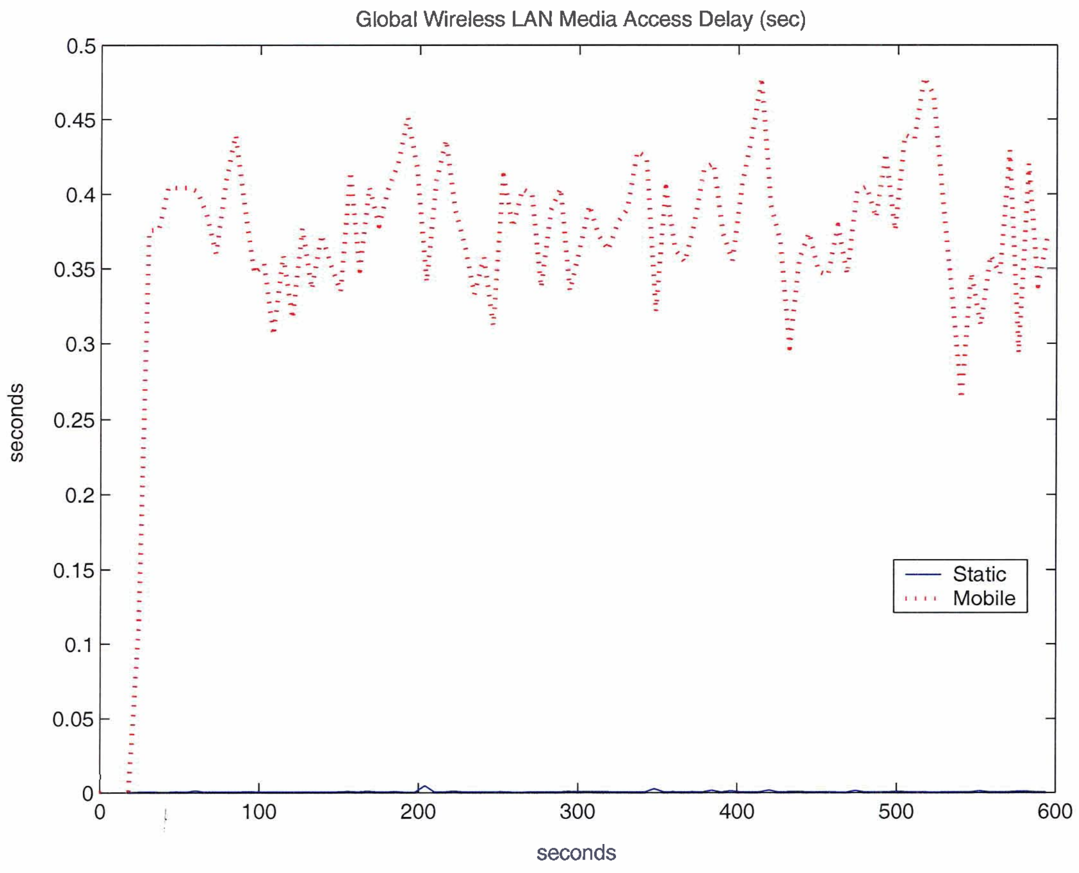


Figure 4.37: Global Wireless LAN Delay (sec)

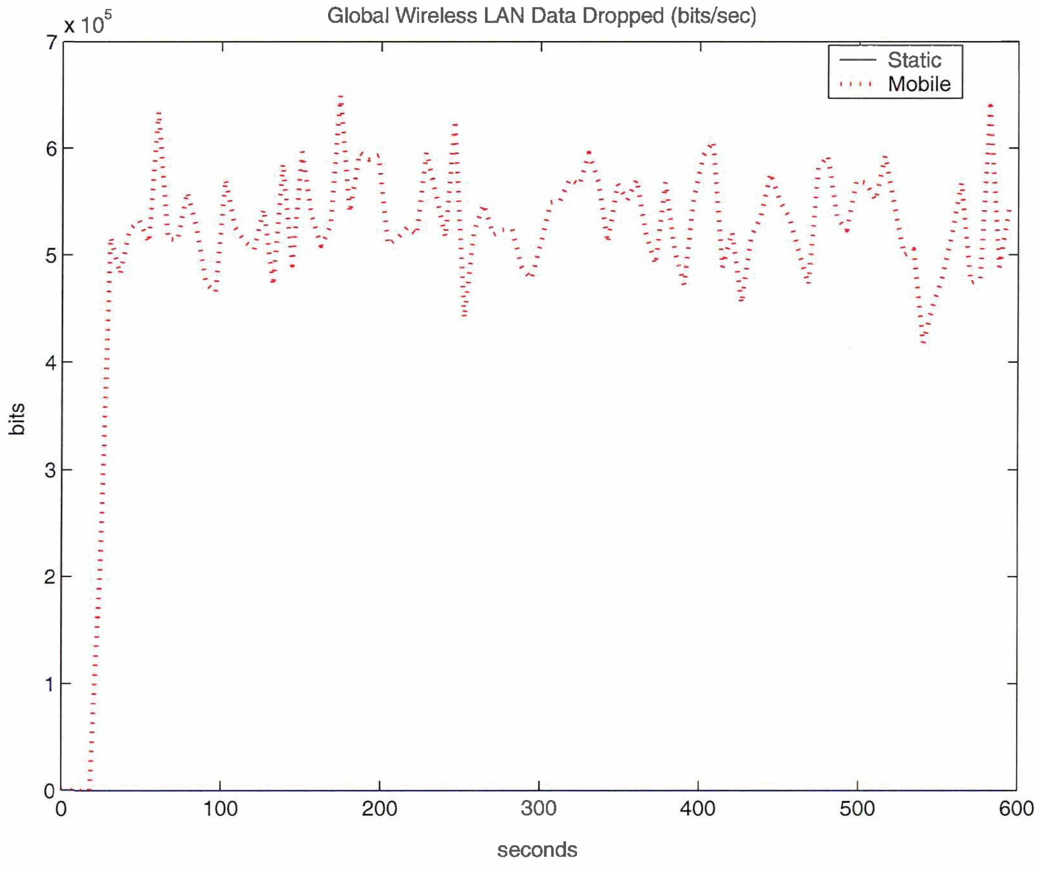


Figure 4.36: Global Wireless LAN Data Dropped (bits/sec)

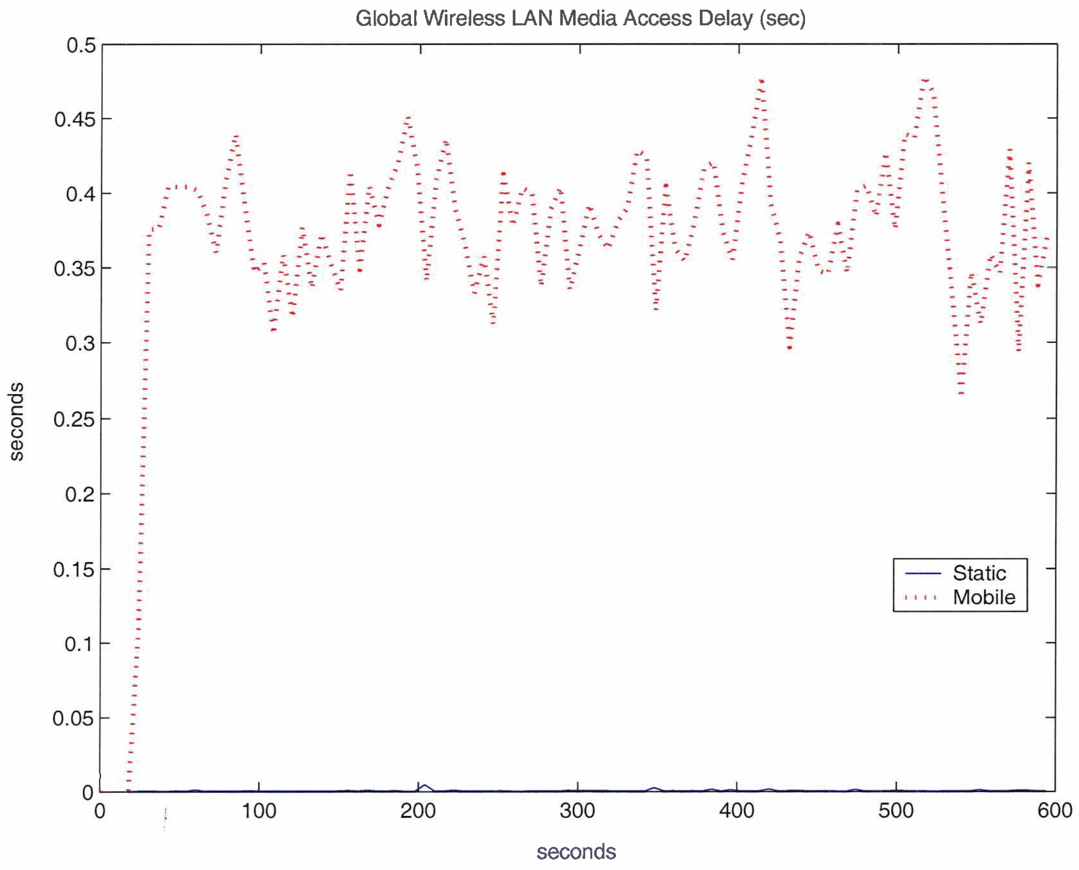


Figure 4.37: Global Wireless LAN Delay (sec)

Chapter 5

Conclusion

In this thesis, we summarize the SIP based traditional conferencing models and analyzed their limitations when applied to IEEE 802.11 wireless networks and used on small handheld devices. Based on this, we proposed a conferencing architecture designed to improve the user scalability and conserve battery energy on the handheld devices. The Conferencing Server in our proposed conferencing model is used as a means of reducing the data processing requirements for our target users. It provides a flexible way to expand the scale of conferencing while maintaining the required QoS requirements. Furthermore, we utilized a voice codec which supports silent suppression in order to increase the bandwidth efficiency.

We implemented an application layer process model in the OPNET simulation tool to model our conferencing server. A series of simulation scenarios were conducted to analyze the conferencing performance in terms of user scalability, flexibility and Quality of Service (QoS) parameters such as end-to-end delay, jitter, packet dropped and WLAN throughput. In the first scenario, we compared the performance of our proposed conferencing model with other traditional conferencing models. It can be seen from our experimental results that the end mixing conferencing model has the best performance when the conferencing user scale is small. On the other hand, the unicast send and unicast receive (UIUO) conferencing model has a longer delay, higher number of packets dropped, and higher throughput compared with other conferencing models.

In the second scenario, we compared our proposed conferencing model with the unicast send and multicast receive (UIMO) conferencing model. The simulation results indicate, our proposed conferencing model has similar QoS performance to the UIMO conferencing model for small number of users, such as ten. However, the UIMO conferencing model is incapable of providing guaranteed QoS as the number of users increases. On the other hand, our proposed model provides more stable services even under a larger user number.

In the third scenario, we conducted additional simulations to illustrate how the handheld mobile users affect the conferencing QoS performance by comparing it with static users.

5.1 Future Work

In the future, our study could be extended to investigate the performance of real time multimedia collaborative communications through the integration of wireless and wired networks using handheld devices. We could also study the performance of other handheld based applications such as on-line learning and multiparty gaming by conducting experiments and user surveys in real world environments. Then, we can propose specific mechanisms to improve the performance based on the user's feedback. We can also study SIP terminal mobility, which allows the user to move between IP subnets while continuing to be reachable for incoming requests and maintaining the ongoing sessions regardless of location change.

The trend now is to design mobile phones that can work under both existing and future cellular networks and the IEEE wireless networks so that they can combine the advantages of these two technologies. Another trend is to integrate existing PSTN with the IEEE wireless networks in order to provide seamless data access and value added services for the users. New standards, such as IEEE 802.16 or WiMax are also being developed and are expected to offer higher data rate and higher capacity[32].

Bibliography

[1] IEEE 802.11 Group. "http://www.ieee.org".

[2] F. Ohrtman. *Voice over 802.11.* Artech House, Boston, MA, 2004.

[3] Jeremiah F. Hayes and Thimma V.J. Ganesh Babu. *Modeling and Analysis of Telecommunications Networks.* Wiley, 2004.

[4] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261 (Proposed Standard), 2002. Updated by RFCs 3265, 3853.

[5] ITU-T. H.323: Packet-based Multimedia Communications Systems, 2003.

[6] H. Schulzrinne and J. Rosenberg. A Comparison of SIP and H.323 for Internet Telephony, 1998.

[7] I. Dalgic, H. Fang. Comparison of H.323 and SIP for IP Telephony Signaling. In *Proceedings of Photonics East*, Boston, Massachusetts, September 1999.

[8] Nortel Networks. A Comparison of H.323v4 and SIP. In *3GPP S2*, Tokyo, Japan, January 2000.

[9] K. Katrinis and G. Parissidis and B. Plattner. A Comparison of Frameworks for Multimedia Conferencing: SIP and H.323. In *8th IASTED International Conference on Internet Multimedia Systems and Applications (IMSA 2004)*, Kauai, Hawai, August 17–19 2004.

[10] A. B. Roach. *Session Initiation Protocol (SIP)-Specific Event Notification.* Number 3265 in Request for Comments. IETF, 2002.

[11] M. Day and S. Aggarwal and G. Mohr and J. Vincent. *Instant Messaging / Presence Protocol Requirements.* Number 2779 in Request for Comments. IETF, February 2000.

[12] J. Rosenberg and H. Schulzrinne. *Models for Multi Party Conferencing in SIP.* Internet Draft. IETF, November 2001.

[13] K. Singh and G. Nair and H. Schulzrinne. Centralized Conferencing using SIP. In *the 2nd IP-Telephony Workshop (IPtel 2001)*, 2001.

[14] Optimum Network Engineering Tool (OPNET). "http://www.opnet.com".

[15] T. Janevski. *Traffic Analysis and Design of Wireless IP Networks.* Artech House, Inc., 2003.

[16] *G.114: Transmission Systems and Media, General Recommendations on the Transmission Quality for an Entire Internet Telephone Connection: One-way Transmission Time.* Number 114 in G. ITU-T, 2000.

[17] J. D. Gibson and W. Bo. Tandem Voice Communications: Digital Cellular, VoIP, and Voice Over Wi-Fi. In *Global Telecommunications Conference (GLOBECOM '04)*, volume 2, 29 Nov.-3 Dec. 2004.

[18] S. Garg, and M. Kappes. Can I Add a VoIP Call? In *Communications, 2003. ICC '03. IEEE International Conference on*, volume 2, pages 779–783, May 11-15 2003.

[19] Audio-Video Transport Working Group, H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. *RTP: A Transport Protocol for Real-Time Applications.* Number 1889 in Request for Comments. IETF, January 1996. Obsoleted By RFC 3550.

[20] M. Handley and J. Crowcroft and C. Bormann, J. Ott. *The Internet Multimedia Conferencing Architecture.* Internet Draft. IETF, 2000.

[21] *G.711: Pulse Code Modulation (PCM) Of Voice Frequencies.* Number 711 in G. ITU-T, 1998.

[22] *G.723.1: Dual Rate Speech Coder for Multimedia Communications Transmitting at 5.3 and 6.3 Kbit/S.* Number 723.1 in G. ITU-T, 1996.

[23] *G.726: 40, 32, 24, 16 Kbit/S Adaptive Differential Pulse Code Modulation (AD-PCM).* Number 726 in G. ITU-T, 1990.

[24] *G.728: Coding Of Speech at 16 Kbit/S Using Low-Delay Code Excited Linear Prediction.* Number 728 in G. ITU-T, 1992.

[25] J. Peterson. *Enumservice Registration for Session Initiation Protocol (SIP) Addresses-Of-Record.* Number 3764 in Request for Comments. IETF, April 2004.

[26] Alan B. Johnson. *SIP: Understanding The Session Initiation Protocol.* Artech House, Inc., 2001.

[27] A. Johnston, S. Donovan, R. Sparks, C. Cunningham, and K. Summers. *Session Initiation Protocol (SIP) Basic Call Flow Examples.* Number 3665 in Request for Comments. IETF, December 2003.

[28] H. Schulzrinne and E. Wedlund. Application-Layer Mobility Using SIP. *Sigmobile Mob. Comput. Commun. Rev.*, 4(3):47–57, 2000.

[29] J. Rosenberg, J. Peterson, H. Schulzrinne, and G. Camarillo. *Best Current Practices for Third Party Call Control (3PCC) in the Session Initiation Protocol (SIP).* Number 3725 in Request for Comments. IETF, April 2004.

[30] F. Vakil, A. Dutta, J-C. Chen, M. Tauil, S. Baba, and N. Nakajima etc. *Supporting Mobility for TCP with SIP.* Internet Draft. IETF, December 2000.

[31] J.Chen and T. Zhang. *IP-Based Next-Generation Wireless Networks: Systems, Architectures, and Protocols.* Wiley-Interscience, New York, 2004.

[32] Teri Robinson. WiMax to the World? *Networker*, 9(4):28–34, 2005.