MULTIMEDIA DELIVERY IN A WIRELESS ENVIRONMENT

By

PETER HANDEL

A THESIS PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE

UNIVERSITY OF FLORIDA

2002

TABLE OF CONTENTS

Abstract of Thesis Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Master of Science

MULTIMEDIA DELIVERY IN A WIRELESS ENVIRONMENT

By

Peter Handel

December 2002

Chair: Professor Abdelsalam Helal
Major Department: Computer and Information Science and Engineering

Our society is definitely infatuated with multimedia. With high quality video and audio comes a steep bandwidth requirement, which is often impossible to satisfy. When one considers this high bandwidth cost across a large audience coupled with our already saturated Internet backbones, traditional unicast quickly becomes a poor choice. Multicasting allows transmission of rich multimedia to a large audience, often mitigating the bandwidth impact equivalent to a single unicast stream.

As mobile computing and wireless technology become less expensive and more widely adopted, mobile users have come to demand a location-transparent quality of service which is difficult if not impossible to satisfy using current data transmission protocols. I propose a multimedia delivery protocol based on the current multicasting protocol, using a custom rebroadcasting extension.

CHAPTER 1
INTRODUCTION

I have always been interested in the concept of connectivity while mobile. The one aspect of this which may become the "Killer App" which the public embraces as the must-have technology of the next century is the ability to efficiently deliver multimedia across a constrained network.

Various other technologies help provide a certain level of quality of service, such as QoS-aware routers and operating systems. They attempt to prioritize packets based on a set of rules, which are often structured to favor time-sensitive packets (such as video conferencing and telnet) and penalize bulk packets (such as ftp). Even these great advances make only a miniscule dent in the bandwidth requirements of full motion video and high quality audio, and thus a new paradigm is required. This new paradigm is even more necessary in the mobile environment, where bandwidth and energy constraints are much more difficult to appease.

This thesis addresses the scenario of a single presentation, which is accessed by many different clients. In the traditional sense, each client would require one unique stream, which would quickly exceed the bandwidth available to the server. In addition, some mobile clients may not be close enough to the server to tune in to the broadcast, but may be close enough to a client that is currently viewing the program. These clients could rebroadcast the stream, but how can we do this efficiently?

The primary goal of this thesis is to first present the constraints high-quality multimedia delivery faces in the mobile environment, and then to present a custom extension to a well-known protocol, which addresses the issues raised. To this goal, the next chapter clarifies not only the magnitude of the problem that wide dissemination of high-quality multimedia poses to the backbone of the internet, but also the size of the audience requesting this service.

Chapter 3 begins with a brief review of the basic protocols of the Internet, necessary as a foundation for any discussion of multimedia transfer over the Internet. It then examines the concept of streaming, and discusses current streaming technology and its limitations. This includes RealPlayer (with over 20 million users), QuickTime, and the Microsoft MediaPlayer, along with the mbone, which is a network of multicast-aware computers and routers that, when necessary, unicast to each other over non-multicast-aware computers and routers. It also evaluates currently proposed yet not widely implemented multicasting technologies, such as the multicasting extensions of IPv6.

The unique additional difficulties mobility affords multimedia transfer is presented in Chapter 4. Here, I discuss the problem mobility presents, certain assumptions on which the code presented relies, and then finally my custom approach, the extension to the common multicasting protocol, which exploits certain ad hoc properties to extend the usable range of a multicasting protocol.

Chapter 5 presents the results of my experimentation and simulation, pitting unicasting against multicasting in the rebroadcast of data to clients outside the base stations range yet within range of another client.

The conclusion, Chapter 6, summarizes the results of the coding experiment and the simulation experiment, and then applies these results to my vision of future multimedia broadcasting.

CHAPTER 2
MULTIMEDIA

Video Formats

A quick glance at the 2001 Q4 Nielsen//NetRatings download statistics for popular media delivery clients such as RealPlayer, QuickTime, and Windows Media Player readily convince us that the delivery of multimedia is arguably one of the most sought-after and heavily used segments of the internet. High-profile web destinations draw magnitudes of users with high-resolution and high-frame rate multimedia, while the backbones of the Internet groan under the immense weight of the bandwidth these streams require. As users grow accustomed to media with higher resolution and higher frame rates, the bandwidth required increases rapidly.

Rich video and multimedia only exacerbate the bandwidth constraints especially prevalent in today's mobile environment. Full motion video at NTSC resolution and 16-bit color requires a data transmission rate of approximately 20MB/sec, which is about 160Mbit/sec – 720x486 x 16-bit color x 30 fps. Putting this into perspective, a 56k modem has a maximum transmission rate of 0.056Mbit/sec, and the standard SCSI protocol can transfer 40Mbit/sec. Since almost no Internet users have access to this speed of service, compression is used to dramatically reduce the amount of data transmitted. Compression, however, can only do so much, as the level of compression (using a lossy algorithm) is inversely related to the quality of the transmitted data.

All compression methods fall under one of two categories: lossless (also known as non-lossy) and lossy. Lossless compression promises to compress data without any loss of information; the compressed data can then be expanded to an exact replica of the original information. In other words, the former promises to reduce the size of the data by detecting redundancy in data with low entropy without losing any information contained in the original data. Commonly used forms of lossless compression include Zip (by PkZip), Gzip, and StuffIt. Lossy compression, on the other hand, uses various techniques to remove superfluous data from the original audio and video stream. Some of the more popular lossy algorithms include JPEG, PNG, and AVI. Figure 1 demonstrates the difference between lossy and lossless compression.

Figure 1 The original (left), versus JPEG compression (right)

As we see on the right, some artifacts of the JPEG compression are visible around the lettering and the top edge. Next, in Figure 2, we zoom in on the top left corner of the processor of each photo, where the reliance on interpolation becomes much more evident.

Figure 2 Zooming in, the original (left), versus JPEG compression (right)

For audio, the easiest and most commonly employed lossy technique for reducing the data requirements is to lower the sampling rate from 44.1khz (CD quality) to 22khz or even 11khz, and then reducing the bit depth from 16 bits to 8 bits or lower. While effective, these methods severely deteriorate the quality of the audio. Psychoacoustic models, on the other hand, promise extreme levels of compression with minimal loss of sound quality by exploiting certain facts about how our ears interpret sounds. A common high-quality compression/reduction method called Perceptual Coding removes sound information that our brain would anyways discard by eliminating quiet sounds that occur immediately after louder sounds. This method is used in the popular MP3 audio format.

To easily reduce the size requirements of a video stream, the video resolution, frame rate, and color depth may be reduced, but this often results in unacceptable video, as motion may appear jerky and the picture may not even be recognizable. Instead, a better approach would be to first compress each frame using a lossy algorithm such as JPEG and then transmit only the differences between frames. While better than merely reducing the resolution and frame rate, this method does not exploit any of the commonalities between frames. The MPEG format addresses this issue, and is consequently based on the following concept: it sends three types of frames, namely I, B, and P. The I frame contains all the information necessary for

one complete frame of video, and is therefore used to start a stream. The B and P frames contain information describing the differences between video frames. When the differences between video frames are low, a higher percentage of the smaller-sized B and P frames suffice for smooth video. However, when the changes between video frames become large (such as in a scene change), more I frames are needed, as neighboring frames may have nothing in common.

Requirements

The requirements for traditional transmission of multimedia content is a server with abundant amounts of bandwidth availability [BAG99], and the processing power and memory to handle multiple client requests simultaneously.

CHAPTER 3
MULTIMEDIA DELIVERY SYSTEM

Internet Basics

The Internet can be seen in its most simple form as a group of computers which are connected. Every computer is able to reach every other computer, either directly or indirectly. The most basic example of a network is two computers that are directly connected to each other, as shown in Figure 3. They may be connected in any number of ways including Ethernet cable, wireless connection, RS-232 cable, and many more.

Figure 3 Basic network between two laptops

The next logical step is to add more hosts, as shown in Figure 4. Here, a packet of information sent from C1 to C3 travels through C2. Depending on the transmission method, C2 may realize that the packet is not meant for it at either the link or IP layer. The advantage of filtering packets at the link layer is that the processor need not spend valuable resources on deciding whether the packet is

destined for itself. The disadvantage to this method is that the network interface must implement additional hardware.

Figure 4 Three-host network

As the number of clients is increased, routers and switches are added to segment the network, acting as a policeman directing traffic. This is shown in Figure 5, where we forward packets of information which are destined for a machine which isn't within the local network by using multiple routers.

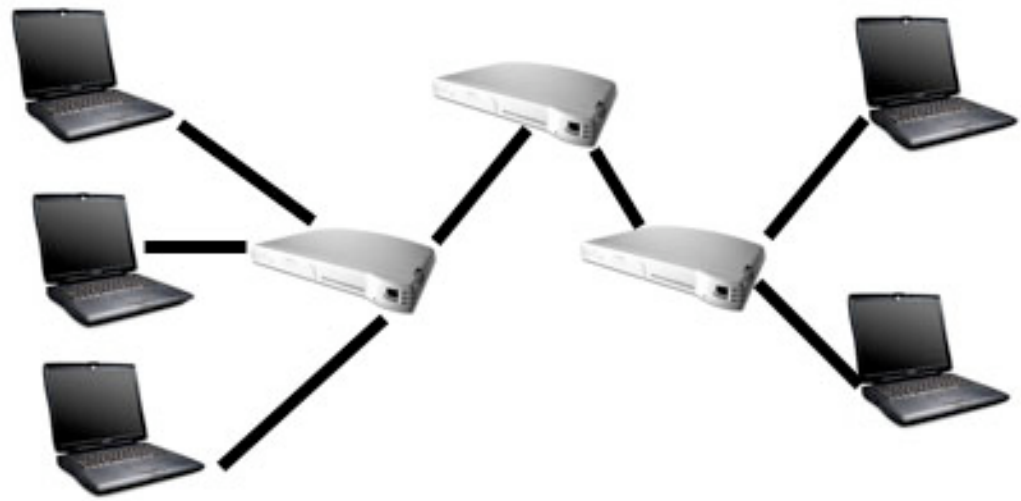

Figure 5 A more complex network

The main difference between a switch and a router is that a switch uses the packet address rather than the protocol itself to determine how to filter traffic and therefore where a packet should go. Adding switches and routers not only increases available bandwidth in busy environments but also adds a level of security, as a host on an endpoint cannot snoop packets meant for another host.

As our imperfect world may inject anomalies into the data stream, a mechanism of detecting and recovering from data corruption becomes necessary. Therefore, the TCP/IP protocol suite is made up of four different layers: the link layer, the network layer, the transport layer, and the application layer. These layers are designed to ensure that data arrives to the correct recipient uncorrupted and optionally in the same order as sent.

The link layer is the layer closest to the actual hardware media. Its purpose is to take care of sending and receiving IP packets and handling ARP/RARP requests and replies. The network layer consists mainly of the Internet Protocol (IP) and Internet Group Management Protocol (IGMP) protocols, which are used as a delivery service, as they contain the source and destination address, along with other information used to help the packet get to its destination. Handling data corruption and optionally packet retransmission and reordering, the transport layer consists of the TCP and UDP protocols. The application layer, as the name implies, is the application itself, such as Netscape or RealPlayer.

TCP and UDP Layer

Within an IP packet, we find encapsulated either a TCP or UDP packet. While the IP layer deals with where a packet comes from and where it is going, the

TCP & UDP layers make sure that the packet didn't become mangled along the way by using 16 bit checksums. UDP is the minimalists choice, as it only provides a checksum and therefore only guarantees that the data is not corrupted. TCP goes a step further and guarantees not only that the data is not corrupted, but also that it is delivered. TCP also uses a sequence numbering system to ensure that packets arrive at the application layer in the order in which they were sent [POS80a, POS80b].

When writing an application designed to transmit multimedia in a streaming fashion, UDP is the preferred protocol, as the extra features TCP provides are superfluous and not worth the overhead they impose. Reordering and packet loss recovery may be done easily and efficiently at the application level, allowing the use of UDP.

Unicast vs. Multicast

Multimedia is most commonly delivered using a unicast connection between a server and a client. A packet is labeled as a unicast packet if its destination is a single host. A multicast packet, on the other hand, is a packet "destined for a set of hosts that belong to a multicast group" [STE94, pg. 8]. While the destination address of a unicast packet is a single host, the destination address of a multicast packet is a special class D IP address, ranging from 224.0.0.0 to 239.255.255.255 [MEY98]. Therefore, when a client desires to start receiving a multicast feed [HAN99a, HAN99b], the IP address of the transmission must be known, and a request to receive packets must be made to the first router (as shown in Figure 6). If client B were interested in receiving a broadcast from the server, it would first inform router D via IGMP by sending a "Group Join" message. Router D then informs router C, which

finally informs router A to begin sending a stream towards router C [DEE89, DEE99, FEN97].

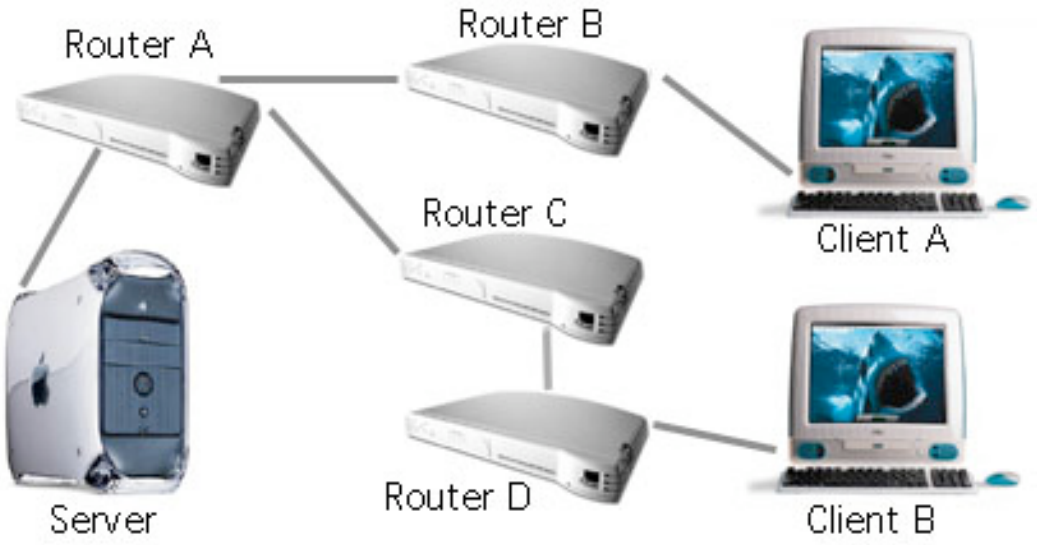

Figure 6 Standard broadcasting model

Thus, one stream of packets from the server can be used to satisfy a large number of clients [PUL99], as the routers take it upon themselves to duplicate the stream as often as necessary [ARM92, BASIC]. This duplication, along with other inherent multicast differences, can introduce some unexpected bandwidth and latency constraints [DUB98].

Currently, the overwhelming majority of multimedia transmission on the Internet is done through unicasting. This is due to not only the high cost and rarity of multicasting-aware server software, but also to the fact that many of the routers on the Internet today are not configured to support multicasting. There are a few techniques that can be employed to overcome the lack of support: namely, tunneling and proxies.

Tunneling uses the concept of encapsulation to overcome multicasting-disabled or multicasting-unaware routers by placing a multicast packet inside of a

standard unicast packet. A unicast connection is therefore used to bridge the multicasting-disabled router. In the example below (Figure 7), the server and client are separated by three routers, of which the first and last are multicasting-enabled [CIS02].

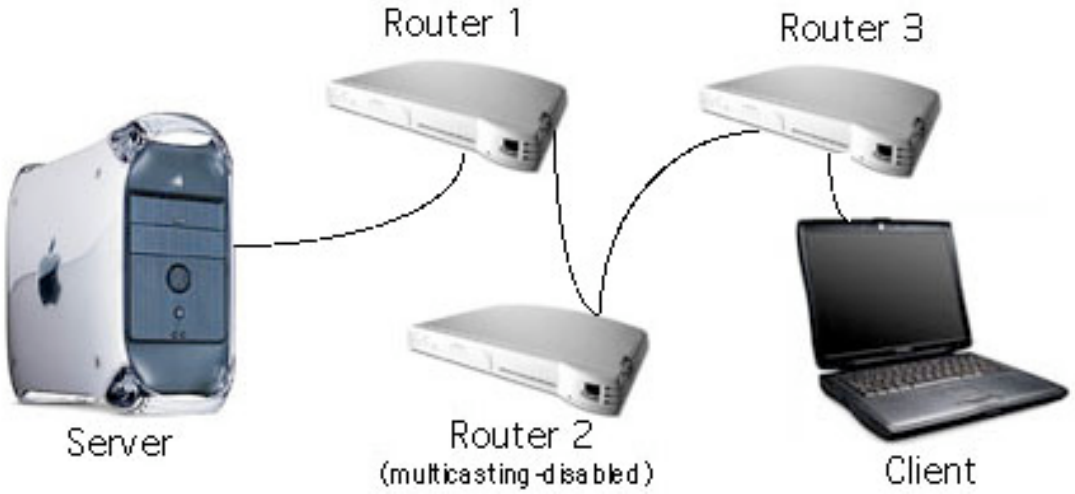

Figure 7 Multicasting across a non-multicast router

Streaming

The concept of streaming has been directly responsible for the explosion of high-quality audio and video on the Internet. Before streaming, Internet users were required to download an entire media clip before playback would start. This made the dissemination of live real-time video impossible; sampling large media clips was likewise difficult, as most Internet users had only a very limited amount of bandwidth at their disposal (usually a 28.8k modem or slower). With the advent of streaming, users are now able to begin viewing a multimedia presentation as soon as the first few blocks of data for the transmission arrive. Not only is this more convenient for users

on low-bandwidth connections, it also opens up the realm of live multimedia broadcasts over the Internet.

When streaming multimedia, the compression format employed may be either stateless or stateful. Stateless implies that any packet of received information can be used, regardless of whether any other packets have been received, have been corrupted, or have been received out-of-order. A stateful connection, on the other hand, requires the correct delivery of certain previous packets for later packets to be useful. The benefit of a stateful connection is that once a certain state has been established, subsequent packets will not require as much information.

The Real-Time Streaming Protocol (RTSP) is an open standard for streaming data. The difference between streaming via standard HTTP and RTSP is as follows:

> RTSP streaming may be defined as the transfer of multimedia over a network while it is playing, as opposed to downloading the data and storing it locally before playing it. With HTTP streaming, the movie data is sent to you as fast as your network connection can handle it. Once enough data has been received, the movie will begin to play. With RTSP streaming, only the movie data is sent as you need it, so the data rate of your stream has to be smaller than the network's current speed. [APP02]

Surprisingly, all the major media players have adopted this open standard, and play multimedia transmitted via RTSP as well as their own proprietary formats.

Current Streaming Technology

Currently, there are a few solutions for multicasting audio and video, but none have good support for mobile clients. Here, I will examine a few of the more popular multicasting solutions, and later discuss why they fall short when applied to the wireless environment.

With over 21 million users per month as of April 21, 2000 and a whopping 115 million unique registered users, RealPlayer is arguably the most widespread media player application in use today. The current G2 version supports numerous features attractive to mobile users, such as the ability to dynamically reduce and increase the quality of the broadcast based on changing network conditions. As network conditions worsen, RealPlayer employs various other tricks to maintain the perception of the original video quality. To achieve a certain target frame rate, the RealPlayer client may create tween frames, which are frames interpolated by the client from received neighboring frames to increase the frame rate and overall smoothness of the video clip. Creating tween frames, however, currently requires copious amounts of processing power, and is therefore not done as consistently as we would like.

The Windows Media Player, while not nearly as old or as ubiquitous as RealPlayer, is still rather widely used, with approximately 8.6 million monthly users as of April 2000. In side-by-side evaluations with RealPlayer at a transmission rate of 80kbit/sec, frame rates have been shown to be higher, at the cost of frame resolution and color accuracy.

Apple, being the newest streaming media player, has had a surprising 7.5 million monthly users of its QuickTime player as of April 2000. This is most likely due to the availability of the Star Wars Episode I: The Phantom Menace trailers exclusively in the QuickTime format. After performing side-by-side comparisons of the frame rate of a presentation at a certain bandwidth, it becomes clear that the Quicktime player, in its current version, does not perform as well as expected,

especially when compared to the Microsoft Windows Media Player (ironically also available for the Macintosh MacOS) and the RealPlayer G2. This may be due to the different codecs employed by the various players.

The currently proposed solutions are also mainly based on the current multicasting technology. The new IPv6 protocol [CRA98] proposes to change numerous core elements of the current IPv4 networking. While the main goal driving the creation of IPv6 was to expand the usable number of IP addresses to 128 bits, the designers also decided to remove broadcasting, as it had proven to be difficult to implement without excessive wasted bandwidth. "There are no broadcast addresses in IPv6, their function being superseded by multicast addresses" [HIN98, pg 2]. IPv6 also offers a built-in (and therefore more finely grained) quality of service control, allowing the system administrator to easily and precisely prioritize bandwidth availability.

CHAPTER 4
MULTIMEDIA DELIVERY IN THE MOBILE ENVIRONMENT

Supporting Multicast of Video in Wireless LANs

The multicasting protocol works surprisingly well given the fact that the Internet was designed for and now almost exclusively carries unicast transmissions. However, once we move to a mobile environment, many fundamental assumptions must be changed: bandwidth may diminish or temporarily become completely unavailable as the client moves away from the base unit. Sensitivity towards power usage is also imperative as mobile computers have a very limited power source.

The base model for these experiments is the following:

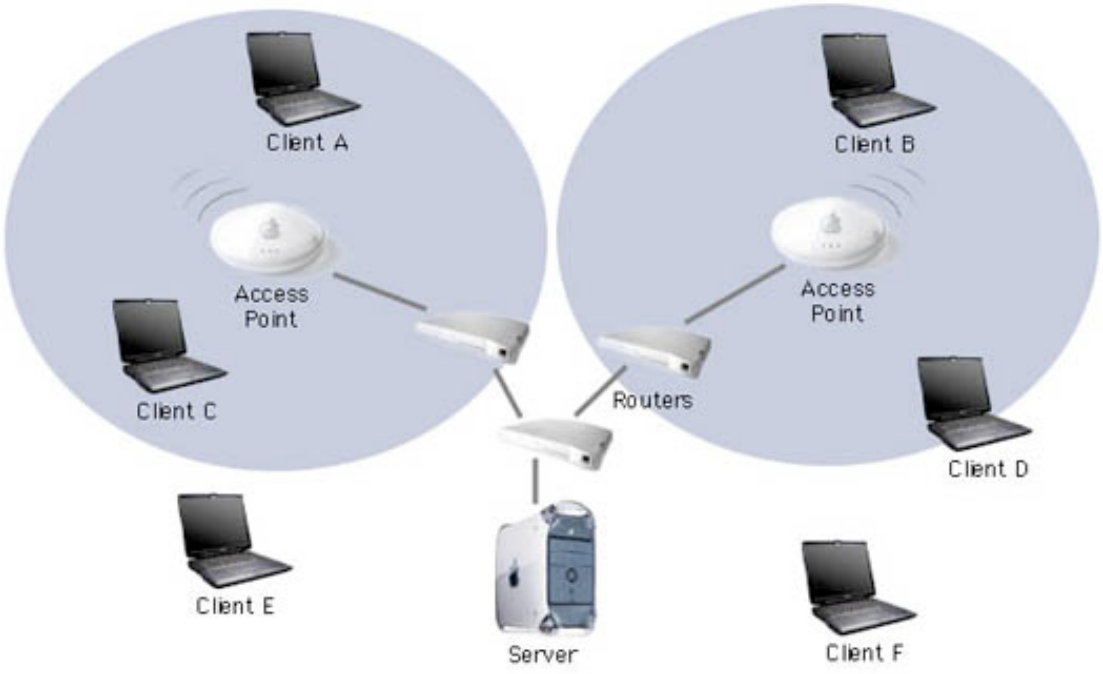

Figure 8 Standard configuration

Here in Figure 8, the server transmits one stream to the first router, which splits the stream based on the location of listening clients. In this case, this router duplicates the one stream from the server into only two streams, and sends them to the respective access points on their respective local networks. The routers communicate amongst themselves via one of the numerous multicast routing protocols [BAL97, EST98, MOY94a, MOY94b, THA99, WAI88] to decide whether a router requires a copy of the stream.

Proposed Protocol

The Wireless Multi-Media Delivery Protocol (WMMDP) proposed here is designed to sit on top of the multicasting protocol:

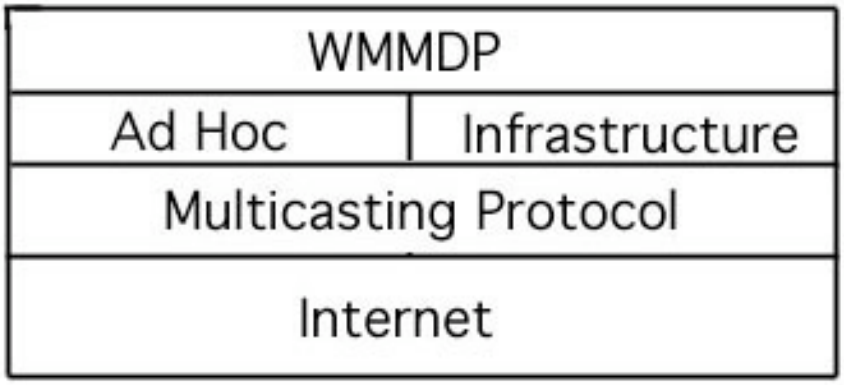

Figure 9 Wireless Multi-Media Delivery Protocol

The extension protocol I propose uses both the Ad Hoc and Infrastructure transmission modes to interface with the standard multicasting protocol, which allows for efficient broadcast of data over the internet.

This proposal involves only the last hop of a standard multicast. The server begins sending data regardless of any client initiation. It is the responsibility of the first router to discard all packets if no clients are listening.

The first client of a subnet to request a certain broadcast will negotiate with its router via IGMP the fact that it wishes to receive this broadcast. The routers then handle getting a stream of the multimedia to the router closest to the client via PIM or any other multicast routing protocol.

The client for this Wireless Multimedia Delivery Protocol (WMDP) listens on port 3845 for the multicast stream. Once the data arrives, it streams the data to a temporary file used to buffer the data, and also to keep older packets for other clients who may request previously used packets. Each client will also listen on port 4845 for communication from other clients. This inter-client communication will consist mainly of requests for missed packets. Each packet will have embedded within itself the packet number (by the protocol, as UDP does not ensure in-order packet delivery) and the source address (by the network stack).

When a client C1 notices that it has missed receiving a packet P1 within a small timeout, it sends a request for retransmission of this packet P1 to any nearby client C2. The different responses C2 provides are discussed below:

Standard Mode

In Standard Mode (SM), C2 simply sends C1 the packet via a standard ad hoc UDP connection. This is the straightforward approach.

Multicast Mode

In the Multicast Mode (MM), a nearby client, C2, which has received P1, will begin rebroadcasting the stream to C1 via ad hoc mode by sending the packets to the special ad hoc multicast address.

The simulation will show the affect of the three rebroadcast situations on stream availability:

1. No rebroadcasting. The clients will need to stay relatively close to the base station—within 300ft in an office setting, and within 2000ft outdoors. We will use 500ft as an average range.
2. Simple direct rebroadcasting. Here, the range is extended beyond the base station, but excessive network utilization will saturate the 2Mbit/sec bandwidth available.
3. Multicast rebroadcasting. Clients rebroadcast to the multicast address rather than to the specific client in need. We will examine optimizations to cut down the number of clients rebroadcasting within a certain range.

An arbitrary assignment of five states will be made to aid in differentiating between the different situations that may occur in the course of client mobility. Each client can be in only one of five states:

0. Standard state; client listening to a stream
1. Our client receives a request for help, and we will help
2. We need a rebroadcast, but have not received a helper
3. We need a rebroadcast, and have received a helper
4. We are receiving help, and another client needs our help

Each client itself listens on three channels:

Channel A: Listens via infrastructure mode to the broadcast

Channel B: Listens via ad hoc mode to the broadcast

Channel C: Listens via ad hoc mode to the control channel

Channel C is used to send and receive help requests from and to clients.

Figure 10 shows the clients possible states. The arrows between states indicate the only possible state changes a client may perform.

State 0: {

Remain in 0: Client continues listening to the broadcast

Go to 1: Client receives request for help, and helps

Go to 2: We wander out of range, and require help

}

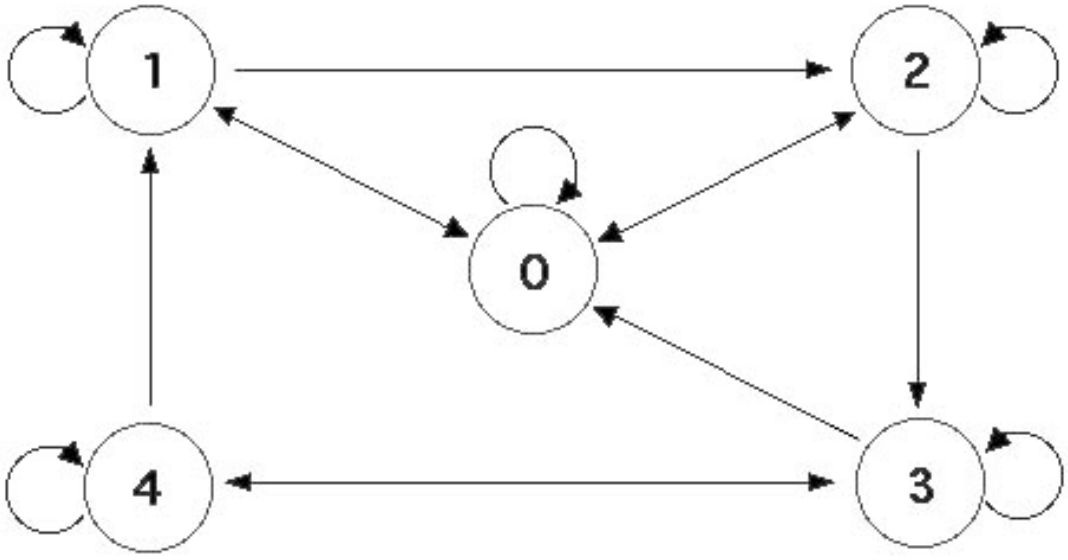

Figure 10 WWMDP State Diagram

State 1: {

Remain in 1: Some client(s) still require assistance; will continue helping

Return to 0: Either helped client sends “no longer needed,” or does not respond to acknowlegement

Go to 2: We have gone out of range. Any clients using our services also suffer.

}

State 2: {

Remain in 2: We still need help, and have not received any

Go to 3: We have received help from someone

Go to 0: We have wandered back into the range of the infrastructure broadcast

}

State 3: {

Remain in 3: We continue requiring rebroadcasts from another helping client

Go to 0: Back in range of the base station, we can now once again use infrastructure

Go to 4: Another client needs help from us; we oblige

}

State 4: {

Remain in 4: We continue to need help, and also continue to rebroadcast for others

Go to 1: We reenter the infrastructure range, but continue helping others

Go to 3: If client we are helping sends “no longer needed” or doesn’t respond

}

Interesting Issues

Several interesting issues may arise due to the mobile nature of the clients. Here, I examine a few of the more common possibilities/occurrences, such as hop count optimization, isthmus effect optimization, and the effect of lazy acknowledgement.

If we track the number of hops each client is from the infrastructure area, we may use this to make wise decisions during rebroadcast negotiations where multiple clients are available. If a client has a choice to receive a rebroadcast from either a distant (high hip count) or close (low hop count) client, it should choose the closer, as not only will it have a lower latency stream, but also may indicate that it is less mobile and therefore more likely to continue receiving the stream.

The isthmus effect is a routing dilemma where, to reach a certain client B from a machine A, packets must be routed in a physically opposite direction. This dilemma in the wired environment is reduced to an optimization problem here. While we do not concern ourselves with the underlying ad hoc routing protocol, we do want to be confident in our decisions on whom to ask to continue rebroadcasting.

Lazy acknowledgement becomes extremely useful when we attempt to prune the amount of clients rebroadcasting the stream. We ask for each client who is listening to our rebroadcast to send us an acknowledgement every 30 seconds. If two consecutive acknowledgements are missed, we prune that specific client from the list of clients listening to our rebroadcast. When the number reaches zero, we know that we may stop rebroadcasting; however, if a mobile client moves into our area and uses

our rebroadcast stream without our knowledge, that client must initiate communication with us and negotiate rebroadcasting.

CHAPTER 5
SIMULATION AND EXPERIMENTAL EVALUATION

As mentioned in the final paragraphs of Chapter 5, the custom extension to the multicasting protocol proposed would take into account the rapidly fluctuating bandwidth available to the mobile client based on the distance from the base station. This is done by rebroadcasting packets from clients receiving the stream to clients wishing to receive the stream yet tragically out of range of the base station. The three major cases considered were: 1) no rebroadcasting, 2) simple direct rebroadcasting, and 3) multicast rebroadcasting. Also, how many nodes must be present in a given area to provide coverage to all nodes?

Before performing any experiments, the following hypothetical outcome seems most likely: In an area with a sparse number of clients, the danger of flooding the local area with duplicate packets is minimal. If many clients converge to one physical area, the resulting duplication will flood the wireless network and become counterproductive.

For the following simulations, we assume a clear 800 feet by 600 feet area with a base station at position 400, 300. Both the base station and the individual nodes use a wireless card with a range of 150 feet, whose signal strength falls off logarithmically at the edge of their range.

Based on the general principles of cell bandwidth availability and range, here is a theoretical graph of the percentage of packets received versus the distance from the base station:

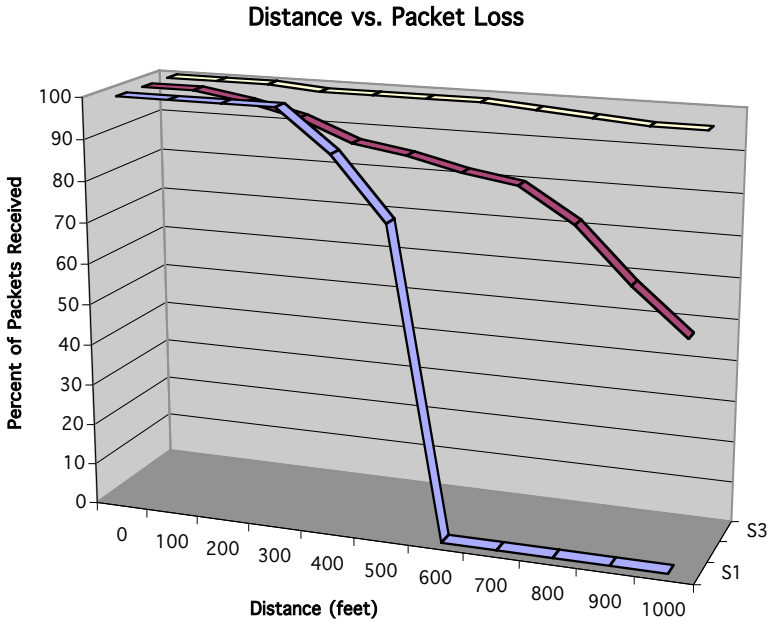

Figure 11 Distance vs. Packet Loss graph

The standard case (blue), with no rebroadcasting, performs well up to the edge of its range (500ft). The unicast rebroadcast mode (red) also performs well, but past the range of the base station, many clients are requesting a rebroadcast, which floods the cell with many duplicate packets. The multicast rebroadcast mode (yellow) strongly reduces the amount of duplicate packets, as many clients can share one multicast rebroadcast.

Network Simulation

I performed numerous simulations of the WMDP using the Berkley Network Simulator version 2. These simulations were designed to differentiate the performance of the three rebrodacasting options mentioned throughout this thesis: the

standard straightforward no-rebroadcasting configuration, the unicast-rebroadcasting configuration, and the multicast-rebroadcast configuration.

After working around numerous incompatibilities between the multicasting package and the wireless package by using suboptimal approximations, we see that the hypothesis above was confirmed. However, since these approximations were suboptimal, it seemed that writing a new network simulator from scratch would be best. My network simulator simulates one base station in the middle of a field, and adds the specified number of mobile clients which may or may not move in a random direction. This simulation is run for 60 seconds, after which it shows the average percentage of the grid coverage and the average percentage of nodes that received the stream. For example, if all nodes were in range of the base station, then the grid coverage would be 12% and the nodes-received percentage would be 100%.

Each 60-second simulation was run 50 times, and then the percentages were averaged. The results are shown below, numerically in Figure 12, and graphically in Figure 13.

Number of Nodes	Grid Percentage	Node Percentage
2	12	9.12
4	12.66	15.36
8	15.96	22.26
12	24.88	31.14
16	40.92	48.12
24	65	67.16

Figure 12 Simulation results

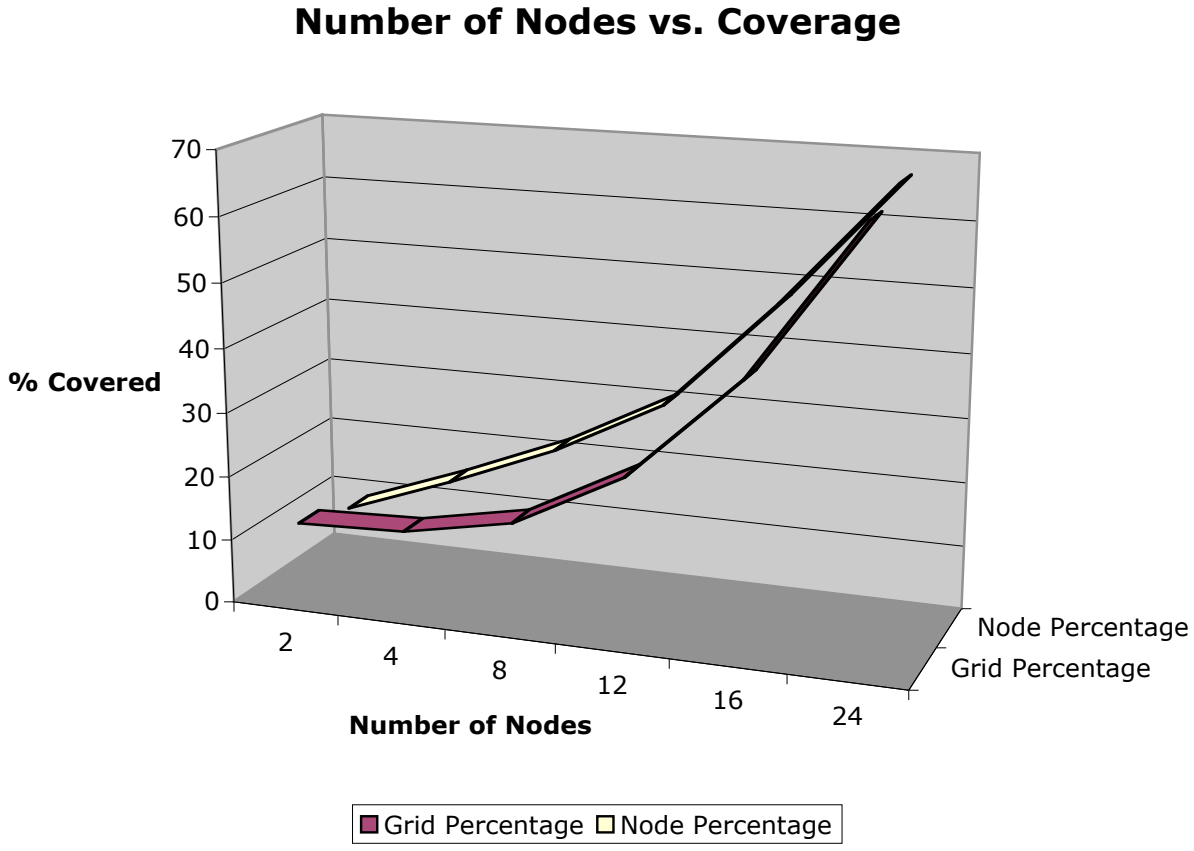

Figure 13 Number of Nodes vs. Coverage

CHAPTER 6
CONCLUSION

In conclusion, multicasting promises to drastically reduce the amount of bandwidth required by a server to satisfy a large number of clients requesting a broadcast of multimedia. However, multicasting by itself does not adequately address availability issues, as mobile users may roam away from the base transmitting station. The proposed extension to the multicasting protocol restores a large amount of mobility previously lost to the range of the base transmitting station, and extends this range based on the number of listening clients within close proximity.

BIBLIOGRAPHY

APP02 Apple Computer, "QuickTime Streaming" July 2002, http://www.apple.com/quicktime/authoring/hinttrack.html Verified: January, 2002.

ARM92 Armstrong, S., Freier, A., and Marzullo, K., "RFC 1301: Multicast Transport Protocol" February 1992, http://www.ietf.org/rfc/rfc1301.txt Verified: November 8, 2002.

BAG99 Bagnall, P., Briscoe, R., and Poppitt, A., "RFC 2729: Taxonomy of Communication Requirements for Large-scale Multicast Applications" December 1999, http://www.ietf.org/rfc/rfc2729.txt Verified: November 8, 2002.

BAL97 Ballardie, A., "RFC 2201: Core Based Trees (CBT) Multicast Routing Architecture" September 1997, http://www.ietf.org/rfc/rfc2201.txt Verified: November 8, 2002.

CIS02 "Cisco IP Multicast Groups External Homepage" ftp://ftpeng.cisco.com/ipmulticast/index.html Verified: November 8, 2002.

CRA98 Crawford, M., "Transmission of Ipv6 Packets over Ethernet Networks" December 1998, http://www.ietf.org/rfc/rfc2464.txt Verified: November 8, 2002.

DEE89 Deering, S., "RFC 1112: Host Extensions for IP Multicasting" August 1989, http://www.ietf.org/rfc/rfc1112.txt Verified: November 8, 2002.

DEE99 Deering, S., Fenner, W., and Haberman, B., "RFC 2710: Multicast Listener Discovery (MLD) for IPv6" October 1999, http://www.ietf.org/rfc/rfc2710.txt Verified: November 8, 2002.

DUB98 Dubray, K., "RFC 2432: Terminology for IP Multicast Benchmarking" October 1998, http://www.ietf.org/rfc/rfc2432.txt Verified: November 8, 2002.

EST98 Estrin, D., Farinacci, D., Helmy, A., Thaler, D., Deering, S., Handley, M., Jacobson, V., Liu, C., Sharma., P., Wei, L., "RFC 2362: Protocol Independent Multicast – Sparse Mode (PIM-SM): Protocol Specification" June 1998, http://www.ietf.org/rfc/rfc2362.txt Verified: November 8, 2002.

FEN97 Fenner, W., "RFC 2236: Internet Group Management Protocol, Version 2" November 1997, http://www.ietf.org/rfc/rfc2236.txt Verified: November 8, 2002.

HAN99a Handley, M., Schulzrinne, H., Schooler, E., Rosenberg, J., "RFC 2543: SIP: Session Initiation Protocol" March 1999, http://www.ietf.org/rfc/rfc2543.txt Verified: November 8, 2002.

HAN99b Hanna, S., Patel, B., and Shah, M., "RFC 2730: Multicast Address Dynamic Client Allocation Protocol (MADCAP)" December 1999, http://www.ietf.org/rfc/rfc2730.txt Verified: November 8, 2002.

HIN98 Hinden, R., and Deering, S., "RFC 2373: IP Version 6 Addressing Architecture" July 1998, http://www.ietf.org/rfc/rfc2373.txt Verified: November 8, 2002.

MEY98 Meyer, D., "RFC 2365: Administratively Scoped IP Multicast" July 1998, http://www.ietf.org/rfc/rfc2365.txt Verified: November 8, 2002.

MOY94a Moy, J., "RFC 1584: Multicast Extensions to OSPF" March 1994, http://www.ietf.org/rfc/rfc1584.txt Verified: November 8, 2002.

MOY94b Moy, J., "RFC 1585: MOSPF: Analysis and Experience" March 1994, http://www.ietf.org/rfc/rfc1585.txt Verified: November 8, 2002.

POS80a Postel, J., "DOD Standard Transmission Control Protocol" January 1980, http://www.ietf.org/rfc/rfc761.txt Verified: November 8, 2002.

POS80b Postel, J., "User Datagram Protocol" August 1980, http://www.ietf.org/rfc/rfc768.txt Verified: November 8, 2002.

PUL99 Pullen, M., Myjak, M., and Bouwens, C., "RFC 2502: Limitations of Internet Protocol Suite for Distributed Simulation in the Large Multicast Environment" February 1999, http://www.ietf.org/rfc/rfc2502.txt Verified: November 8, 2002.

STE94 Stevens, W. Richard, *TCP/IP Illustrated,* Volume I Copyright © 1994 by Addison-Wesley, Reading, MA.

THA99 Thaler, D., "RFC 2715: Interoperability Rules for Multicast Routing Protocols" October 1999, http://www.ietf.org/rfc/rfc2715.txt Verified: November 8, 2002.

WAI88 Waitzman, D., Partridge, C., and Deering, S., "RFC 1075: Distance Vector Multicast Routing Protocol" November 1988, http://www.ietf.org/rfc/rfc1075.txt Verified: November 8, 2002.

APPENDIX
NETWORK SIMULATION SCRIPT

```
#
# Copyright (c) 1996 Regents of the University of California.
# All rights reserved.
#
# Redistribution and use in source and binary forms, with or without
# modification, are permitted provided that the following conditions
# are met:
# 1. Redistributions of source code must retain the above copyright
#    notice, this list of conditions and the following disclaimer.
# 2. Redistributions in binary form must reproduce the above copyright
#    notice, this list of conditions and the following disclaimer in the
#    documentation and/or other materials provided with the distribution.
# 3. All advertising materials mentioning features or use of this software
#    must display the following acknowledgement:
#      This product includes software developed by the MASH Research
#      Group at the University of California Berkeley.
# 4. Neither the name of the University nor of the Research Group may be
#    used to endorse or promote products derived from this software without
#    specific prior written permission.
#
# THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND
# ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
# IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
# ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
# FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
# DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
# OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
# HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
# LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
# OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
# SUCH DAMAGE.
#
# @(#) $Header: /usr/src/mash/repository/vint/ns-2/tcl/ex/mcast.tcl,v 1.12
1999/09/10 22:08:41 haoboy Exp $
# updated to use -multicast on and allocaddr by Lloyd Wood

#
# Simple multicast test.  It's easiest to verify the
# output with the animator.
# We create a four node start; start a CBR traffic generator in the center
# and then at node 3 and exercise the join/leave code.
#
# See tcl/ex/newmcast/mcast*.tcl for more mcast example scripts

# ======================================================================
# Define options
# ======================================================================
set opt(chan)           Channel/WirelessChannel    ;# channel type
set opt(prop)           Propagation/TwoRayGround   ;# radio-propagation
model
set opt(netif)          Phy/WirelessPhy            ;# network interface type
set opt(mac)            Mac/802_11                 ;# MAC type
```

```
set opt(ifq)            Queue/DropTail/PriQueue    ;# interface queue type
set opt(ll)             LL                         ;# link layer type
set opt(ant)            Antenna/OmniAntenna        ;# antenna model
set opt(ifqlen)         50                         ;# max packet in ifq
set opt(nn)             15                          ;# number of mobilenodes
set opt(adhocRouting)   DSDV                       ;# routing protocol

set opt(cp)             ""                         ;# connection pattern
file
set opt(sc)     "scen"    ;# node movement file.

set opt(x)      2000                            ;# x coordinate of topology
set opt(y)      2000                            ;# y coordinate of topology
set opt(seed)   0.0                            ;# seed for random number
gen.
set opt(stop)   500                            ;# time to stop simulation

set opt(ftp1-start)      160.0
set opt(ftp2-start)      170.0
set opt(udp-start)       0

set num_wired_nodes      2
set num_bs_nodes         1

#
# ======================================================================
# check for boundary parameters and random seed
if { $opt(x) == 0 || $opt(y) == 0 } {
       puts "No X-Y boundary values given for wireless topology\n"
}
if {$opt(seed) > 0} {
       puts "Seeding Random number generator with $opt(seed)\n"
       ns-random $opt(seed)
}


set ns [new Simulator -multicast on]

$ns node-config -addressType hierarchical
AddrParams set domain_num_ 2           ;# number of domains
lappend cluster_num 2 1                ;# number of clusters in each domain
AddrParams set cluster_num_ $cluster_num
lappend eilastlevel 1 1 [expr $opt(nn) + 1] ;# number of nodes in each
cluster
AddrParams set nodes_num_ $eilastlevel ;# of each domain

set tracefd  [open out.tr w]
set namtrace [open out.nam w]
$ns trace-all $tracefd
$ns namtrace-all-wireless $namtrace $opt(x) $opt(y)

# Create topography object
set topo   [new Topography]

# define topology
$topo load_flatgrid $opt(x) $opt(y)

# create God
create-god $opt(nn)
set god_ [God instance]

#create wired nodes
set temp {0.0.0 0.1.0}        ;# hierarchical addresses for wired domain
```

```
for {set i 0} {$i < $num_wired_nodes} {incr i} {
    set W($i) [$ns node [lindex $temp $i]]
}

# configure for base-station node

$ns node-config -adhocRouting $opt(adhocRouting) \
                -llType $opt(ll) \
                -macType $opt(mac) \
                -ifqType $opt(ifq) \
                -ifqLen $opt(ifqlen) \
                -antType $opt(ant) \
                -propType $opt(prop) \
                -phyType $opt(netif) \
                -channelType $opt(chan) \
              -topoInstance $topo \
                -wiredRouting ON \
              -agentTrace ON \
                -routerTrace OFF \
                -macTrace OFF

$ns color 1 red
# prune/graft packets
$ns color 30 purple
$ns color 31 bisque

set temp {1.0.0 1.0.1 1.0.2 1.0.3 1.0.4 1.0.5 1.0.6 1.0.7 1.0.8 1.0.9 1.0.10
1.0.11 1.0.12 1.0.13 1.0.14 1.0.15 1.0.16 1.0.17 1.0.18 1.0.19 1.0.20}
                                    ;# hier address to be used for wireless
                                    ;# domain
set BS(0) [$ns node [lindex $temp 0]]
$BS(0) random-motion 0              ;# disable random motion

#provide some co-ord (fixed) to base station node
$BS(0) set X_ 1.0
$BS(0) set Y_ 2.0
$BS(0) set Z_ 0.0

#configure for mobilenodes
$ns node-config -wiredRouting OFF

  for {set j 0} {$j < $opt(nn)} {incr j} {
    set node_($j) [ $ns node [lindex $temp \
           [expr $j+1]] ]
    $node_($j) base-station [AddrParams set-hieraddr \
           [$BS(0) node-addr]]
}


$ns duplex-link $W(0) $W(1) 5Mb 2ms DropTail
$ns duplex-link $W(1) $BS(0) 5Mb 2ms DropTail

$ns duplex-link-op $W(0) $W(1) orient down
$ns duplex-link-op $W(1) $BS(0) orient left-down

set mproto DM
set mrthandle [$ns mrtproto $mproto {}]
set group0 [Node allocaddr]

set udp0 [new Agent/UDP]
$ns attach-agent $node_(1) $udp0
$udp0 set dst_addr_ $group0
$udp0 set dst_port_ 0
```

```
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0


set rcvr [new Agent/LossMonitor]

for {set j 0} {$j < $opt(nn)} {incr j} {
    $ns attach-agent $node_($j) $rcvr
    $ns at 1.2 "$node_($j) join-group $rcvr $group0"
}

$ns at 1.0 "$cbr0 start"

$ns at $opt(stop).0002 "puts \"NS EXITING...\" ; $ns halt"
$ns at $opt(stop).0001 "finish"

proc finish {} {
       global ns
       $ns flush-trace

       puts "running nam..."
       exec nam out.nam &
       exit 0
}

$ns run
```

BIOGRAPHICAL SKETCH

Peter Handel was born in 1976 to two professors in St. Louis, MO, where he started using computers at the age of five. He received his bachelor's degree in physics from St. Olaf College, and is currently finishing a master's degree in computer science at the University of Florida Gainesville. He accepted a position as Senior Engineer on the MacOS X project at Apple Computer, Inc., in the summer of 2000.