Theses and Dissertations   1. Thesis and Dissertation Collection, all items

2001-12

Multimedia data capture with multicast dissemination for online distance learning

Collins, Michael Christopher

Monterey, California. Naval Postgraduate School

https://hdl.handle.net/10945/6185

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

MULTIMEDIA DATA CAPTURE WITH MULTICAST DISSEMINATION FOR ONLINE DISTANCE LEARNING

by

Michael Christopher Collins

December 2001

Thesis Advisor:	Don Brutzman
Second Reader:	Don McGregor

Approved for public release; distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** December 2001	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: Multimedia Data Capture with Multicast Dissemination for Distance Learning		**5. FUNDING NUMBERS**	
6. AUTHOR(S) Michael Christopher Collins			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE**	

13. ABSTRACT
 	
    Distance Learning Environments (DLEs) are elusive to define, difficult to successfully implement and costly due to their proprietary nature. With few open-source solutions, organizations are forced to invest large amounts of their resources in the procurement and support of proprietary products. Once an organization has chosen a particular solution, it becomes prohibitively expensive to choose another path later in the development process.
    The resolution to these challenges is realized in the use of open-standards, non-proprietary solutions. This thesis explores the multiple definitions of DLEs, defines metrics of successful implementation and develops open-source solutions for the delivery of multimedia in the Distance Learning Environment. Through the use of the Java Media Framework API, multiple tools are created to increase the transmission, capture and availability of multimedia content. Development of this technology, through the use of case studies, leaves a legacy of lectures and knowledge on the Internet to entertain and enlighten future generations.			

    Distance Learning Environments (DLEs) are elusive to define, difficult to successfully implement and costly due to their proprietary nature. With few open-source solutions, organizations are forced to invest large amounts of their resources in the procurement and support of proprietary products. Once an organization has chosen a particular solution, it becomes prohibitively expensive to choose another path later in the development process.
    The resolution to these challenges is realized in the use of open-standards, non-proprietary solutions. This thesis explores the multiple definitions of DLEs, defines metrics of successful implementation and develops open-source solutions for the delivery of multimedia in the Distance Learning Environment. Through the use of the Java Media Framework API, multiple tools are created to increase the transmission, capture and availability of multimedia content. Development of this technology, through the use of case studies, leaves a legacy of lectures and knowledge on the Internet to entertain and enlighten future generations.

14. SUBJECT TERMS Multimedia, Multicast, Streaming Media, Real Server, Windows 2000, JMF, Java, Media Player, Real Player, Video Capture, Video-On-Demand, Web Learning, Digitalization, Distance Learning, Asynchronous Education			**15. NUMBER OF PAGES**
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

i

THIS PAGE INTENTIONALLY LEFT BLANK

MULTIMEDIA DATA CAPTURE WITH MULTICAST DISEMINATION FOR ONLINE DISTANCE LEARNING

Michael C. Collins
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1991
M.B.A., University of West Florida, 1992

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE
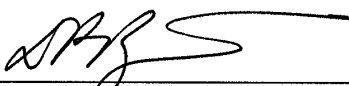IN
MODELING, VIRTUAL ENVIRONMENTS AND SIMULATION (MOVES)
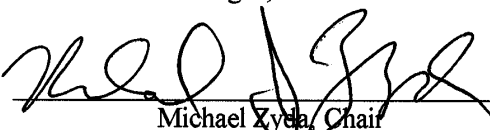
from the

NAVAL POSTGRADUATE SCHOOL

December 2001

Author: _____
Michael C. Collins

Approved by: _____
Don Brutzman, Thesis Advisor

_____
Don McGregor, Second Reader

_____
Michael Zyda, Chair
Modeling, Virtual Environments and Simulation (MOVES)

iii

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Distance Learning Environments (DLEs) are elusive to define, difficult to successfully implement and costly due to their proprietary nature. With few open-source solutions, organizations are forced to invest large amounts of their resources in the procurement and support of proprietary products. Once an organization has chosen a particular solution, it becomes prohibitively expensive to choose another path later in the development process.

The resolution to these challenges is realized in the use of open-standards, non-proprietary solutions. This thesis explores the multiple definitions of DLEs, defines metrics of successful implementation and develops open-source solutions for the delivery of multimedia in the Distance Learning Environment. Through the use of the Java Media Framework API, multiple tools are created to increase the transmission, capture and availability of multimedia content. Development of this technology, through the use of case studies, leaves a legacy of lectures and knowledge on the Internet to entertain and enlighten future generations.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. PURPOSE

The purpose of this work is to describe the current state of Distributed Learning Environments (DLE), identify key technologies involved in the creation of DLEs and show how the diverse fields in this arena may be unified for a synergistic effect. This synergy allows a new training paradigm to be realized for organizations such as universities, corporations and the military. Specifically, this thesis identifies and develops open-format, standards-based Distributed Learning solutions and shows their usability through case study production of multiple course archives.

B. MOTIVATION

Current methods of training in large organizations typically do not acknowledge or compensate for a dispersed workforce that must maintain existing production levels.. Unfortunately, most work tends to preclude educational opportunities. Current training models that remove personnel from their normal schedules and place them in unfamiliar locations for training or education tend to produce several undesirable results. First, employees need to adjust to the new environment. Second, ongoing work concerns do not diminish and are still prominent in the student's mind. Finally, work stoppage or hindrance may unsettle an organization to the point of negatively labeling these educational opportunities.

One method for lessening these challenges is to create a *synchronous* Distance Learning Environment. Many members from a geographically dispersed group can participate from remote locations as a collective body. Such groups benefit from real-time feedback from a competent instructor as well as the knowledge of the group. The group can provide emotional and intellectual support when a person might otherwise feel isolated or unmotivated in the learning process.

A second method for minimizing learning and organizational challenges is to provide an *asynchronous* training environment. Employees may attend training when it is least disruptive, at a time of their own choosing and without coordinating a class time

1

with others. Such an environment is usually produced using Internet-based technologies such as streaming media, media on demand, web pages, e-mail and more. All of these technologies lend themselves to a high degree of automation and standardization if properly employed.

After an extensive review of literature in this arena, there is no single source or reference offering information about a complete and consistent solution. The solution needs to encompass the whole process from data capture to web-based data delivery. Many proprietary concerns offer numerous books and articles related to their specific product. However, there are no compilations that integrate current commercial and open-source solutions to illustrate the life cycle of asynchronous/synchronous distance education, or demonstrate how to optimize the different methodologies for appropriate efforts. This work fills an important gap in the published literature.

C.   OVERVIEW AND THESIS ORGANIZATION

This thesis covers a large number of topics that are diverse but directly interrelated. The chapters that follow explore the current technology for distance learning, how to create content and how to deliver content over the Web. The main thrust is in the asynchronous learning environment, but synchronous learning is also explained where appropriate. This work provides a complete exploration of the entire cycle, from designing software to conducting multiple case studies required for distance learning. It shall utilize real-world situations and examples that require this technology; demonstrating and evaluating how it was integrated.

This thesis is designed as a springboard into a highly complex, cutting-edge area of technology. A multitude of issues and technologies are addressed to enable understanding of this material.   The thesis then takes the concepts and exercises them via a series of increasingly demanding case studies. The final capstone case study places the reader in a comprehensive scenario and an actual production environment using the open-standards multimedia models developed.

A short abstract of each chapter follows.

CHAPTER I – INTRODUCTION: A general overview of the entire paper.

CHAPTER II – DISTANCE LEARNING: The current state of distance learning. Multiple methods of delivering information are explained as well as technologies on the horizon. Integration challenges will be identified with possible solutions explored.

CHAPTER III – VIDEO AND AUDIO CAPTURE: Multiple methods of video and audio capture are utilized in this research. Hardware and software solutions are evaluated regarding their effectiveness and amount of automation. Multiple media formats are examined for situational appropriateness.

CHAPTER IV – JAVA MEDIA FRAMEWORK (JMF): This Java API has the ability to capture and deliver multimedia for a Distance Learning Environment. The API also contains sample source code to jumpstart the creation of highly specialized capture and playback applications. Multiple programs were developed to demonstrate solutions and are provided for further use and development.

CHAPTER V – CONTENT DISTRIBUTION: Multiple methods of content distribution are explored. Several tools for real-time streaming of cached or on-demand video delivery are assessed for strengths and weaknesses. Other tools such as whiteboards and remote desktop control applications are also included in the final solutions.

CHAPTER VI – NPS VIDEO TELECONFERENCE CENTER CASE STUDY: Two classes that were delivered to and from the Naval Postgraduate School using their state of the art distance learning equipment were observed. Numerous technical, administrative and pedagogical problems were encountered and are documented. The lessons learned as well as methods of distance learning education are amplified and categorized.

CHAPTER VII – DR. FRED BROOKS CASE STUDY: The first proof of concept. A fifty-minute video is captured and displayed on the Internet for on-demand delivery. Multiple capture and encoding devices were used to evaluate effectiveness of media formats. Bandwidth considerations were also assessed.

CHAPTER VIII – DR. RICHARD HAMMING CLASS CASE STUDY: This chapter combines the research of the previous case studies and explores the conversion of

a pre-created class, how to package it, and how to deliver it. By focusing on a single lecture series of previously created material, another real world problem is faced. Some of an organization's best training materials may have already been created and merely needs to be transferred to a new media format. Also, by re-using a proven educational package, this case study can concentrate on the technology required for the stated tasks.

CHAPTER IX – SIGGRAPH 2001 CASE STUDY: This chapter addresses the capstone project. All research in this paper led to the capture, conversion and delivery of more than 250 hours of lecture, notes and PowerPoint slides within a one-week period. The magnitude of content easily represents the beginnings of a commercial educational facility and provides enough material for sustained growth.

CHAPTER X – CONCLUSIONS AND RECOMMENDATIONS: After exploring so many hardware and software solutions, it became apparent that there is no single solution for all situations. This chapter presents many of the challenges and solutions that were utilized. As comprehensive as this research has been, it has revealed more areas of future research than it has addressed. Future work is presented by topic.

D. SUMMARY

The work contained in this thesis integrated extensive, inter-related technologies and then took a giant leap to explore the complete process of creating a multimedia Distance Learning Environment. Multiple metrics are used to address the cost involved in producing such environments. Costs affect not only monetary contributions but manpower and intellectual skill as well. Finally, when synergy was identified in connecting different hardware and software solutions, each specific technical avenue was pursued to completion. Typically, open-source software tools and the Java API can be utilized to create a viable solution. The Java code produced by this thesis is all open-source and can be extended to meet future, specialized challenges. Many code snippets are used for clarification throughout the paper while the appendixes contain full source code for several capture and playback programs. Use of multimedia tools, such as Adobe Premiere and JMStudio, are also explained in the appendixes. This thesis lays the foundation for a complete distance-learning solution that can significantly benefit any organization attempting to educate or train.

II.   DISTRIBUTED LEARNING ENVIRONMENTS

A.   INTRODUCTION

This chapter presents a background on Distance Learning Environments and describes their characteristics. Additionally, the differences between *synchronous* and *asynchronous* education are explored. The challenges and benefits of the two environments are discussed. Proprietary and open-source solutions are then compared and contrasted. Finally, trends in distance education are extrapolated from current organizational trends and projected into the future.

B   BACKGROUND

There is no typical Distributed Learning Environment (DLE). DLEs span the spectrum from lecture notes that remain on a web site to fully functioning teleconferencing centers. Some class content is developed as a grassroots movement while others are part of a carefully crafted corporate image. The developers of the courses' contents range from students to instructors to professional distance learning consultants. One of the reasons why it is difficult to quantify a Distance Learning Environment is that there are so many genres of course presentation that qualify under the title. The United States Navy (USN)'s Office of Training Technology has stated the following about Distance or Distributed Learning:

> "The Department of Defense (DOD) is changing its approach to training. It is moving away from training large groups of individuals in formal residential settings because of high travel costs and facility operating and maintenance costs. Distributed or distance learning is also replacing correspondence courses when instructors and students are geographically separated. The primary objective of distributed learning is to extend the learning environment to students at remote locations." [United States Navy 01]

Unfortunately, this definition does not cover the full system required for a DLE. Foremost, the definition of distance learning must be expressed. Placed in its simplest terms, the primary source of instruction or facilitation is spatially and/or temporally removed from the receiver of the information. Hence, a Distance Learning Environment is an environment that allows information to be passed from one locale to another using

real-time or post-time instruction. Therefore, the definition of a DLE must include the mechanisms for creating, sending and receiving information across the geographic and temporal distances. A more robust definition of a DLE, as stated by the author, is as follows:

> "A Distributed Learning Environment is a system that facilitates the transference of instruction, knowledge or training from one geographic locale to one or more remote locations. The scheme removes space and time constraints previously imposed by traditional education. The system must also include the means of content creation, delivery, storage and reception. Furthermore, there must be a feedback loop between the student and instructor to retain the benefits of current educational systems."

The main points of this definition can be amplified when analyzed on a point-by-point basis. First, there must be a geographic and/or time difference between an instructor and student. The classic example is when a class is presented from a central campus and observed by several satellite classrooms in real-time. Normally, such lectures are videotaped to allow review at a later time. Even if some students are in the same classroom as the instructor, the people in the remote locations are still participating in a distance learning experience.

Second, there must be an attempt to transfer information. The attempt does not have to be successful, but it must still be the driving purpose of the environment. The focus of learning is what distinguishes this environment from online games and entertainment chat rooms. The instruction can be presented in one or more media usable by the receiver. The ability to simultaneously and effectively employ many custom tools for instruction can amplify or destroy a learning experience.

Third, the removal of traditional classroom constraints is a factor. If the limitations of the classic learning environment are not removed or relaxed, there is no reason to pursue the costs associated with distance education. The cost of distance education can be measured in both money and manpower. Current tools include Internet connections, video equipment, recording suites and computers. The new environment also increases the instructor's preparation time. Instructors need to understand their content and how it will be delivered using the distance learning equipment. Anecdotally,

the initial workload in preparing for a distance learning class is almost doubled over the traditional classroom. This phenomenon was observed during the NPS Teleconference Case Study later in this thesis. Nonetheless, where most classroom instruction is ephemeral and only presented to attending students, a properly recorded distance learning session may be presented to multiple audiences.

Fourth, multiple technologies must be considered as part of the process. The process begins with content capture. Devices such as word processors, e-mail clients, audio recorders, video recorders and teleconferencing software can be employed to prime the distant-learning environment. The content is then delivered to the receiver immediately or with a time delay. If there is a planned time delay, storage device archives are employed for later retrieval by recipients. Immediate delivery implies the use of network delivery systems. A standard feature of today's distance learning classrooms is Internet connection for the transference of data.

The student must then have the means to receive the information. In a campus environment, remote "satellite" classrooms are usually networked to the central classroom using a high-speed network. Remote students may have to rely on cable modems, digital subscriber lines, slower analog modems or postal service. The speed of their connection has profound effects upon the number and type of instructional tools that may be utilized.

Finally, feedback is required for both the instructor and the students. The instructor may need to check for comprehension and the student may need clarification. The feedback can be as simple as an e-mail response to a question or as complex as an oral defense before a geographically distributed review board. Also, many nonverbal indicators of understanding, which are taken for granted in a traditional classroom, are highly filtered or not present in a distance environment. New cues and indicators regarding student comprehension must be developed and employed in the new classroom environment.

C.   SYNCHRONOUS LEARNING

The Distance Learning Environment can be divided into two classes based upon the timing of the presentation: *synchronous* and *asynchronous*. The *synchronous* learning

environment is created when instructors and students interact via a real-time connection. The capability of instant question recognition and feedback make this environment a close model to the interaction of a traditional classroom.

Audio and video channels are transferred to students in a perceived real-time instruction mode. If the student does not understand the presented material, they may gain the instructors attention and ask for amplifying information. If the teacher perceives that a student does not comprehend the presentation, they may ask leading questions or offer amplifying information. By noticing one student's difficulties, the instructor may actually be addressing the concerns of several students within the classroom.

D.   ASYNCHRONOUS LEARNING

The second classification of education is defined by the attribute of delayed delivery of instruction. The use of Video-On-Demand (VOD) schemes can be realized in the form of digitized instruction available on Web-enabled servers or the creation of a videotape library. Correspondence classes may also fall within this classification.

This category of instruction may become one of the most difficult to deliver due to the inability to offer timely correction of course content. The teacher must insure that all content is correct and not diminished by mistakes in presentation or concept. If a mistake is made, the damage may last until the next method of correspondence is available. In some cases, the only recourse for a few poorly spoken words is to recreate the entire lecture for a new capture session.

Also, as is the case with synchronous learning, the support staff involved in asynchronous learning increases according to the complexity of the tools utilized. Lessons from the television industry are directly transferable and show the need for camera personnel, directors, hardware technicians, content or script creators, and presenters. The teacher becomes the center of his or her impromptu television show. While several people have been able to assume all of these roles for their class, a saturation point is eventually reached where the one person cannot perform all of the aforementioned tasks simultaneously and successfully.

E.   COMMERCIAL SOLUTIONS

Many organizations have adopted commercial solutions for their distance learning needs. The needs of an institution can be directly reflected in its size and organization. Common sense dictates that the larger the organization, the more requirements will be placed upon the system by distance education. The demand placed upon a system is in both quantity and complexity. A small organization may only need periodic training for a select audience. A large organization is usually faced with a large number of audiences that must learn numerous and disperse volumes of knowledge. The medium organization falls between the two extremes; occasionally with high demand and insufficient resources.

1.   Small Organization

Every organization has the ability to create or rent a Distance Learning Environment. Companies such as Kinkos have led the field with teleconferencing centers in several major cities. A small organization can rent facilities in cities where its organization exists and conduct remote training. This may be one of the most costly solutions, but is still a possibility for a company that may not have the budget or expertise to provide their own services.

Interestingly, many computer systems are already equipped with the software necessary for a Distance Learning Environment. A small investment in training or reading product documentation can allow a small organization to capitalize on its current investments by using them to set up a DLE. For example, Microsoft Office's PowerPoint has a built in capability to combine slides with video and voice. Hence, notes as well as the presenter can be streamed to a geographically dispersed organization requiring training. Multicast technologies can also be utilized so that in multicast-enabled networks, bandwidth costs can be reduced.

There are also free software applications that can stream content over an existing local network infrastructure as well as modify the media format used. Most of these applications were developed to support the sales of other applications or products

available from the parent company. One such application in Microsoft's Media Encoder that is available at http://www.microsoft.com/windows/windowsmedia/en/download/.

This encoder has many advantages and disadvantages. First, it is free to use, but the only formats that it creates are ones solely compatible with Microsoft's products. The encoder can convert any media format from QuickTime or MPEG, but only to ASF or WMV capable of streaming by another Microsoft product. If an organization is already using these products such as Microsoft's Media Player, there are no short-term concerns. In fact, an organization that already uses these products may have an advantage since the client software is already installed on all of the machines within the organization.

Second, the encoder can be used on any current Microsoft Operating System to stream content. There is no need to gain the added expense of streaming servers when a Windows 98 (or later) machine can be utilized to stream and archive a video session at the same time. If the files are shared on a network drive, all members of the small organization can have access to the sessions in an asynchronous manner.

2.   Medium Organization

Many companies are now starting to specialize in consulting and creating Distance Learning Environments. These companies can scale their products from recommendations to full solutions of capture, transcoding, generation of multiple display methods, language translation, and delivery. For example, MentorWare Inc. was contracted at the JavaOne 2001 convention by Sun Microsystems to provide software for content capture and delivery of all sessions. The solution was limited to voice, text, and slides in Macromedia's proprietary Flash format due to bandwidth and storage considerations.

Despite format impediments, the JavaOne case is where a large organization scaled commercial services to the exact level desired. In this case, even though Sun is number 125 on the Fortune 500 with close to 16 billion dollars in revenue last year, it still used a solution that might be used by a medium-sized organization. By combining company personnel with trained distance education consultants, Sun was able to create a custom solution that allowed state of art technology to be utilized by in-house talent.

3.   Large Organization

The budget and requirements of a large organization may allow it to hire a full turnkey solution. One such solution is IBM's Learning Services, which provides Performance Consultants, Curriculum Designers, Content Developers, E-Learning Technologists, Instruction Specialists and Mentoring Specialists. In summary: a company provides the event or even the subject upon which it desires training and IBM will provide all of the rest. This method frees company staff from working outside their area of expertise and maximizes outsourced work.

The nature of the company may allow it to develop its own Distance Learning Environment. Capture and server technology from companies such as Microsoft, Real Networks, CISCO, 2NetFx and Streaming21 can provide a robust solution for content capture, creation and delivery. The company may already have several high-speed network connections between its disperse work force or purchase increased access for the added content. This company may even have a department dedicated to multimedia capture and network delivery. The department might merely hire more personnel to meet the higher workload.

F.   OPEN-SOURCE SOLUTIONS

Open-source solutions for distance education have been limited in implementation and scope. Beyond the initial MBone tool set, developed several years ago, there has not been any substantial pursuit of improving the tools or creating new tools until recently. One technology, which offers great promise in this arena, however, is the Java Media Frameworks (JMF) API. The JMF set of classes allows for capture, display, transcoding, special effects and more. JMF is explored extensively in Chapter IV. Frequently asked questions about the MBone can be found at
http://www.cs.columbia.edu/~hgs/internet/mbone-faq.html.

1.   Open-source Tools

Currently, some colleges are experimenting with open-source tools such as the MBone Tool Set. The MBone Tool Set consists of several applications capable of creating and connecting to multimedia sessions on the MBone portion of the Internet.

University College London (UCL) and several other research universities with several grants from DARPA, EPSRC and EC created the applications.   All tools are open-source and also available in precompiled formats for various operating systems at http://www-mice.cs.ucl.ac.uk/multimedia/software/.

a.   *Session Directory (SDR) Tool*

The tools are designed to meet a number of the Distributed Learning Environment's needs. First, session advertisements are created and found on the Session Directory (SDR) application. Using this tool, two-way through n-way conference sessions are made available to the entire audience of SDR listeners. The only requirement for a user to view the available programs and conferences is to download the tools, install the tools, and connect to the MBone. The MBone is the multicast enabled portion of the Internet. Since there are many routers on the Internet that do not forward multicast packets, the MBone can be referred to as a virtual network within the larger network known as the Internet.

After checking a well-known multicast group for a list of available multicast addresses, each session is created randomly on a different but free multicast channel. The use of multicast addresses solves and creates many problems. Multicast solves limitations in current computer networks. First, if the session is made available to many users, bandwidth and computational constraints prevent the number of users from scaling in a unicast environment; a problem multicast does not suffer. Also, as soon as session clients connect from remote networks, broadcast schemes are no longer feasible.

However, the primary problem limiting a multicast-enabled system is the need for MBone connectivity. Many Internet Service Providers (ISPs) are not comfortable with the burden of adding multicast support to their routers. Until a sizeable portion of their client's request or demand multicast-enabled routes, there are few reasons for the service providers to allocate resources within their network. Fortunately, service providers are beginning to see the added advantage of how multicast, when properly employed, can decrease their overall network traffic. Until the day arrives when the entire Internet is multicast enabled, user networks can continue to tunnel their multicast packets through the non-multicast networks with the use of mrouters and wrappers.

12

b. *Video Conferencing (VIC) Tool*

The Video Conferencing Application is designed to work in conjunction with the Session Directory Application by using the information supplied when a person selects a session. SDR passes the address of the application's video address to VIC so that the application can begin to intercept and display the pictures streaming on the channel. Once a session is established, a client is also capable of broadcasting their own video stream so that other subscribers can see who is using the channel. In this manner, subscribers can evaluate the content of others with the extra information of facial expressions and hand gestures.

c. *Robust Audio Tool (RAT)*

Another tool, also launched from the SDR, is the Robust Audio Tool (RAT). RAT is capable of capturing and presenting audio channels from the multiple sessions selected using SDR. All users connected to a particular session can receive and send sound to all others connected to the session. By having sound separated from video and sending separated from receiving, the user has the maximum number of possible combinations available to customize their session. For instance, an instructor can set a policy requiring all students to have their microphones muted unless they have a question to ask.

The benefits of using audio channels are numerous. Since speech is a natural method of communication, most people can quickly and easily transfer knowledge using audio contact. Sound can also be used to establish a sense of presence in a Distance Learning Environment. By using ambient noise and amplifying sounds, a student can gain a greater sense of presence than just by the visual channel. Many studies have been conducted which indicate that sound accounts for approximately 20 percent of a person's reality [Heilig 92].

d. *White Board (WB) Tool*

The White Board (WB) is a two-dimensional graphics program that allows the members of a session to transfer images that are created using a computer mouse. It allows for the use of multiple colors so that differences may be easily pointed out.

Different users can have an assigned color to distinguish their comments and drawings. One session member may draw objects and another may use arrows or circles to emphasize a particular point.

The WB application also fulfills the need for situations requiring rapid, rough diagram prototyping. While it is possible to send finished bitmap images that may be more refined, the package does not contain an open-source tool that can be used for the creation of slides or presentation quality images. This lack of a more sophisticated drawing tool hinders the use of professionally created slides and does not capitalize on the number of presentations already available in HTML or created with Microsoft Office or Sun's Star Office products.

e. Networked Text Editor (NTE) Tool

Networked Text Editor is the equivalent of a chat room. While people are listening or watching a presentation, they can participate in written conversation with others in the session. This application allows for the transference of ideas without interrupting the pacing and delivery of someone speaking. By using the NTE, multiple conversations can take place in the background while the primary speaker continues in their delivery. This method closely mimics the dynamics that actually occur in a traditional classroom environment.

The NTE application has also been used in new and innovative ways. A problem with Distance Learning Environments is the poor ability to interject questions and comments without interrupting the entire class. By using the NTE, instructors have created policies stating they will only answer questions that are written in the shared text editor. In this manner, the instructor does not stop lecturing until they are finished their presentation and a written record is maintained of all questions asked. The written record allows other instructors to be exposed to possible sources of confusion. Unfortunately, this policy does not allow students to stop and ask for an explanation of a topic while the instructor is in the midst of presentation. The rest of the presentation can be useless if the student does not understand a fundamental concept. Also, if the instructor accidentally "mis-speaks", students can be extremely confused or believe a wrong fact to be true.

14

2. Open-Source Delivery

One of the largest open delivery projects is the MIT OpenCourseWare (MIT OCW) Project. MIT plans to make all of its courses, lecture notes, videotapes, course outlines, assignments, reading lists and more available on the World Wide Web free of charge. As the technology of the classroom becomes more sophisticated, the planned MIT OCW delivery mechanisms should be flexible enough to adjust to the new media. MIT plans to begin releasing courses in Fall 2002. The initial delivery is estimated to include over 500 courses with a planned warehouse of over 2000 courses within ten years. The project is estimated to cost 7.5 to 10 million dollars a year for the first ten years to convert, create and deliver content. While private donations will fund the project, the content will be free for anyone who has access to the Internet. [MIT 01]

The tools utilized to create and convert the MIT content have not been revealed. However, the content of several courses are already available at MIT's web site. For example, Dr. Gilbert Strang's Linear Algebra Class was taped in the fall of 1999. There are thirty-three one-hour digitized video taped lectures with class notes, syllabus and assignments currently available. From observing the videos, it is likely there was a dedicated cameraman that was probably a Teacher Assistant (TA). The tapes were digitized in several of Real Audio's streaming media formats. If this class is an example of how the project shall proceed, MIT has decided to use commercially available and proven products to create their free courseware.

G. FUTURE TRENDS

1. Interest in Distance Education

The interest in distance learning has become immense. Almost every large organization is pursuing the means of delivering distance education. The United States Department of Education estimates that the current number of 8.3 million college students shall grow to 9.6 million within eight years; a 15 percent growth.

There are three solutions for colleges to meet this projected increase in demand. First, colleges and universities can begin large-scale construction of new facilities. This is by far the most costly solution since multi-million dollar facilities are usually

supported by endowments and increased tuition. Second, class size can be increased. Since some institutions, especially state universities, already have some classes that approach several hundred students, this method may simply be absorbed as an acceptable cost of business. However, the value gained from these large, auditorium-style classes may need to be evaluated. Finally, many of these institutions are creating Distance Learning Environments where disperse facilities are optimized. The DLEs also allow students with restricted schedules to attend class from work or home. The time saved in commuting as well as the added income can be invested in the educational experience.

2.   Content Variety

Content and delivery methodology are greatly increased using the technology associated with DLEs. First, asynchronous capabilities give students the ability to gain knowledge from an educational experience when their schedule allows. The added flexibility opens the instruction to a larger audience not normally able to attend. There is also a trend for certification training to begin offering on-line training since the instructors certified to teach these courses are usually restricted to larger cities and for limited engagements. The added cost of transporting an instructor for local training can grow quickly. For example, one firm in San Jose, California charges $1800 dollars per student for a one-week course on their premises and they provide all equipment. The same organization charges a minimum of $25,000 for on-site training and class size is limited to ten students. Also, the hiring organization is responsible for providing equipment where rental fees can be several thousand dollars. Hence, the price is almost doubled for on-site training.

The decrease in cost and increase in government sponsorship of Internet-based technologies has prompted some sociologists to project that the traditional classroom will be reserved for the wealthy and represent an economic class division. Some studies project that the wealthy will be the only ones willing to pay the added expenses of dormitories and transportation [Hamilton 01]. While this concept has a certain ring of truth, there will always be a need for traditional as well as distance education. In the end, the litmus test may be which method of learning works best for the individual. The

sociologists may have overlooked the value of ritual and the added enforcement of physical presence that is still not developed through distance education.

Second, asynchronous capabilities allow students to review class material at times when it is beneficial to them. For instance, if a concept is not clear, they might review right before an exam. If they are about to interview for new employment, they can refresh themselves with material that may be directly related to the new opportunity. Also, some organizations such as Heald College of San Francisco, California offer "lifetime" review policies for their students. The DLE adds minimal additional expense to the presenter but allows this policy to be honored.

Third, organizations create a level of credibility not previously possible by allowing inspectors and accreditation committees' access to classroom content without having to interrupt a classroom environment. The Heisenburg Uncertainty Principle states that merely observing something changes its nature. If the classroom environment does not have to change, the inspectors can see how the events occurred without their presence.

All of these factors lead to several conclusions about the future of education. Distance education will grow without limits based upon the factors previously explained. Due to the increased use of distance education, instructors and students will begin to create and expect a certain level of technology utilization. As the technology becomes more advanced, the distance education shall likewise become more robust and productive. Some of these technologies will be richer multimedia content, provide increased bandwidth for more data transference in less time, customized content, expert and exemplar delivery of material, and more.

A multitude of technological capabilities shall combine to make this possible. Moore's law states that computational power in microprocessors will double every eighteen months to two years. Companies such as AT&T Broadband and Qwest are reported to have less than 40 percent utilization of their networks [Heinzl 01]. The United States has experienced a doubling of its fiber optic bandwidth capabilities every eighteen months. New media formats allow streaming of video such as high definition television at record compression ratios. 2NetFx has servers commercially available that

can transmit 480p, 720p and 1080i HDTV formats at less than 40Mb/second. Other companies such as Microsoft are Internet-enabling many of their productivity tools to allow seamless integration in a Distance Learning Environment. The combination of all of these technologies with the current trend in decreasing cost should create a rich and powerful Distance Learning Environment that will be revolutionary in nature.

Some areas likely to prove revolutionary are three-dimensional (3D) graphics modeling, simulation, data warehousing and augmented learning. Some problems, especially in chemistry, engineering and physics are difficult to represent in a two dimensional display. The power of 3D graphics allows computationally intensive images to be displayed with realistic physical properties. The advanced graphics also allows for visualization of large-scale simulations. The results from these simulations as well as a multitude of information can be kept in large data-warehouse facilities for later retrieval through data mining.   Finally, artificial intelligence (AI) algorithms are being utilized more regularly on multiple Web-based structured data types.

The day may come where the student's computer is an active participant in discussion with other computers and users working on the same topic. A recent Hewlet-Packard report estimates that only ten percent of Internet traffic is peer-to-peer machine communications. They estimate that in five years, ninety percent of all Internet traffic will be peer-to-peer machine communication. Since it has already been proposed that distance learning shall utilize the latest technologies, a logical conclusion is that software agents shall work on behalf of students and instructors. With large, centralized data centers, the agents shall have a plethora of data points upon which to draw and store their conclusions.

H.   SUMMARY

The state of Distance Education (DE) is poised for expansive and profound growth. While many still criticize the quality and quantity of DE, there can be no doubt that more courses will be offered and attended in the future. With a projected revenue stream of two billion dollars in the next five years in the United States, organizations are pursuing multiple avenues to make Distance Education a reality. The large influxes of

money combined with multiple emerging technologies guarantee the increase and augmentation of Distance Learning Environments.

THIS PAGE INTENTIONALLY LEFT BLANK

III.   VIDEO AND AUDIO CAPTURE

A.   INTRODUCTION

The first, and perhaps most challenging, step in the creation of a multimedia Distance Learning Environment is the capture of video and audio presentations. This chapter explores the hardware that allows the digitalization to occur, compressed media formats that can be utilized to digitized video and audio productions, and additional software that can be used to perform transcoding. The hardware is considered via three different categories: low end, high end and specialized. While many of the specialized cards might be considered high end, they deserve their own category since the cards typically digitize content into a single set of proprietary formats. Numerous compressed media formats have been available for many years, while some of the newer formats have only emerged in the last year. Finally, with the increase in personal computer speed, conversions that were previously only accomplished with hardware can now be performed with software solutions.

B.   HARDWARE

Through the years, a multitude of capture devices have been developed to store audio and video images. From its start in film to the modern day Charge Coupled-Devices (CCD), image capture has become a highly standardized process. Sound capture has also enjoyed the benefits of standardization.   With the recent and continuing maturation of these technologies, costs are dramatically decreased and quality continues to increase.

1.   Capture Devices

a.   *Video*

Currently there are two common methods for capturing video. One is through the use of film and the other is through the use of video signals. Film has been the choice of moviemakers and students for the last sixty years. Film is merely a strip of specially treated material that has a specific set of reactions when exposed to light. Light exposure is controlled through the use of a shutter that receives light through a lens. The

lens can adjust focal points and modify the final image that is displayed upon the film. Depending upon the film in use, the shutter is adjusted to match the number of exposures per second that is required by the film format. The focus lens is matched to the size of the film. For instance, a popular format for the home user was Super 8mm film that was exposed at 16 frames per second. Once exposed, the film needs to be developed in a darkroom and then displayed through a projector. The image cannot be transmitted through a digital format without undergoing a separate digitalization process.

The television industry overcame this limitation with the use of cameras that utilized video tubes connected to a matrix of light sensitive particles. As varying levels of light hit the matrix, the tube amplified the signal to a receiver. The signal was then broadcasted through the airwaves. In the early sixties, the tube was replaced in video applications by semiconductor Charge Coupled-Devices (CCDs). Although some television cameras still use the tube technology, the majority of the home and low-end video production cameras use CCD technology. Also, many high-end HDTV cameras now use CCD technology since film for the larger format (such as 65mm IMAX film) can become an unreasonable burden due to size and weight.

CCD devices can be broken into two categories: analog and digital. The analog devices save the CCD images in an encoded format on a tape as an analog magnetic signature. Standard magnetic media are in the form of 8mm-wide tape. When the video is copied to another tape such as a VCR tape, the magnetic signature must be read and placed in a usable format. The signal is then transferred through cables such as RCA connectors and then placed upon the VCR's tape in its magnetic format. The fact that the signal must be read from a tape, transferred using an electrical analog signal and written to a magnetic tape creates what is referred to as "Generation Loss". This means that the copy is not as good as the original. Typically, only two or three generations of copying can be sustained before video-quality reduction becomes unacceptable.
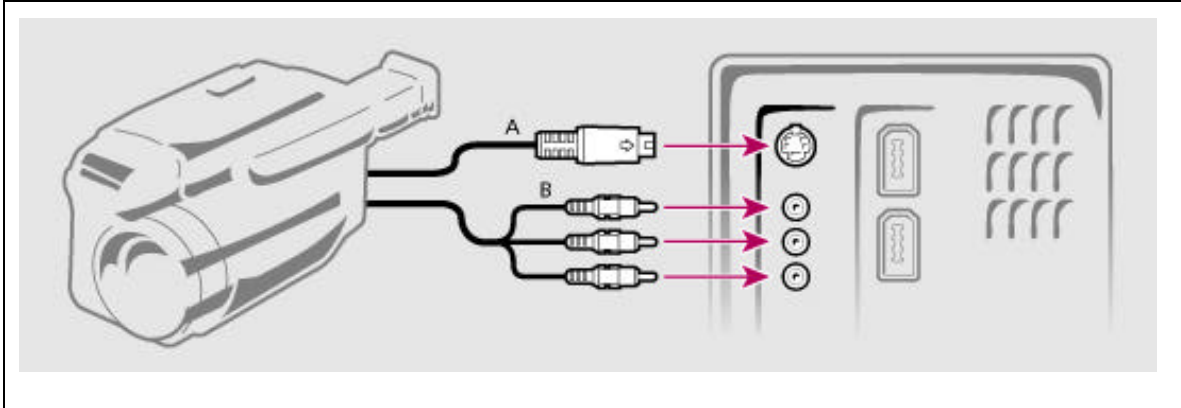
Figure III-1:   Video Connectors [From: Adobe Premier 6.0]

The second method of video-signal capture is through digital means. Instead of storing images as an ongoing analog signal of magnetic patterns, a digital camera has the ability to store the information related to video as sets of ones and zeros. The camera then has the means of transferring the information to another device as ones and zeros. There is no loss in fidelity and the copy is as good as the original. Newer devices contain connectors such as IEEE 1394 FireWire. A computer or encoder box can take the signal through a FireWire port and manipulate or store the signal in a purely digital format.

Format	Lines of Resolution	Tape Size	Media
Standard VHS	230-250	Large	VHS Tape
VHS-C	230-250	Medium	VHS Tape in Small Housing
Super VHS	380-400	Large	SVHS Tape
Super VHS-C	380-400	Medium	SVHS Tape in Small Housing
8mm	230-250	Small	8mm Tape
Hi8	400	Small	Hi 8mm Tape
Digital Video(DV)	500	Extra Small	DV-Mini Cassette

Table III-1:   Video Tape Formats.

Two standards for video signal transmission are RGB and YUV. There are a multitude of formats that are variations of these two types but the principles remain

23

the same. In RGB, each pixel is assigned a value that represents the Red, Green and Blue intensities. Sometime formats include the extra alpha channel and are referred to as RGBA. The alpha channel can depict a fourth component of transparency. When three bytes are used to represent RGB information, it is referred to as 24-bit color. CCD cameras, digitizers, scanners and computers all begin with a RGB format. The images generated from these devices may remain in RGB format for transfer or converted to a form of YUV for signal transmission. There are also many compression schemes that modify the RGB information and are covered in Section B in this chapter.

YUV is based upon on signal that carries Luminance information and a signal that shows the color differences. Luminance is the black and white, or brightness, part of a component video signal. It is also referred to as the "Y" signal. Chrominance is the signal that carries the color information in video. U and V represent color, which are equal to R-Y and B-Y or Cr & Cb (Chrominance Red and Chrominance Blue). A realistic representation can be obtained without direct transmission of the Green color band and the savings in bandwidth makes this a popular compression scheme. Additionally, by separating luminance and color intensity, compression schemes can dramatically decrease the size of transmitted and stored signals.

b.   Audio

A microphone is used to capture analog sound waves and fed through a sound-digitizing card for computer processing. One of the most commonly used types of microphone is the charged capacitor diaphragm. When the flexible diaphragm within the microphone is moved by sound or air pressure, it completes a capacitor circuit to allow the flow of electrons in the opposite direction of normal flow. The reverse flow in electrons can be sensed as a Pulse Code Modulation (PCM) signal. The PCM relies upon three distinct variables of sampling size, storage unit and number of channels.

The sampling size is the number of data points created for each analog wave. For example, the analog waveform of sound is represented digitally by sampling different points along the wave to give digital representation. The more data points that are created, the better represented are the waveforms. If wave amplitude is needed between two points, the algorithm interpolates the desired value. The less distance

between the samples, more granularity, the better representation the wave will have from interpolation. Compact Disk (CD) music, also known as high fidelity, utilizes a 44KHz-digitizing scheme or 44,000 data points for each second of analog sound wave. In contrast, human speech is normally represented over telephone circuits utilizing 8KHz since the human voice remains intelligible within a 4KHz spectrum. Nyquist's theorem shows that a sound must be sampled at twice its reproducible frequency range to be adequately represented.

The storage size is the signal strength of each data point. The number of bits used to represent a signal point defines the fidelity of that point. If one bit is used, there can only be a representation of on or off, signal or no signal. With sixteen bits, 2^{16} or approximately 65,000 distinct levels can be represented by each signal value. High fidelity sound may utilize 24 bits or more for proper representation. 8 bits (i.e. 2^8 or 256 distinct levels) can adequately represent speech.

Finally, the number of channels plays a role in the total size of the PCM signal. Each channel directly increases the size of the analog representation. A monophonic signal consists of only one channel. Stereophonic systems incorporate two signals and can be rendered with one channel per speaker in a two-speaker configuration. Surround sound, or Dolby Digital 5.1, is designed for a five-speaker representation where one speaker is utilized as a subwoofer. Some IMAX films, extending the 5.1 approach, include 18.2 sound for even finer granularity of the sound-source location. However, for many applications, a single or dual channel is more than adequate.

2. Multimedia Ports

There are a multitude of audio and video sources that may be accessed by a computer. Most computers contain sound cards that have standard connectors for a microphone and line-in ports. In a Windows Operating System, these two ports can be accessed by the sound properties application available under the Control Panel or Task Tray. The sound panel that is displayed controls sound rendering and recording. The volume levels for each input and output port can be adjusted using this application. Also, the panel allows an operator to choose which input or output port to use. Some sound

cards contain additional ports such as Midi, CD digital, and computer-generated sound. The play and record panel allows the choice of which port to use.
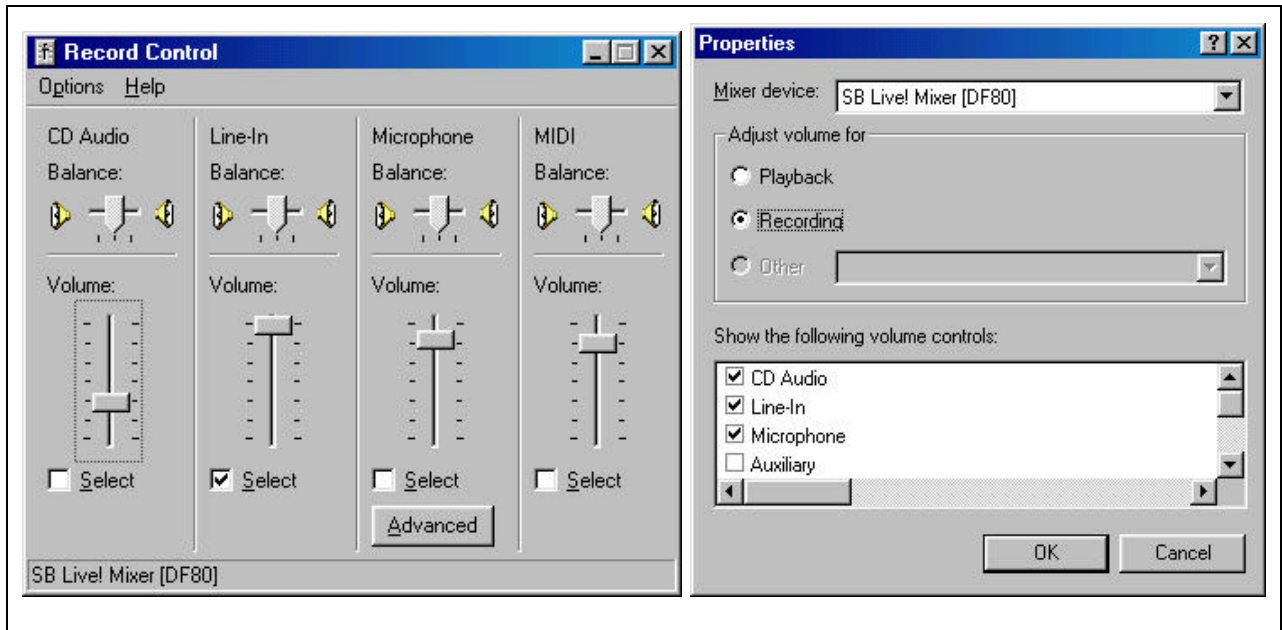
Figure III-2:   Windows Audio Control Properties.

Standard computers also contain a video display card. Most of these cards do not have the capability to capture video but can only render images to a monitor or television. The ATI All-In-One-Wonder card is an exception since it captures and renders video. However, a video camera such as an USB port Web Camera or video capture card can be connected to allow capturing and rendering of video. A few high-end capture cards also contain their own sound ports and hardware compression support. `.mpeg, .asf, .wmv` and `.ra` are a few of the compression formats supported by video hardware manufacturers.

• ATI Technologies	• Intergraph
• Aztech Labs	• miro Computer Products
• Diamond Multimedia	• Optibase
• Digital Vision	• Optivision
• DPS	• Silicon Valley Technology
• FAST Multimedia	• Viewgraphics, Inc.
• Hauppauge	• Winnov
• Intel	• Xing Technology

Table III-2:   Video Capture Card Companies.

A few common video interfaces are *S-Video*, *Component Video* and *Composite Video*. A S-Video connector has inputs of Y (luminance) and C (chrominance) signals separately to reduce interference between Y and C signals, and to help reproduce less noisy images. Component Video is a video signal in which the Luminance and Chrominance signals are kept separate. This requires a higher bandwidth, but yields a higher-quality picture. In Composite Video the luminance and chrominance signals are combined in an encoder to create the common NTSC, PAL or SECAM video signal allowing economical broadcasting of video.

C.   CODECS

Codec is an acronym that stands for Encoder/Decoder. It is a general name assigned to video and audio data that has been processed through a filter either for storage, transmission or playback. Only when the data is sent through another program filter can it be viewable and audible. The main reason for codecs is that bandwidth is limited and directly proportional to cost. By utilizing the processing power of a computer, bandwidth requirements may be minimized with the use of different codecs.

Unfortunately, many codecs are proprietary and can cost several thousand dollars to use and distribute as part of an application. For instance, the MP3 licensing models

from Thomson Multimedia is quite extensive and precise. Table III-1 describes numerous situations where patented codecs need to be licensed. Since the majority of MP3 players and encoders are based upon these patents, legal action can be taken against unlicensed users of their "psycho audio" model. Currently, no MP3 royalties are required for non-commercial organizations, organizations with revenues less than $100,000 per year, individual not-for-profit audio library use or games distributed in less than 5,000 units [www.mp3licencing.org].

Software Products		
mp3	Decoder	· **US$ 0.75** per unit; or · **US$ 50 000.00** - **US$ 100 000.00** one-time paid-up
	Encoder / Codec	· **US$ 2.50** - **US$ 5.00** per unit
mp3PRO	Decoder	· **US$ 1.25** per unit; or · **US$ 90 000.00** - **US$ 150 000.00** one-time paid-up
	Encoder / Codec	· **US$ 7.50** per unit

Hardware Products	ICs / DSPs	
mp3	Decoder	· **US$ 0.75** per unit (Hardware Products); **US$ 0.50** per unit (ICs / DSPs) · Software (optional): **US$ 100 000.00**
	Encoder / Codec	· **US$ 2.50** - **US$ 5.00** per unit · Software (optional): **US$ 250 000.00**
mp3PRO	Decoder	· **US$ 1.25** per unit (Hardware Products); **US$ 0.90** per unit (ICs / DSPs)
	Encoder / Codec	· **US$ 7.50** per unit
	Upfront payment	· **US$ 500 000.00**

Games	
mp3	· **US$ 2 500.00** per title
mp3PRO	· **US$ 3 750.00** per title

Electronic Music Distribution / Broadcasting / Streaming	
mp3	· **2.0 %** of related revenue
mp3PRO	· **3.0 %** of related revenue

Table III-3:   Fraunhofer and Thomson MP3 Licensing Model [From: www.mp3licencing.org].

While the MP3 model might seem insubstantial for most users, the new video format of MPEG-4 is currently under review for licensing and may be based upon a "per-use" licensing model. This may mean that every video encoded into MPEG-4 format will generate a payment to the patent holders, as opposed to the current model of MP3 where each encoder or decoder application generates royalty payments. Furthermore, a one-time fee can be utilized as well for MP3 usage. The one-time fee allows application developers to pay a single one-time charge and then produce as many encoders and decoders as the company desires without further payment.

According to some companies, MPEG-4 is predicted to be the video equivalent of MP3 in Internet popularity. Fortunately, there are many formats that are already in the public domain and based upon years of research. While they do not always have the compression ratios or fidelity of MP3 or MPEG-4, most formats are highly specialized for the bandwidth and processing power associated with their type of application. Hence, MPEG-4 may not dominate the Internet except in its own specialized applications.

1.   Sound

There are numerous codecs strictly for sound as well as a combination of video and audio tracks. Some of the sound codecs have been available for years while others are newly developed to meet current needs. Some compression schemes use weighted logarithmic scales, predictive algorithms of the next sample point, frequency masking or tricks of human perception. Most sound codecs have an associated file extension. For instance, U-law compression is usually represented as an `.au` file and MPEG Audio Layer 3 is usually saved as a `.mp3` file. However, when these formats are combined with a video format such as MPEG-4, they utilize the video format extension such as `.avi`.

AIFF (.aiff)	**QuickTime (.mov)**
8-bit mono/stereo linear	Audio: 8-bit mono/stereo linear
16-bit mono/stereo linear	Audio: 16-bit mono/stereo linear
G.711 (U-law)	Audio: IMA4 ADPCM
A-law	Audio: G.711 (U-law)
IMA4 ADPCM	Audio: A-law
AVI (.avi)	Audio: GSM mono
Audio: 8-bit mono/stereo linear	**Sun Audio (.au)**
Audio: 16-bit mono/stereo linear	8 bits mono/stereo linear
Audio: DVI ADPCM compressed	Audio: 16-bit mono/stereo linear
Audio: G.711 (U-law)	Audio: G.711 (U-law)
Audio: A-law	Audio: A-law
Audio: GSM mono	**Wave (.wav)**
Audio: ACM	8 bits mono/stereo linear
GSM (.gsm)	Audio: 16-bit mono/stereo linear
GSM mono audio	Audio: G.711 (U-law)
MIDI (.mid)	Audio: A-law
Type 1 & 2 MIDI	GSM mono
MPEG Layer II Audio (.mp2)	DVI ADPCM
MPEG layer 1, 2 audio	MS ADPCM
MPEG Layer III Audio (.mp3)	ACM
MPEG layer 1, 2 or 3 audio	

Table III-4:   Sound Codec File Associations

a. Mu-Law or A-Law

The original file format for Mu-law (also A-law) was `.au`. It concentrated on speech digitalization and relied upon the 8 KHz sampling rate of voice. The original Java specification allowed sound cards to directly play the sounds by

30

looking at its weighted logarithmic scaling of sounds. These algorithms are sometimes referred to as G.711. [DeCarmo 99]

b. TrueSpeech

TrueSpeech was also adopted for voice over telephony. It was often saved under the WAV format for Microsoft and AIFF file format for Apple. It is sometimes referred to as G.723.1 and was heavily adopted by the international telephone community since it works well with voice systems. It is one of the primary algorithms for Voice over IP since it has good audio fidelity and bandwidth capability as low as 5.3 Kbps. [DeCarmo 99]

c. Dolby Digital

Dolby Digital or Audio Code Number 3 (AC-3) was designed to compress up to six channels of audio information. The intent for this compression scheme is to provide a surround-sound experience in home theater. By providing audio for speakers carefully positioned around the listener, a more immersive experience can be gained. There are algorithms capable of better sound fidelity than Dolby Digital, but they cannot provide the multi-channel capability of Dolby Digital. For this reason, it is the format of choice for DVD, satellite and digital cable movies. [DeCarmo 99]

d. Linear Prediction Coder

Linear Prediction Coder (LPC) attempts to utilize mathematical equations to predict the sound of voice. It is often a distorted imitation of the original but understandable. It can be used in low-bandwidth considerations where sound quality is not important so long as the user can decipher the voice. The sound is often referred to as "tin-like" or metallic. [DeCarmo 99]

e. Global System for Mobile Communications

Global System for Mobile Communications (GSM) is an extension to LPC and used extensively in the Cellular Phone Industry. It generates 13.2 Kbps for the standard 8KHz-sampling rate. It provides excellent sound quality but utilizes more computational power than competing codecs. It is a mature technology since mobile phone companies have utilized GSM for many years. [DeCarmo 99]

f. Motion Pictures Expert Group (MPEG) Audio

Motion Pictures Expert Group (MPEG) Audio is best known by its Layer 3 implementation. The popularity of its small bandwidth consumption coupled with acceptable music-sound presentation has made it the primary method of transferring music across the Internet. This codec first masks frequencies are not normally perceptible to the human ear. Further sound samples are reproduced using a Huffman encoding process. Usually, the result is an extremely small file size with pleasing sound fidelity. [www.mpeg.org]

g. Audio File Structure

File storage structure is broken down into chunk-based, track-based, and other formats. Chunk-based formats such as .wva, .avi, .iff and .aiff allow multiple types of media to be stored in the same file. Audio from different sources is allowed to be stored in a single location.

Apple's QuickTime format of MOV is track-based. It allows pointers within the file to point to new tracks. Its popularity stems from the control of starting and stopping layered tracks with precise control. Additionally, QuickTime was the first multimedia format that became popular with networked computers.

Another unique format is the object-based Musical Instrumental Digital Interface (MIDI). MIDI is a method of referencing sounds stored within a sound card's memory. The format merely names the musical device, duration of sound and amplitude of sound. The MIDI table present on each individual computer completes the request. Sound representation from machine to machine may not be consistent since different sound cards have different MIDI tables.

2.   Video

The field of video presentation is expansive and complex. Billion dollar companies are competing daily for the best methods of video display, transmission and digitalization. New codecs influence industries larger than the computer industry by impacting mobile phone communications, satellite television and digital cable television. If an algorithm halves the current bandwidth requirements, satellite television can offer

twice as many channels without the expense of licensing new frequencies or launching new satellites. Cell phone companies can provide video in addition to audio by utilizing the same infrastructure. For these reasons and more, the next section presents video information that is specific to this thesis but easily extrapolated to other realms.

a. Video Presentation

Current methods of video display need to be described before video compression can be explained, defined and discussed. One of the main concepts when explaining video signals is *aspect ratio*. The aspect ratio can be defined as the ratio of the width to the height of the image. Hence, a television screen four feet wide needs to be three feet tall to achieve a 4:3 aspect ratio. This picture size can also be represented with a simple mathematical reduction to 1.33:1 or merely referred to as 1.33.
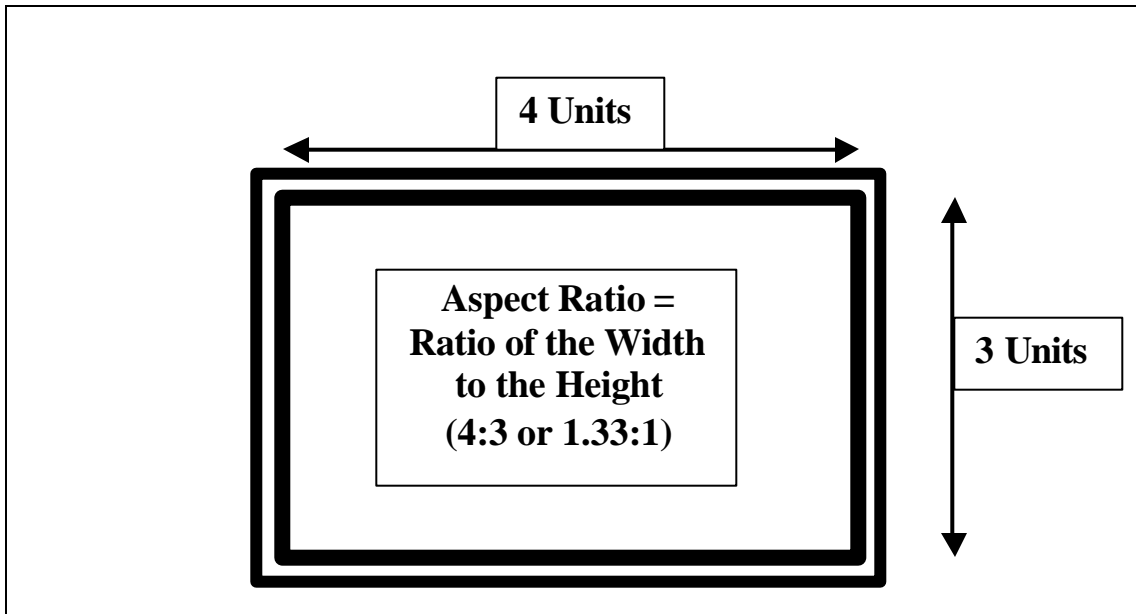
Figure III-3: Aspect Ratio.

Second, the definitions of *interlaced* and *progressive* scans need to be addressed. An electron gun is used in a computer monitor and television screen to charge florescent particles attached to the screen. When the electrons hit the florescent areas, they glow red, green or blue depending upon the color associated with the pixel. The video image is now displayed on a television or monitor using one of two methods. The first method has the electron gun excite the appropriate pixels in the first row. As it

finishes, it drops down to the third row. Every odd row is painted until the gun reaches the bottom of the screen. At this point, the gun paints all of the even rows from the bottom to the top of the screen. This cycle repeats until the monitor or television is turned off or stops receiving a signal. This method is called *interlaced* since every other row is alternatively painted until the entire picture is drawn. Interlaced scans are popular since motion in an image can be perceived even with a relatively slow electron gun. The mind is also tricked into perceiving a fully drawn picture despite the fact that only half of an image is updated during each pass.

The second method is where the image is sequentially drawn across each row in progressive order by an electron gun. As each row is drawn to completion the electron gun moves down one row and begins to draw the row directly below it. The gun must now be almost twice as fast as an interlaced gun since it must complete the full picture in the time that the interlaced electron gun only painted half of a picture. However, because of the speed of the progressive scan electron gun, people can perceive twice as much detail in the image than with interlaced representation.

The speed of the line draw defines the refresh rate. The refresh rate is how many times the image, which is in the video buffer, can be completely drawn on the screen each second. Two standard refresh rates for computer monitors are 60 Hz and 85 Hz. This means the full screen is painted with an electron gun 60 and 85 times per second, respectfully. The main benefit of a high refresh rate is to prevent flicker or a small distracting pulse of images as the rows are painted. Many studies indicate that the sixty hertz refresh rate has many problem issues since it is sometimes synched or in phase with surrounding lights. By changing a monitor's refresh rate to eighty-five hertz, most of these interference issues can be resolved.

Finally, frame rate is often confused with refresh rate. Although they can, on occasion, be the same, the two are usually quite different. Each unique image drawn on the screen is a frame. A television signal or computer application generates a full image of the information to create a full image at a specified rate of thirty or twenty-five frames per second depending upon the broadcast format. For television, there is an advantage in having images arrive at the television before they need to be displayed. If

the images are in a compressed format, there is now time to uncompress and display the images. If there is extra bandwidth available, more information can be encoded with the signals (such as stock tickers or error-checking connections) combined with the displayed images and sound. In the computer industry, many games are synchronized for optimum performance. Optimum performance includes many factors that may have nothing to do with the difficulty of drawing certain images. For instance, 3D images may need to check for collisions and redraw the image accordingly. Another example is when a networked game runs on a network clock for data exchange. As opposed to being limited by the video card, the program's frame rate may be limited by the computer's computational power. Thus, frame rate usually reflects the computer's maximum capability at any given moment.

Given the previous definition, video capture can be described based upon the standards of the television industry. One standard is the National Television Systems Committee (NTSC) with 525 lines of interlaced scan, 4:3 aspect ratio and thirty frames per second. In Europe and the Middle East, the main standard is Phase Alternative Line (PAL) with 625 lines of interlaced scan and a 4:3 aspect ratio. The frame rate is twenty-five frames per second. Newer standards include the High Definition Television (HDTV) standard of 480 progressive scan (480p), 720 progressive scan (720p) and 1080 interlaced scan (1080i) lines with a 16:9 aspect ratio. On a computer screen, two possible sets of dimensions are 1920x1080 or 720x480 pixels. To place this resolution in perspective, a 35mm film contains approximately 2000 lines of resolution and an IMAX film consists of 4092 lines of resolution. If the image needs to be modified for a NTSC or PAL compatible display, the video can be down sampled to fit the 4:3 aspect ratio of 720x360 pixels. Unfortunately, the analog signal of NTSC and PAL are not exactly related to pixel dimensions but they are closely approximated with the above examples.

b. Video Codecs

There are numerous video compression methods. The best-advertised, most current formats can be categorized as Microsoft, Apple, Real Network and MPEG. There are other formats available, but the aforementioned formats cover the majority of codecs currently in use. Some legacy codecs include Cinepak, H.261 and H.263. H.261

and H.263 utilize Discrete Cosine Transform (DCT) with Motion Compensation (MC), which was improved for later use in MPEG compression. Cinepak utilizes Vector Quantization (VQ) to provide excellent compression ratios for most video content. Most companies maintain patents and proprietary control over their premiere formats but have released software or code to allow the development and creation of compatible media.

Microsoft Inc. uses Advanced Streaming Format (`.asf`) and, more recently, Windows Media Video (`.wmv`) formats for its video files. The `.wmv` encoding consists of MPEG-4 video and Layer-3 audio encoding. Microsoft states that this format can generate CD-quality sound with 64 Kbps, near-VHS video quality at 250 Kbps and near DVD quality at 500 Kbps. It is currently one of the best-quality codecs with one of the smallest bandwidth consumptions. The primary drawback to this codec is its closely held proprietary status. The codec can only be streamed, encoded and played by Microsoft products. While many of the encoding and playing tools are free, the server software necessary for streaming content is among the most costly in the industry. There are also few tools available for editing. Microsoft Producer, one of the first tools to address this deficiency, was released in November 2001. While it is a free product, it still requires Microsoft's Office XP in order to work.

Apple Computers Inc. has popularized the QuickTime format with its `.mov` file structure. Apple provides free streaming servers but does not provide free encoders outside of its native operating system. The free QuickTime Player can be utilized to play all QuickTime movies, but a commercial version is required to allow a file to be converted to Apple's proprietary Sorenson 1 and Sorenson 2 codecs. Companies such as Adobe Systems Incorporated have also developed products such as Premier to capture, edit and create files in the `.mov` format.

Real Networks Inc. has created an extensive multimedia streaming market with its `.ra` format. Real Networks provides a free player, which can be upgraded to a commercial version with more controls and quality of service features. Demonstration versions of their server software are also available for download. Real Networks has gained a large following since they provide proven commercial packages of server software, software encoders, hardware encoders, players and editing tools. They provide

one of the largest bodies of online knowledge regarding streaming media. They also have an extensive network of trained and certified contractors capable of creating company solutions.

The Motion Pictures Expert Group (MPEG) has pursued three different video standards of MPEG-I, MPEG-2 and MPEG-4. The group is also currently working on MPEG-7 for content annotation. No single company claims full rights to the MPEG standards since they are developed by a collaboration of dozens of companies. Individual companies, however, own individual patents related to the MPEG standards. The standards are based upon Discrete Cosine Transform (DCT), Motion Compensation (MC) and Motion Prediction (MP). Sound is compressed using frequency masking and Huffman Encoding. The MPEG-1 standard was published with C software code that allows programmers to utilize the standard. Since then, encoders for the newer MPEG standards must be licensed from the appropriate patent holders.

c. Encoding Methodologies

The above codecs utilize methods such as Run Length Encoding (RLE), Vector Quantization (VQ), Discrete Cosine Transform (DCT), Contour-Based Images (CBI), Frame Differencing (FD) and Motion Compensation (MC). While dozens of books have been written on each individual methodology, the following brief descriptions summarize each approach.

Run Length Encoding (RLE) can represent groupings of similar pixels with a compression scheme that depends on the number of similar pixels in a row. For instance, 22 22 22 22 22 22 22 22 22 can be compressed to 8 22. The decoder checks the code words and represent the next 8 pixels with the color of 22. This method is only practical when there are a limited number of colors to represent a picture. It is not practical when there is a possibility for more colors than pixels in a picture. It may only be suitable for grayscale or even 8-bit images.

Vector Quantization (VQ) is a method of dividing an image into blocks of pixels. For instance, Cinepak utilizes a 4 by 4 (i.e.16-pixel) array. A representation of the most common blocks is stored in a table with a binary "lookup" value. The algorithm now decides which blocks of the image the standard blocks in the lookup table represent.

The more common blocks are given a shorter binary value for even more compression performance. The image is now represented with a series of binary lookup codes. The decoder receives the image information and can recreate the picture rapidly since it merely refers to its own block table. This method leads to "blocky" representation when there is high contrast or fast moving images, but it has a fast decoding method and a high compression ratio. As was illustrated with Run Length Encoding (RLE), the compression ratio can be increased for Vector Quantization by reducing the number of "representative" blocks. This leads to more image artifacts and a more "blocky" picture but also a faster encoding and decoding timeline. Discrete Cosine Transform (DCT) takes a block of 8 by 8 pixels and applies Equation (1).

$$F(u,v) = \frac{C_u}{2}\frac{C_v}{2}\sum_{y=0}^{7}\sum_{x=0}^{7} f(x,y)\cos\left[\frac{(2x+1)u\boldsymbol{p}}{16}\right]\cos\left[\frac{(2y+1)v\boldsymbol{p}}{16}\right] \qquad \text{Eq. (1)}$$

$$\text{where: } C_u = \frac{1}{\sqrt{2}} \quad \text{if u = 0 or} \quad C_u = 1 \quad \text{if u > 0}$$

$$C_v = \frac{1}{\sqrt{2}} \quad \text{if v = 0 or} \quad C_v = 1 \quad \text{if v > 0}$$

The resulting matrix, after the Discrete Cosine Transform is applied, normally reduces to a few distinct numbers in the upper left corner with the rest of the matrix as zeros. It is not unusual to have the 64-pixel matrix reduced to a few numbers in the upper left hand corner with zeros in all of the remaining positions. The resulting matrix can be represented by vectorizing the unique numbers in a zig-zag pattern and placing an end-of-stream symbol where the zeros begin. The block can then be recreated with an Inverse Discrete Cosine Transform (IDCT) given in Equation (2). The matrix resulting from this formula is an exact replication of the original matrix that was modified by the DCT. There is no distortion during the matrix transformations.

$$f(x,y) = \sum_{u=0}^{7}\sum_{v=0}^{7} \frac{C_u}{2}\frac{C_v}{2} F(u,v)\cos\left[\frac{(2x+1)u\boldsymbol{p}}{16}\right]\cos\left[\frac{(2y+1)v\boldsymbol{p}}{16}\right] \qquad \text{Eq. (2)}$$

Contour-Based Images are closely related to object rendering. Edges of objects are detected from images and textures are extracted. The edges are now

represented by polynomial formulas. Some codecs such as MPEG-4 version 2 identify such objects and employ lookup tables to limit bandwidth usage; thus allowing easier composition of the final scene. The formal implementation of this concept is called Binary Format for Scenes (BIFS) [www.mpeg.org].

Frame Differencing uses the differences in one frame as compared to another frame. Once again, quantization can be utilized to achieve compression since there are usually few-to-no changes between sections in consecutive frames. In order to prevent visible errors, key frames are generated at specific intervals as new base points for future frame comparisons. Otherwise, errors created by close matches in one key-frame propagate through the rest of the images.

Motion Compensation detects an object moving across the screen. MPEG-4 version 2 expects to predict and change the rendered position of objects as sprites. Currently, since object detection is a difficult problem, pixels within a large block are tracked for movement. A vector is generated to represent the distance and direction of the movement. The entire matrix does not need to be transmitted but only the movement vector. In MPEG-1, a 16 by 16 pixel matrix is used to predict such movement.

D.   COMMERCIAL SOFTWARE CAPTURE AND CONVERSION

There are numerous applications that can capture and convert video. This thesis explores four commercial products that represent a cross section of the different formats available. Microsoft, Apple, generic-implementation AVI and MPEG formats are evaluated here for file compression, quality, characteristics of use and interoperability.

1.   Microsoft Media Encoder

The Microsoft Media Encoder (MME) allowed for capture in `.wmv` format. The encoder provides the user with several encoding choices with corresponding streaming bandwidths ranging from 28.8 Kbps up to 1500 Kbps. Each encoding scheme is uniquely associated with the final delivery method. For instance, 56 Kbps is associated with modem delivery while 250 Kbps is associated with Cable or DSL delivery. The application uses "wizard" interfaces to walk the user through session configuration.

A MME session can broadcast a video stream, record video to a file or accomplish both at the same time. Once established, the session's statistics can be monitored and controlled from the application's session panel. When the session is a live streaming event, key factors such as bandwidth, video preview and the number of users connected to the session are displayed. For a session, in which the video is saved to a file, information such as file size, remaining space on disk, time left for recording, etc. are displayed.
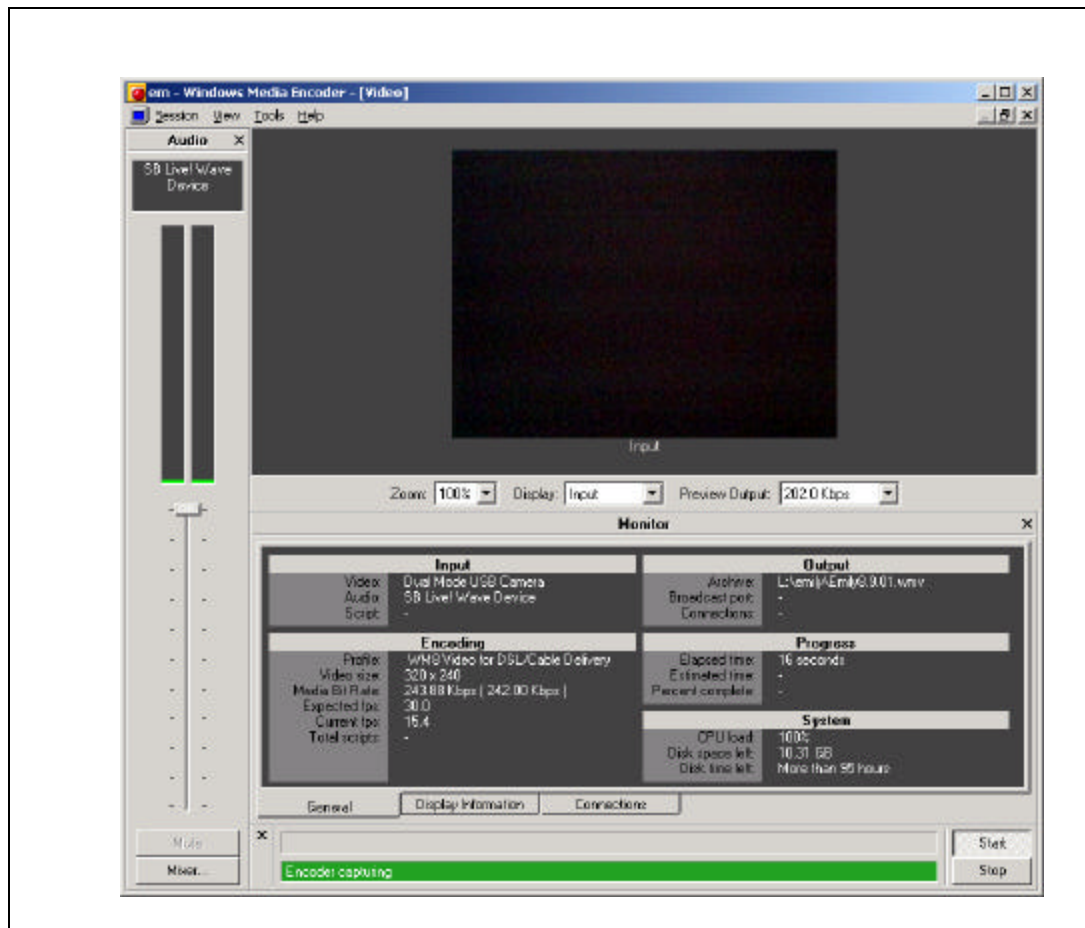
Figure III-4:   Microsoft Media Encoder (MME).

2.   QuickTime Player Professional

The QuickTime Player application cannot capture software from a video source in real-time. However, it can convert video that is already captured to the higher compression scheme of Apple's Sorenson codecs. For proper compression

to occur, the files need to be in an `.avi` or `.mov` format. While the player can render MPEG video, the current version was not able to convert the format without losing the sound channels.

The QuickTime Player Professional is a downgraded version of the Sorenson Video 3 and is sometimes referred to as the Standard Edition. The Sorenson Video 3 has all of the features of the Standard Edition with improvements such as better performance, enhanced quality, support for alpha channel/chroma key, color watermarks, automatic key-frames through scene, change detection, bi-directional prediction, support for One-Pass and Two-Pass Variable Bit Rate (VBR) compression, block refresh for packet loss correction, media key support through secure encryption, compression time packetization for error resiliency to packet loss and support for Mac OS X and multiprocessors. The increased features come with an increased cost of $1100.

The QuickTime application converts between video formats by opening the source file with the player. Once the file is open, the Export menu item may be selected. It is located below the File menu option on the Menu Bar. The export feature can compress the file for 20 Kbps, 40 Kbps or 100 Kbps streaming formats. The different options allow optimization for high motion or better sound fidelity. The tradeoffs in different video characteristics can be weighed to maintain the stream's bandwidth.
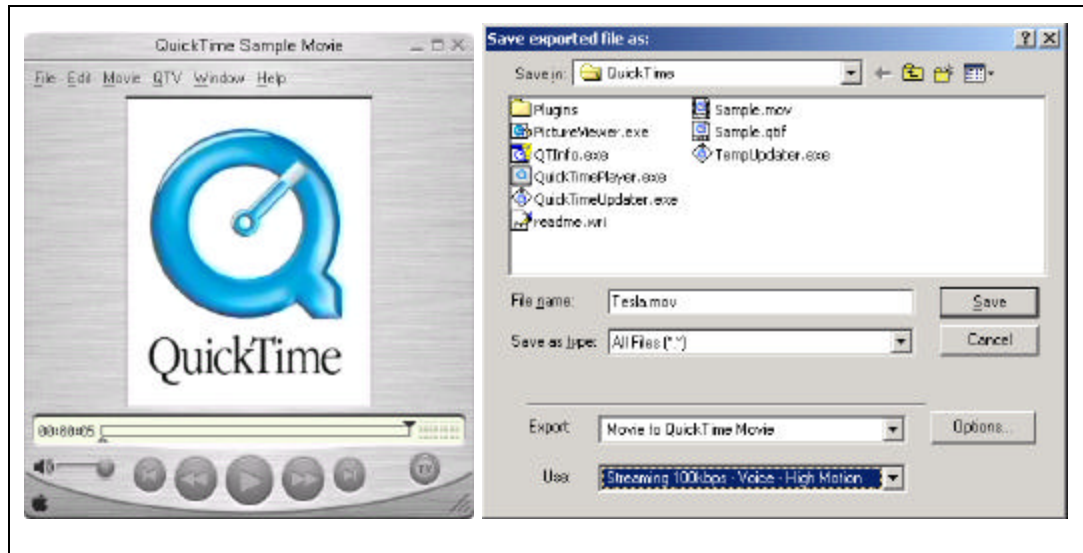
Figure III-5:   QuickTime Player Professional.

3.   AverMedia and AVI2MPEG

Most video capture cards are pre-packaged with software capable of displaying and saving video. Most generic or low-end capture cards do not support advanced compression codecs but save the file in an `.avi` format. The `.avi` format can quickly fill a computer disk. A 320x240 image captured at 30 frames per second generated two gigabytes of data in less than fifteen minutes.

One method of reducing the file size is by using a file-format conversion program. In order to explore additional formats, an open-source program was utilized. Avi2mpeg is a Windows DOS based program licensed under GNU's Not Unix (GNU) and can convert any `.avi` file to MPEG-1 compression. Additionally, it is based upon source code released with the original MPEG specification [www.mpeg.org]. The program completed the conversion without any loss but required an extensive period of time to finish the conversion. An eight-to-one ratio of conversion time to recording time was typical for the conversion process; equating to a one-hour movie needing eight additional hours to perform the conversion. The machine used for this test had a one-gigahertz processor with an Ultra 66 EIDE disk drive.

4.   Ravisant

Ravisant's CinePlayer DVR is an example of newer software that capitalizes upon
the computational power of modern computer processors. The software requires a 400
MHz or better CPU for most applications. Until recently, MPEG conversion was only
possible in real-time by utilizing specialized hardware. Ravisant's solution encodes
video, displays it for monitoring, and saves it to a file. The program does not offer
conversion from one format to another, but does allow the original capture source to be
stored in `.asf`, MPEG-1 and MPEG-2 formats. Single frames may also be captured, but
the program's main focus is to capture and playback television shows. The program's
primary limitation is that it can only run on Windows 98 or ME Operating system. Other
operating systems such as Linux, MacOS or even Windows 2000 cannot utilize the
current software [www.ravisentdirect.com/store/cpdvrp.html].
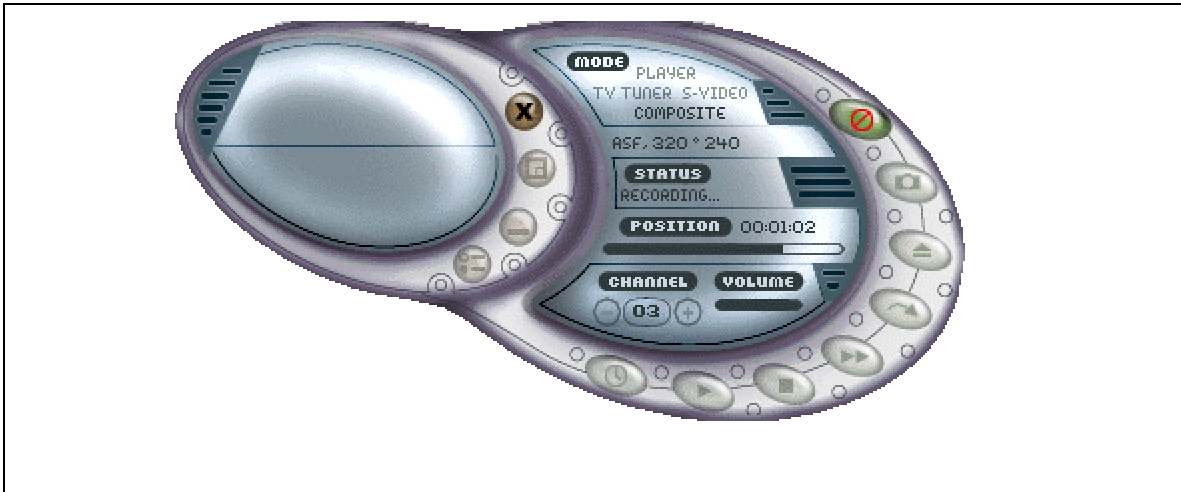
Figure III-6:   Ravisant CinePlayer DVR.

By using the DirectX C++ libraries unique to the Windows 98/ME line of
operating systems, Ravisant was able to develop a robust software application rivaling
most hardware MPEG encoders. Additionally, version 2.5 and above no longer use
Microsoft's MPEG engine. Instead, the software now uses Ravisant's own MPEG
engine. The main difference between the two engines is that Microsoft's has better edge
detection while Raviasant's has better motion prediction. Placing the same video side-
by-side, encoded through the two engines, easily displays this difference.

43

5. Players

The three multimedia players that dominate the personal-computer market are Microsoft's Windows Media Player, Apple's QuickTime Player and Real Network's G2 Player. Each player was able to render its company's proprietary formats. More significantly, each player cannot render the proprietary formats of its competitors. For instance, no player besides Apple's can render a `.mov` file, which was compressed using Apple's Sorenson compression. The same held true for Microsoft's `.asf` and `.wmv` formats as well as Real's `.ra` format

E. SUMMARY

The ability to capture audio and video has become easier with the increase in computer performance and decrease in cost. While high-end solutions perform specialized functions and can create captured content worthy of broadcast for DVD quality, lower-end solutions can easily create content suitable for Internet distribution. Additionally, the use of proprietary media formats does not necessarily gain a performance advantage when working with web enabled content. Some solutions are streamlined for a particular product line but are limited by their cross-platform restrictions and the inability to transcode to new formats. Meanwhile, open-source standard formats are freely available and work in cross-platform environments using a multitude of renderers and capture devices. Unless a specific platform or format solution is desired, low-end standards-based solutions easily rise to the needs of Web-enabled challenges.

IV. JAVA MEDIA FRAMEWORK (JMF)

A. INTRODUCTION

The Java Media Framework (JMF) offers the first set of open-source, open-standard Java libraries for the capture, rendering and transcoding of multimedia. The Application Programming Interface (API) supports multiple formats, capture devices, and streaming standards. JMF has the added capability of full compatibility with all Java APIs. These strengths create many interesting and unexpected capabilities such as rendering streaming media in a Java 3D environment or on a J2ME handheld device. The use of Java also allows the programs created in JMF to work wherever a Java Runtime Environment (JRE) is present. An application created on a Windows platform can thus also run on a Linux or Apple machine. JMF is currently the only option for cross-platform, open-source multimedia development.

This section discusses the multiple models available for the JMF API that allow it to capture, render, transcode and stream multimedia. The player model, processor model and session models are discussed. Additionally, JMF is based upon a well-structured format allowing for easy extensibility. Any component available for the API can be inherited or overridden by another class. Hence, custom components are easily created. One such example, explored in this paper, was when a newly developed screen capture library was developed for release with a newer version of the Java Software Development Kit (SDK). JMF adapted the new libraries to create a custom datasource for video capture by recording the computer screen display. Finally, this section covers the example program included with the JMF API, JMStudio.

B. JMF APPLICATION PROGRAMMING INTERFACE (API)

In order to gain cross-platform capabilities that meet the various needs of diverse Internet users, the Java programming language was chosen for the creation of video captures, rendering and storage software. Java's API for rendering, capturing, processing, transmitting and saving video is named the Java Media Framework (JMF). The JMF 1.0 API was originally developed by Sun Microsystems, Inc., Silicon Graphics

Inc., and Intel Corporation. The JMF 2.0 API was developed by Sun Microsystems, Inc. and IBM. The current release is JMF 2.1.1a, which has significant performance improvements that include its own DirectAudioRenderer and increased interoperability with streaming servers from companies such as Apple, Sun and Kasenna. The Java API allows for interoperability with other Java APIs and provides great extensibility. For instance, images captured using the JMF API may be rendered as textures upon three-dimensional objects using the Java3D API libraries.

1.   Overview

The JMF API is designed to interact with time-based media and utilizes an optimized family of modules that can pull or push a data stream from one component to the next. The basic model is founded upon an input source that can be processed and then sent to an output source. The input source can originate from a capture device such as a microphone, camera, file or network connection. The data may be processed into a different format, transcoded, compressed, decompressed or have an effect adjust the final multimedia presentation. The output of the file may be rendered to a video screen or speakers, saved to a file, or streamed across a network connection.
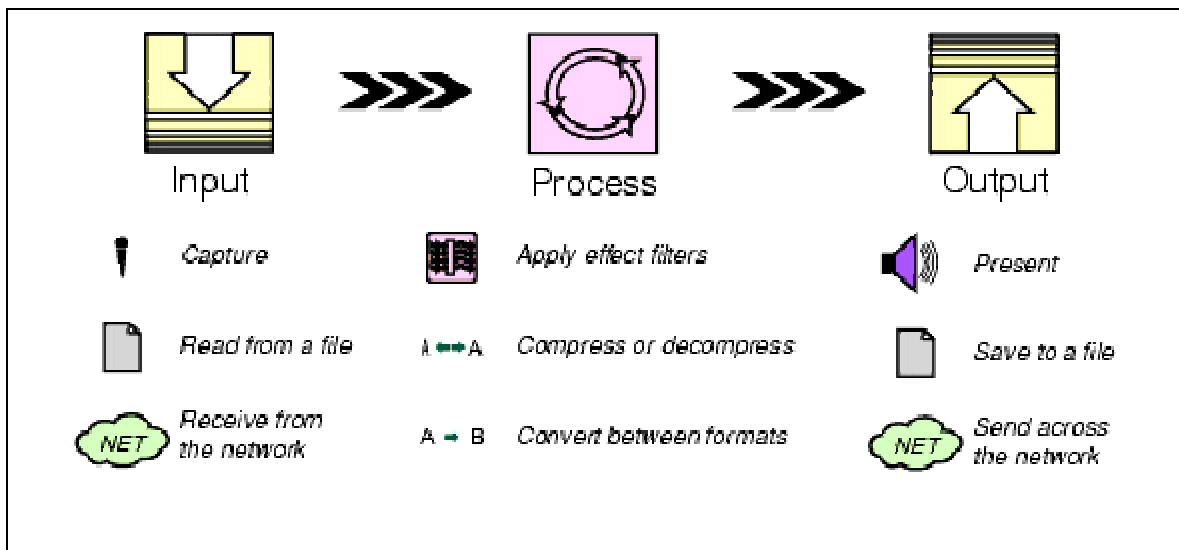
Figure IV-1:   Media Processing Model [Picture from: JMF 2.0 API]

2. Player Model

JMF API 1.0 was the genesis of the player model. The model was designed to allow video and audio to be rendered (i.e. played). The model was designed to grab a DataSource from either a file or a network connection, and then render the multimedia through speakers and monitor screens. JMF API 2.0 expanded the model by allowing the player to render from various capture devices. Due to the modularity of the JMF API, the modification only occurs in the DataSource component. The Player then connects to the DataSource in standard fashion. The Player, however, cannot modify the DataSource and, therefore, must render the media exactly as it is presented from the DataSource.
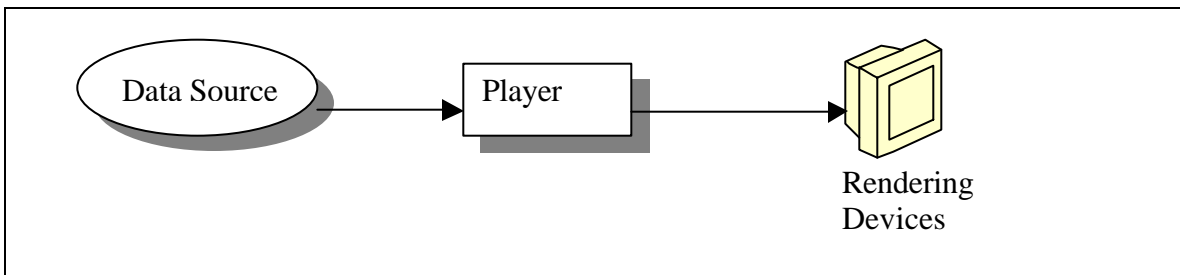
Figure IV-2:   Player Model [From: JMF 2.0 API]

The Player Model is based upon the Clock interface that is divided into two primary states: *Stopped* and *Started*. The Player Model further divides these two states into six states of *Unrealized*, *Realizing*, *Realized*, *Prefetching*, *Prefetched*, and *Started*. Each state returns event handler messages to allow tracking of state progression. For instance, the shift from *Realizing* to *Realized* returns a *RealizeCompleteEvent*. If the program was designed to not progress until a state event is returned, better control can be maintained over the movement between states. This approach helps to precisely synchronize multiple events within an asynchronous application.
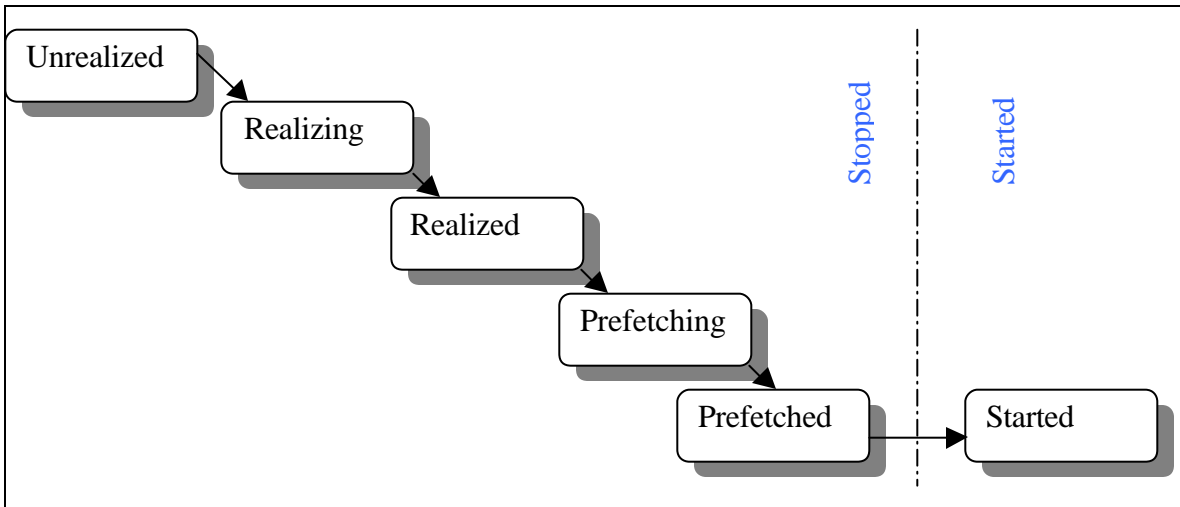
Figure IV-3:   Player State Model [From: JMF 2.0 API]

The Player divides the Clock state of Stopped into five phases. The first phase is *Unrealized* when the Player is initialized. It does not yet have any information about the media it is supposed to support. The second phase is *Realizing* during which information is gathered about resources that may be required. Also, information that needs to be gathered only once, not including exclusive-use devices, is gained. The third phase, *Realized,* moves the Player into a state where it knows which resources are necessary to render the media it is supposed to present. It is connected to all objects necessary to display visual components and available controls. However, it has not taken control of the resources. The fourth phase, *Prefetching*, allows the Player to take control of exclusive-use resources and takes any final steps in preparation to render the media stream. Final buffers are also allotted to allow the display of the media. The fifth phase is *Prefetched*, in which all resources are in place and the Player is ready to switch to *Started*. The final phase is *Started*. The time-based media and media time are mapped to the running clock. Once the correct Clock time is reached, the components of the Player are displayed. The stop method removes the player from the *Started* phase and returns it to the *Stopped* phase.

3. Processor Model

The Processor Model is an extension of the Player Model. It has all of the Player's functions and methods. In addition, it allows for the modification of the data stream created from a DataSource. The Processor is no longer restricted to the implementation defined by the source of data since it can extract individual tracks and modify them. The Processor is also capable of outputting its media as a DataSource. Hence, other Processors and Players can connect to the Processor for further manipulation of the stream and added functionality. The DataSource can also act as the input for a DataSink. The DataSink can be a file or network connection. With this added functionality, the Processor Model is capable of storing or streaming video and audio files.
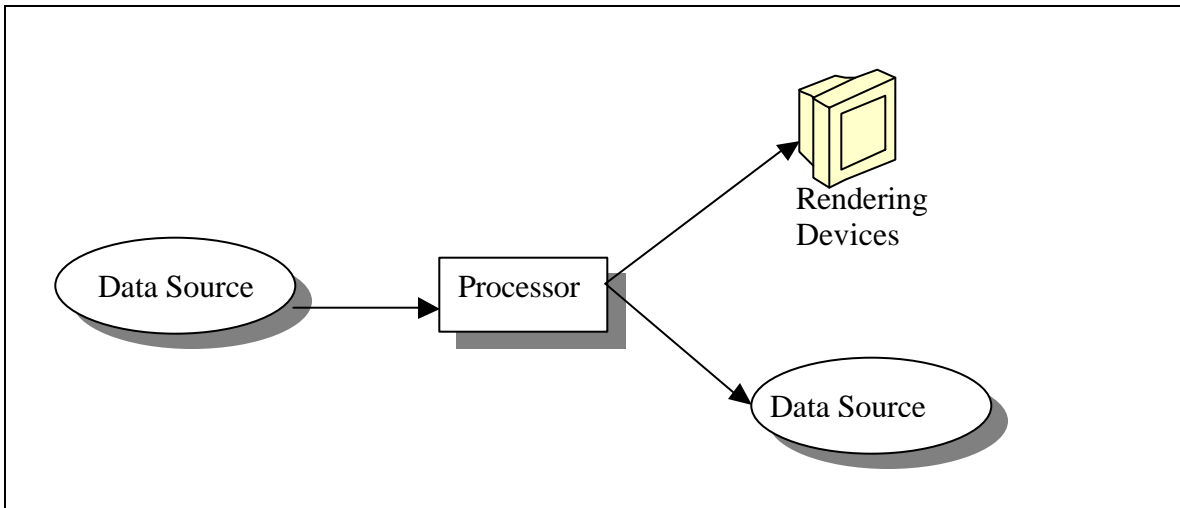
Figure IV-4:   Processor Model [From: JMF 2.0 API]

The Processor Model also contains two additional states directly before the Realizing state. The first state is *Configuring* and is located after *Unrealized*. In the *Configuring* state, the Processor connects to the DataSource, demultiplexes the stream into its individual tracks and accesses information about the input stream. The next state is the *Configured* state. It occurs when the Processor is connected to the DataSource and the data format has been determined.
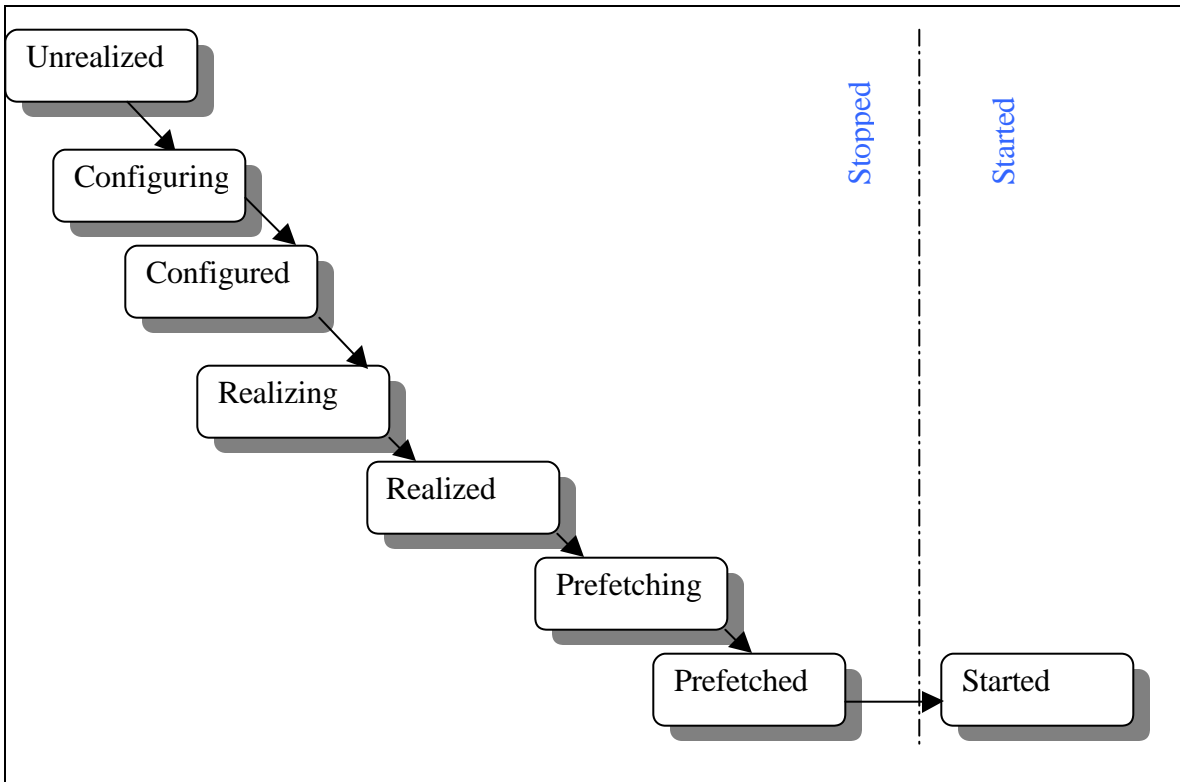
Figure IV-5:   Processor State Model [From: JMF 2.0 API]

4.   Session Model

The Session Model utilizes a Session Manager to control the flow of Real-Time Transport Protocol (RTP) and Real-Time Control Protocol (RTCP). These two protocols are used to establish a streaming media session. The streaming media session is an association of applications that are managed to control the connection and transfer of real-time data. A connection uses a minimum of two ports between two machines. One port is used to control the session while the second is used to stream data. If more than one medium is used such as audio and video, common practice is to open pairs of ports to stream the individual tracks. Using this method, a person can connect to only the data in which they are interested and save bandwidth.

The RTP and RTCP packets use Transport Layer Protocols such as Transmission Control Protocol (TCP) or User Datagram Protocol (UDP). TCP is a connection-oriented protocol that attempts to insure that data is transported from one machine to another in a reliable form. If the packets do not arrive, the sending machine will retransmit the

missing information. UDP is not connection-oriented and does not attempt to insure that packets arrive in a reliable manner. If a packet does not arrive, there are no mechanisms to signal for a retransmission. Since time-based media loses its value as time passes, most RTP servers and clients utilize UDP for content transfer. However, since there are a limited number of control messages and they are important for controlling a larger-bandwidth consumer, RTCP normally utilizes TCP. RTP and RTCP are not otherwise normally available to networked Java applications since RTCP requires access to low-level Internet Group Message Protocol (IGMP) datagrams. IGMP (and hence RTCP/RTP pair) can alternatively be implemented via the Java Native Interface (JNI) [Myjak 01].

Once a session is established, the modularity of JMF utilizes the Processor Model to complete its functionality. When receiving data from the network, the Session Manager performs all of the necessary functions to maintain a connection. It can use a Uniform Resource Locator (URL) to specify a server's Internet Protocol (IP) address. The Manager automatically handles the RTCP messages that maintain the connection. The tracks can now output from the Manager as a DataSource. The rest of the Processor Model connects to the DataSource as before and can process the data, render the data, save the data or perform all three of the functions.
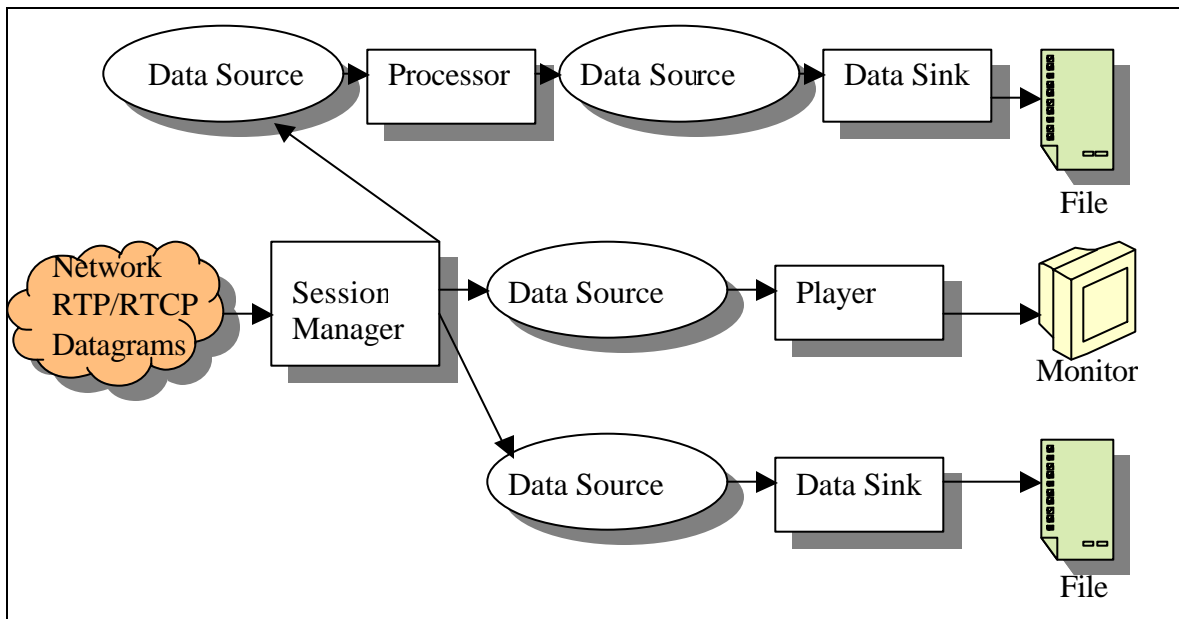
Figure IV-6: JMF RTP Reception model [From: JMF 2.0 API].

51

The transmission of streaming data works in the same manner as receiving data. In this case, the Processor Model is used to capture or display multimedia information. The data stream is sent to the Session Manager. The Manager will connect to any client which uses its IP address, correct protocol and correct port number. A separate session is established for each client that connects to the serving computer.

Due to the unreliable nature of UDP transference, a RTCP/RTP monitoring agent may become necessary to track network connectivity, network utilization, and quality-of-service (QOS). Additionally, JMF's ability to send and receive RTCP and RTP packets qualify it as a possible media for Virtual Reality Transport Protocol (VRTP). All of these uses demand a monitoring program which can provide useful statistics. RTPMonitor is one such program. [Afonso 99].
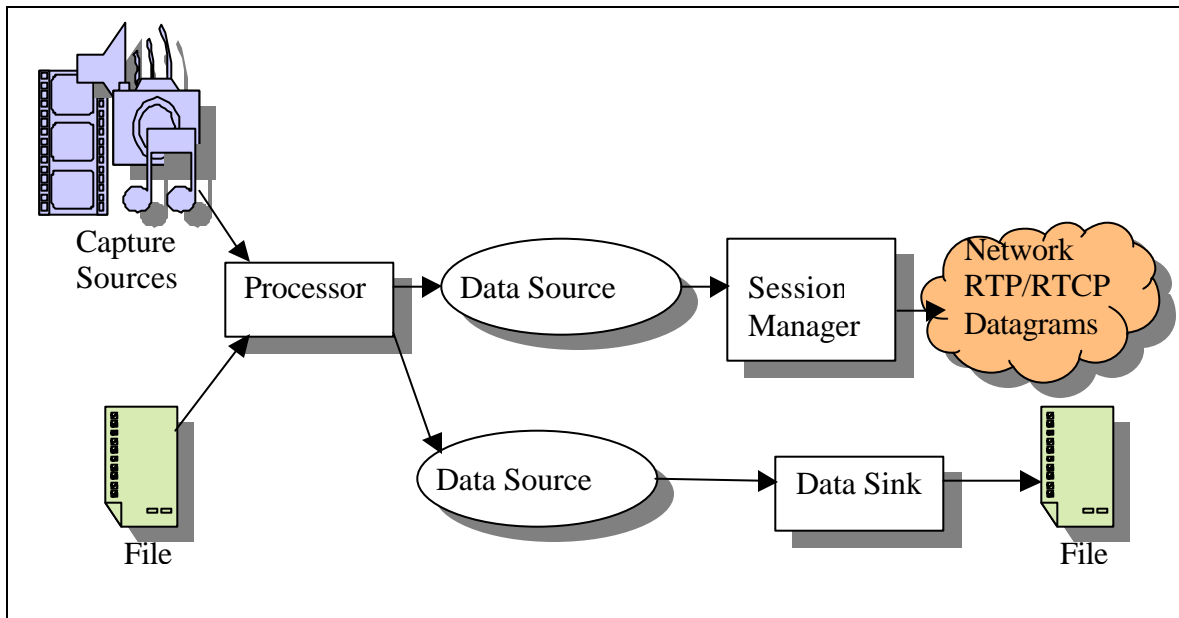
Figure IV-7:   JMF RTP Transmission Model [From: JMF 2.0 API].

5.   Extensibility

The extensibility of JMF is realized through its modularity. Any JMF component may be replaced by a custom component created by the programmer. JMF does not limit the programmer to using only pre-canned solutions that fulfill the needs of IBM and Sun. The programmer may change any component that supports the capture, processing and

rendering of time-based media. Two methods are provided for supporting custom processing: plug-ins and custom components.

a. Plug-Ins

The JMF API describes five types of plug-ins that are utilized to provide the programmer direct access to the time-based media. They are *Demultiplexer*, *Effect*, *Codec*, *Multiplexer* and *Renderer*. The *Demultiplexer* separates the individual tracks if they are multiplexed into a single stream. An example is a QuickTime movie that contains audio and video tracks interleaved as a single stream. The *Effect* plug-in creates special effects for each individual track of media. The *Codec* plug-in can encode and decode data. The *Multiplexer* plug-in places the multiple tracks into a single stream that can be used as a DataSource. Finally, the *Renderer* plug-in processes the track data and delivers it to the appropriate display device such as a video screen or speaker.
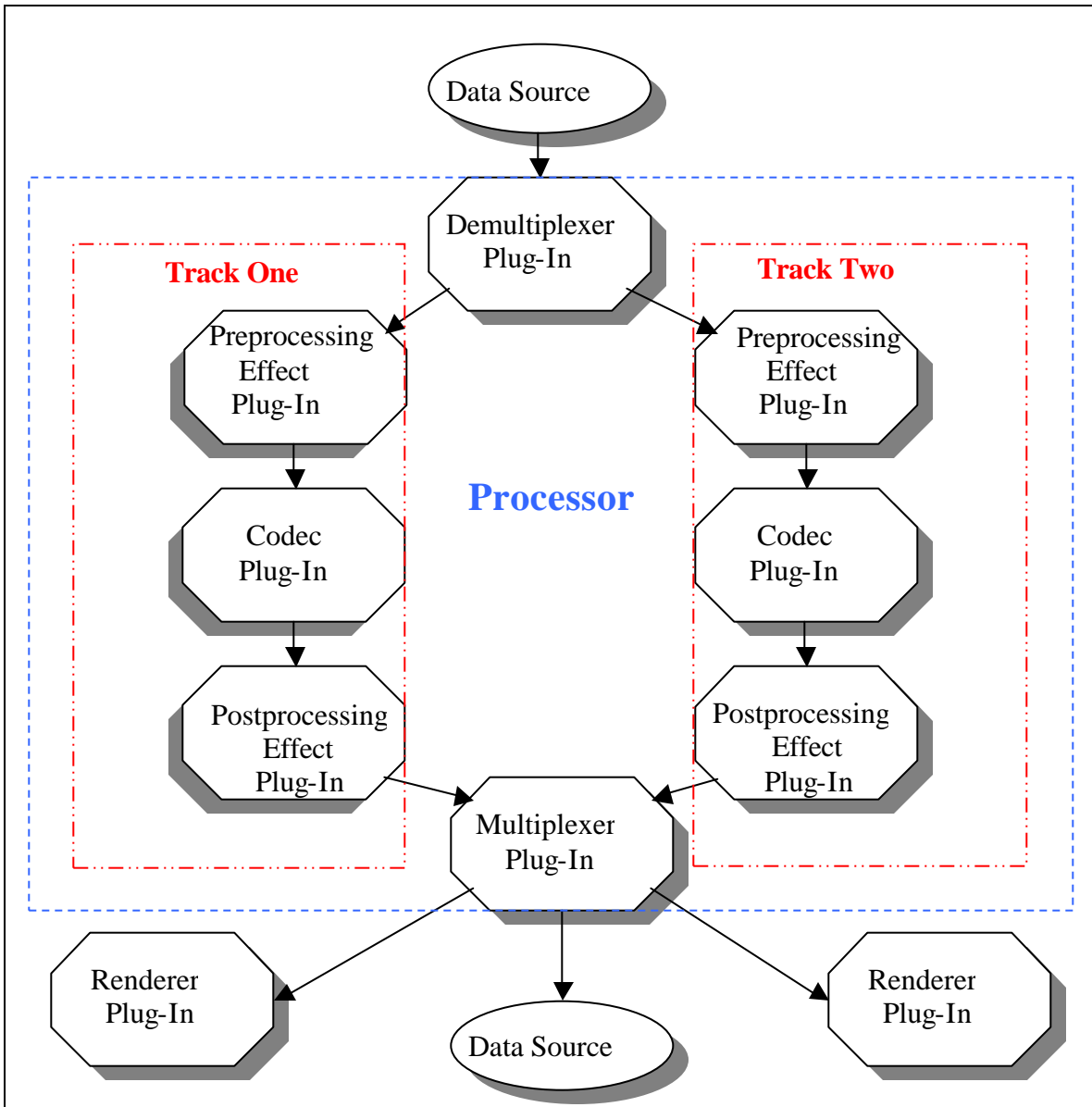
Figure IV-8: Processor Model Plug-Ins [From: JMF 2.0 API]

b. Custom Interfaces: DataSource and MediaHandler

A separate level of customized control can be realized in the form of custom JMF interfaces. Programmers can create their own custom versions of Controller, Player, Processor, DataSource and DataSink. For instance, the original specification of JMF does not support screen capture. Nonetheless, the `java.awt.Robot` class supports the capture of screen images and can be used as a custom DataSource. Using

Java 2 v.1.3, the new class has become available independent of any changes in JMF. Example code is located at

http://java.sun.com/products/java-media/jmf/2.1.1/solutions/ScreenGrabber.html

This interesting example allows a processor to connect to the Data source using a self-defined "//screen" protocol. The concept is to allow the customization of one component and then have the other JMF components interact with it seamlessly.

C. JMSTUDIO SUMMARY AND RECOMMENDATIONS

When the Java Media Framework (JMF) Application Program Interface (API) is installed upon a machine, Sun's example program named JMStudio is installed as well. The program demonstrates the built-in functionality of JMF's classes in a single integrated application. Every major function that can be accomplished by JMF is demonstrated through the multiple menus within JMStudio. Source code for JMStudio is also available for publicly released codecs and all JMF code, provided as plain-text .java source-code files. Proprietary codecs are distributed only in precompiled binary form. Appendix A contains highlights of JMStudio's capabilities as well as several non-documented operating procedures.

Upon comparison with several commercial time-based media software packages, it appears that JMStudio might have been improved in several areas. First, some software allows the user to select the length of time for the encoding session; i.e. user enters the time in seconds or number of frames desired for the recording and then presses the record button. The application can then operate independent of the user and terminate the recording session when the time ran out or number of frames was recorded. A similar feature for preset starting times also needs to be added to JMStudio.

Second, the higher-end software allowed for direct capture to a highly compressed format. A common format for most packages was the ability to save directly to .mpg, .asf or .ra formats. The ability to directly compress the video allowed for larger image capture for a longer duration of time than non-compressed or slightly compressed formats. This is a significant restriction since Windows operating systems did not allow

55

individual `.mov` or `.avi` files to exceed two gigabytes in size. This limitation imposed strict policies upon capturing extended periods of video.

Third, there was no single-button ability to capture a session. JMStudio required several setup screens and prior knowledge of which menu items needed to be selected in the proper order. Without this prior knowledge, it is extremely unlikely that any inexperienced individual is able to operate a capture session. Many commercial applications presented the user with a single button that used default settings to allow immediate use of the program. Additional menu items are provided for a customer that desires more options or a custom capture session.

Finally, there are no batch encoding or transcoding capabilities within JMStudio to provide for sequential conversion of multiple files during a single session. JMStudio only allows for the transcoding of a single file at a time. Microsoft, however, provides a separate utility for batch encoding files to multiple Microsoft formats. Apple is similar to JMStudio since it provides export functions that do not batch transcode. The ability to have a capture program transcode a capture file automatically greatly reduces the time required to create streamable formats. It is unlikely that a single choice of capture formats will satisfy all of the needs for a diverse market. With the multitude of Internet connections and their abilities, different users will desire different formats. Example Codec bitrates are 28 Kbps, 56 Kbps, 100 Kbps, 250 Kbps or higher. Each connection provides tradeoffs in media fidelity for bandwidth consumption. A batch encoder can account for these differences and provide multiple versions of a single media format without a direct increase in human effort.

D.   CONCLUSION

The Java Media Framework (JMF) Application Programming Interface (API) is well suited for the creation of distance learning tools. Its modular and extensible nature creates strengths unmatched by any other API in any programming language. While Java still lacks the speed of a compiled computer language, computer hardware speed and capability is accelerating at a rate where this issue may soon be nullified. By exploring the models employed in JMF, the ease of media tool development is realized. Additionally, the limitations and extensibility of JMStudio are discussed. In summary,

computer hardware has finally reached an acceptable performance level to allow the Java

programmer to effectively produce, process and play multimedia.

THIS PAGE INTENTIONALLY LEFT BLANK

V. CONTENT DISTRIBUTION

A. INTRODUCTION

Content distribution of streaming media encompasses additional technical solutions necessary to store and distribute audio and video. The storage of media can be mounted in large computer RAID solutions or burned into recordable compact disks. The distribution of content relies upon computer servers, software installed upon the servers, network protocols and physical connections to the other computers. Client computers connect to the servers through the use of the same network hardware and software. As more capabilities are demanded and become technologically feasible, more content will be available in higher fidelity on the Internet than ever before.

This section discusses the technical requirements necessary for the end-to-end (server-to-client) connection of streaming media. The nature of and specific requirements for this data type are addressed. Multiple storage solutions are compared to extrapolate future trends in price and capability. Particular server software solutions covering the spectrum of currently available technology are analyzed and evaluated. The protocols and networks used as the underlying transportation device of streaming media are also examined. The examination here is necessarily broad in order to encompass the entire range of solutions available.

B. STORAGE AND DISTRIBUTION

Few applications place such large demands upon a computer system and network as video and audio recording. Sheer volume accounts for many multimedia issues. Each pixel can contain one to four bytes of data. With standard capture sizes of 160 x 120, 320 x 240 and 640 x 480 corresponding to 19,200, 76,800 and 307,200 pixels per image. Assuming an average of three bytes per pixel, each image will consume 57.6 Kilobytes, 230.4 Kilobytes and 921.6 Kilobytes. Images are captured in a myriad of rates from 1 per second up to 30 per second. Assuming an average of 15 images or frames per second, the storage requirement per second is 864 Kilobytes per second up to 1.4 Megabytes per second. Similar calculations can be performed to reveal that

uncompressed Compact Disk quality music can generate close to 1.5 Megabytes per second as well.   Also, when larger formats such as HDTV 1080i are captured, Terabytes and Petabytes are a normal measure for file size. Equation 3 describes the calculation of video capture size in Megabytes.

$$\frac{\text{(pixel width) x (pixel height) x (color bit depth) x (fps) x (duration in seconds)}}{8,000,000} \qquad \text{Eq. (3)}$$

To use the formula, suppose you want to capture a three-minute video at 15 frames per second, with 24-bit color and an image dimension that is 320 by 240 pixels. The file size, using the formula is:

(320 pixels) x (240 pixels) x (24 bits) x (15fps) x (180 sec) / 8,000,000 = 622 Megabytes

The only variables affecting an uncompressed video are image dimension, frames per second, color depth and duration. By modifying any of these variables, the digitized video size will be increased or decreased. Another set of variables is incorporated when compression is employed. While compression schemes can achieve a solid 15:1 reduction in size, long sessions with high-quality recording parameters quickly generate data in the gigabytes range. Therefore, large file sizes are the nature of digitized video.

In addition to the large file sizes, read/write speeds for hard drives need to be considered. Current standards specify 66 Megabytes per second transfer rates and faster. The needs for this thesis were fulfilled using an Ultra 66 EIDE disk drive. However, for applications such as HDTV, FireWire and FiberChannel, drives with transfer speeds in excess of 400 Megabytes per second are needed. These drive speeds are necessary to allow the storage as well as the timely delivery of synchronized media. Drive speed is critical since a session can be ruined if delivery of sound and video are not synchronized or jerky. However, once an audio/video stream has been captured and if necessary, further compressed during post processing, different access rates must be considered for delivery.

1.   Web Servers

Streaming media servers can be divided into two categories: Video-On-Demand (VOD) and scheduled media sessions. Many media servers are beginning to incorporate

the ability to steam live events. However, these servers are not classified as scheduled delivery since the process of offering a live event is normally not automated.

Servers explored for VOD were Microsoft Windows 2000 Server, Apple's Darwin and Real's G2 Server. Servers capable of scheduled delivery are Cisco's IPTV, 2NetFx's ThunderCastIP Server, Streaming21's Media Server and Media Caster. Also, new models of streaming services are beginning to develop which can be utilized in restricted bandwidth personal devices. One commercial company leading this arena with MPEG-4 technology is PacketVideo using their PVAuthor, PVPlayer and PVServer software tools. Most servers provide live broadcasts. However, except in a few cases noted, most servers are restricted to streaming their own proprietary codecs.

a.   Microsoft

Microsoft's Server Operating Systems are fully integrated with streaming media. Microsoft's NT 4.0 Server Operating System required a separate, free download from the Microsoft web site. The 2000 Server Operating System (OS) contains the Media Server that can be installed at the same time as the OS or installed at a later time with the Server Compact Disk. The Server software is designed to allow the Web Server, Internet Information Server (IIS), to stream Microsoft encoded multimedia. The server is signaled to begin streaming when a web link to the server is preceded with "mms://". If the file address is not preceded with this protocol identifier, the web server will merely begin to download the file. Such file requests are therefore not streamed.

For live broadcasting, however, the Microsoft Media Encoder application is necessary since the server does not have capture capability. The web pages on IIS can point to the broadcast point created with the encoder. Also, a copy of the live broadcast can be immediately stored on the Media Server for instant viewing. The tight integration of the operating system, web server, encoder and media server has created an environment where ZDLabs independently verified that a eight-way Compaq server was able to support 9000 28.8 Kbps streams or 2400 100 Kbps streams [http://www.microsoft.com/windows/windowsmedia/compare/ZDLabs.asp]. The encoder by itself is capable of supporting up to 50 unicast streams directly from the

broadcast point. Hence, Microsoft's Media Server is one of the most robust in its class but is restricted to Microsoft's proprietary formats.

b.   *Apple*

Apple's solution for streaming media in a Microsoft/Intel environment is to employ its free Darwin Streaming Server (DSS). The server allows the delivery of live or prerecorded media. Clients can also view media on demand. It has skip protection to cope with possible Internet congestion, authentication methods to guard protected media, playlists feature to simulate a live television or radio station, Web-based administration tools and the ability to set up a hierarchy using several layers of servers to broadcast streams to a scalable audience.

The Darwin Streaming Server (DSS) can function under a large number of operating systems. Some of these operating systems are Mac OS X, Linux (RedHat 6.2, Intel), Solaris 7 (SunOS 5.7), FreeBSD 3.5 (Intel) and Windows 2000/NT (Service Pack 5 or above). Since the code for the server is open-source, the use of DSS on other operating systems is highly encouraged. A suitable compiler and addressing of operating system characteristics should allow the server to be ported to new environments.

The DSS server supports a variety of encoded media. Preferred video formats are Sorenson Video, H.263, Motion JPEG A and H.261. Preferred audio formats are MP3, Qdesign Music, Qualcomm PureVoice, DVI 4:1, A-law 2:1 and 16 bit RAW. Other supported video formats include Cinepak, Motion JPEG B and MPEG-1.

With the proper formats place upon the server, a web page can be used to connect to the server using an embedded link. Figure V-1 contains example code that can be included on a web page to embed a QuickTime movie. It is a graphical link where the

`SRC=`http://my.webserver.com/linkimage.mov

is the link to the image and the

`href="`rtsp://my.streamingserv.com/sample.mov`"`

is the link to the streaming movie. When a client-machine user sees the web page displaying these links and clicks on it, the movie will be streamed to the QuickTime client if the proper QuickTime plugin is installed.

```
<HTML>
<BODY>
This is a sample use of the EMBED tag.<BR>
<EMBED SRC="http://my.webserver.com/linkimage.mov" width="150"
height="64" href="rtsp://my.streamingserv.com/sample.mov"
target="QuickTimePlayer">
</BODY>

</HTML>
```

Figure V-1:   Launching embedded QuickTime movie via HTML page.

c.   *Real Networks Inc.*

Real Network's G2 server comes in multiple versions with multiple capabilities. The main difference between the different versions (besides price) is the number of licensed connections that the server is capable of connecting. The software will run on Compaq Tru64 v5.1, FreeBSD 3.0, HP/UX 11, IBM AIX 4.3.3, IRIX 6.5, Linux 2.2 (libc6) Intel, Sun SunOS 5.6 (Solaris 2.6), Sun SunOS 5.7 (Solaris 2.7), Sun SunOS 5.8 (Solaris 2.8) and Windows NT/2000 Intel. For the Windows versions, it is recommended that the server versions of the operating system are utilized since server software optimizes memory management.

The free version of this software is called Real Server Basic. A detriment to the software is its non-scalability. The application installs only on one machine with a limitation of 25 connections. The Real Server Plus costs just under $2000 for sixty connections and Real Server Professional costs just under $6000 for 100 connections. Real Server Professional also comes with management software for enterprise rollouts and reliability.

Real Network servers support over 45 media types including MP3, SMIL and Flash. The proprietary .ra format is supported in multiple streaming speeds and intelligently managed through communication between the client and server to optimize the data flow. The server can also stream QuickTime formats to a QuickTime player using the standard embedding method with an RTSP URL pointing to the Real Server. While capable of streaming Sorenson, Qdesign and Qualcomm PureVoice, the Real

63

Player cannot render the content. Additionally, the server software is capable of broadcasting Apple's QuickTime movie format through the use of the Sorenson BroadCaster. The broadcaster encodes the capture stream and the Real Server reflects the user to the location of the broadcaster software.

d.   Cisco

Cisco's IP/TV is a subset of the Cisco Content Networking family. The full solution consists of Cisco 3400 series Broadcast Servers, Control Server, IP/TV client viewer and Archive Servers.   The Cisco IP/TV solution consists of two Broadcast Servers to stream and store live events. The first is the 3424 and is capable of streaming low bandwidth formats such as Microsoft Windows Media (MWM). The second server is the 3425 and is capable of streaming MPEG-1 and MPEG-2 video content for television-quality presentation. Archive Servers allow for on-demand viewing and rebroadcast a live event. The Control Server sets the policies for the Broadcast Servers and relays information to the clients. The Control Server automatically performs load balancing and network-performance optimization.

The Cisco solution capitalizes upon multicast technology as well as Cisco's underlying network routing infrastructure. By using multicast, bandwidth is optimized to deliver a maximum number of distributed feeds utilizing the smallest bandwidth possible. The content can be streamed across the Internet's Multicast Backbone (MBone) or tunneled using its SmallCast tunneling capability.   A twenty-license solutions costs approximately $6500. Additionally, the Cisco Enterprise Content Delivery Network (ECND) can provide a solution to allow the intelligent streaming of media to thousands of sites worldwide.

e.   2NetFx

2NetFx is pursuing the highest fidelity of data delivery possible. They utilize proprietary server and client software that works with packaged hardware to broadcast and render high-fidelity video and audio. The two components making this fidelity possible are the ThunderCastIP server and StreamRider Player software. The software was used during the Siggraph 2001 Conference for the delivery of High Definition Television (HDTV) content to the main conference floors. Two monolithic

structures housed extremely large HDTV displays for the audiences to watch the presentations from the main halls.

ThunderCastIP HDTV is the first multicast server for high-definition television over IP-based networks. The content can be a live event or pre-recorded and is compressible to less than 50Mb/s. The software contains features such as remote management and scheduling, datacasting, multiple live encoder support, network QoS, group and user management, VOD media management; plus support for MPEG-1, MPEG-2, MPEG-4, H.263, and HDTV.

StreamRider Player software lets users view and subscribe to MPEG-1, MPEG-2, MPEG-4,, H.263, and HDTV broadcasts from multiple servers, interactively select stations, capture and store video locally, adjust video window location and size, and play archived videos from multiple servers. It is optimized for multiple delivery systems such as satellite, LAN/WAN, ATM and fiber. The player combined with the server completes the system for delivery and decoding of high-fidelity multimedia.

f.   PacketVideo

The final trend in multimedia delivery is the provisioning of streaming services. Companies such as PacketVideo are leading this arena with proprietary services, servers, communications networks and supportable, proprietary protocols such as MPEG-4. By concentrating on the newest open standard to emerge from the Motion Pictures Expert Group (MPEG), PacketVideo is able to provide limited bandwidth in less powerful computer devices such as Pocket Computers and Internet-enabled cellular phones.

The PacketVideo solution is named PVPlatform 2.0 and provides for a full end-to-end (server-to-client) solution. The PVAuthor 2.0 tool allows for encoding of video and audio from live as well as previously encoded files. It supports transcoding from formats such as `.mpeg, .wav, .avi,.jpeg, .bmp` and `.mp3` into the MPEG-4 standard. The final content is configurable to a single image with audio, audio only or video and audio. The files are optimized for networks from 9.6 Kbps to 768 Kbps. Multiple frame rates up to 30fps can be specified. Also, frame sizes from 128 x 96 up to 352 x 288 can be specified.

The second component to the PVPlatform 2.0 is the PVServer 2.0. The server uses an extensible architecture allowing for the addition of new modules. The server can run on Solaris, Linux and HP-UX. It can support up to 1000 simultaneous streams over 64kbps. The files can be streamed or downloaded to the client component, PVPlayer. PVPlayer compensates for wireless transmission errors and supports the rendering of multiple formats. The company also allows its products to be modified by customers through a Software Development Kit (SDK).

2. Portable Media

The term "portable media" can be applied to many devices. While computer backup devices such as tape drives and robotic jukeboxes create media that is portable, latency during data retrieval degrades accessibility. Therefore, for this section, Compact Disks (CD) and Digital Versatile Disks (DVD) shall be discussed. While CD usage has wider availability, DVDs are beginning to rise in popularity due to their larger data storage capacity and hardware availability.

a. *Compact Disk (CD)*

Compact disks developed for computer data storage is termed Compact Disk-Read Only Memory (CD-ROM). The CD-ROM is identical in dimension to its predecessor, the audio CD. However, the CD-ROM contains areas that are divided into sectors containing data and error correction codes. There is also a file system that allows required files to be accessed easily and quickly. Current standards allow for multiple formats that allow the storage of up to 700 MB of computer data. This is the equivalent of 500 high-density (1.44MB) floppy disks. Current CD-ROM drives can transfer the full contents of a CD-ROM in less than two minutes.

A version of Compact Disk known as Compact Disk-Recordable (CD-R) allows a user to save data onto a blank, unformatted disk using a personal computer and CD writer. The writer is usually called a burner since it uses laser heat on the ink dye contained on the CD-R. Once burned, the disk can be played in any computer CD-ROM drive. Hence, 700 MB of data can be hand carried from machine to machine. Of interest: current prices place storage costs at approximately twenty-five cents per gigabyte of data; making CDs one of the cheapest storage medias.

b.   *Digital Versatile Disk (DVD)*

DVD-Video was one of the first formats developed for the Digital Versatile Disk (DVD). It used high-quality video MPEG-2 encoding and high-fidelity sound such as Dolby 5.1. The DVD format was required since CDs did not contain enough storage space for the larger files created by the higher-quality formats. The movie industry embraced this standard for the delivery of full-length movies.

A natural extension, as in the case of the CD, was to create a DVD-ROM standard for computer data storage. Game companies were the first to use DVD-ROM by converting multiple-CD games into a single-disk format. Later, the recordable version was developed as DVD-R; allowing users to create their own DVDs with a personal computer and DVD burner. While the prices are still disproportionably high when compared to similar data storage in CDs, DVD-R costs are decreasing rapidly. There are currently two storage capacities of 3.95 GB and 4.7 GB. While larger storage capacity is possible, standards and commercial products are not yet released to match. Current prices are approximately five dollars per gigabyte; making DVDs one of the most expensive storage medias available.

C.   PROTOCOLS

Multiple protocols are used in the delivery of streaming media. Network protocols can be considered using the International Standard Organization's Open System Interconnect (ISO/OSI) model to separate the multiple layers of a computer network. The appropriate protocols are placed within the layer that identifies its function. The seven layers, in order, are *Physical*, *Data Link*, *Network*, *Transport*, *Session*, *Presentation* and *Application*. A second model, known as the TCP/IP Network Model, consolidates several of the OSI layers into four distinct layers.   The four layers are *Physical/Link*, *Network*, *Transport* and *Application*. The TCP/IP model shall be used for this paper.

1.   Physical/Link Layer

The *Physical/Link Layer* represents the physical hardware and data link division of the model. The physical hardware sends one and zero signals in a predefined manner across a network transmission medium. The ones and zeros can be in the form of

electrons, lights, radio waves, etc. The layer specifies the media type, connectors and signaling which connects the multiple signaling devices. An example is an Ethernet PCI card, Ethernet hub, 100BaseT wire, RJ-45 connectors and Ethernet protocol. Once all devices are connected and communicating, they are located in the same collision and broadcast zones. These zones represent areas in a network where a signal sent by one computer can be detected by all nodes connected to the wire. If two computers send a signal on a shared physical medium at the same time, they will collide. Hence, these zones are typically called broadcast and collision zones.

The link state provides a source and destination address for each physical node in the network. The Media Access Control (MAC) portion of this layer defines how frames are transmitted across a physical wire, physical addressing of nodes, network topology, error notification and possibly flow control. The second portion of this layer, Logical Link Control (LLC), provides for service-access point (SAP) identifiers, flow control to upper layers and, possibly, identification and encapsulation of the higher-level protocols.

2.   Network Layer

The *Network Layer* connects multiple logical addresses and manages the paths through the network for the link. This layer connects multiple data links. In order to accomplish its tasks, the layer sends data packets and route packets across the network. The data packets contain payload data as well as upper-layer protocol control information. The route packets contain information about the multiple networks connected within the internetwork. The Internet Protocol (IP) and Internet Packet Exchange (IPX) protocols are used at this level. Since logical addresses are utilized, a network hierarchy can be established for optimum network traffic.

3.   Transport Layer

The *Transport Layer* distinguishes between upper layer applications, establishes end-to-end connectivity and provides for the reliability of the data transfer service. The layer distinguishes the upper applications from the same transport stream by using port identifiers. For instance, if port 25 is used, the Simple Mail Transport Protocol (SMTP) is assumed. Since other machines will make the same assumption, two machines can exchange information based upon this understanding even without prior communication.

Many other schemes are also possible. Additionally, if a reliable transport protocol such as Transmission Control Protocol (TCP) delivers packets, the packets are acknowledged by the receiver to the sender, retransmitted if not acknowledged, reassembled in their correct sequence and stored until ready if they arrive too quickly.

4. Application Layer

The *Application Layer* encompasses the remaining three layers of the OSI model; providing Session, Presentation and Application functions. Hence, this layer handles communications through mechanisms such as Named Pipes, encryption and operating system support for applications. Events supported at this level are operating system dependent and allow for an end-to-end connection by sending data from an application down the stack, across the network and back up the stack to communicate with an application on another machine.

D. NETWORK CONNECTIVITY

The hardware required for the transference of data across a computer network is heterogeneous, complex and diverse. Older protocols, devices and connections are still prevalent in legacy portions of the Internet that did not want to change a working system or had insufficient funds to upgrade. Companies such as IBM, DEC, Intel, Xerox, Cisco and Juniper have all left their mark upon the networking world. While some companies have faded into Internet history, new ones are creating expanded opportunities for future computer networks. Interoperable connectivity in spite of diversity is a central feature.

1. Local-Area Networks (LANs)

Local-Area Networks (LANs) are normally defined as a relatively small network located internal to an organization. A room or building full of networked computers can fill this definition. This arrangement is often termed and intranet. Once the LANs are internetworked with other networks, they may be considered as a Wide-Area Network (WAN). WANs may be part of the Internet, an Extranet, or both. An Extranet is a WAN providing service for a limited number of clients.

Currently, the most popular LAN interface is the Ethernet protocol with its associated hardware. The Ethernet protocol uses a Media Access Control (MAC) address

for each node connection to a network. The MAC address does not have to be unique, but is currently produced with each device receiving a unique address. The address comprises of twelve hexadecimal numbers. The first six numbers are IEEE assigned to a vendor for an Organizationally Unique Identifier (OUI). The vendor assigns the last six. Additionally, the protocol uses copper, wire or radio frequency as its transport media. Data can be transferred at rates of 10 Mb/s, 100 Mb/s or 1000 Mb/s in a multiplexed mode. In a dedicated, non-multiplexed mode, the data transfer can be doubled.

2. Internet Service Provider (ISP)

Most organizations connect to the Internet through the use of a provider. The provider may be one of the backbone providers such as AT&T, Quest, Sprint or MCI. The provider may be at the secondary level such as Bell Atlantic, SBC, Bell South or Cox Cable. Finally, the provider may be at a tertiary level who pool their resources with other tertiary providers under one of the upper providers. Diverse, large, parallel and fast networks are beginning to grow throughout the world. These networks have connected to the traditional Internet providers and extend the Internet.

A client connects to these providers either through a telephone, cable or network connection. Modems, routers or specialized devices allows the user to connect through Plain Old Telephone Service (POTS), Digital Subscriber Line (DSL), Cable television, Integrated Service Digital Network (ISDN) or other connection. The connection can be through copper or fiber-optic connections, microwave receivers or satellite links.

3. First-Generation Internet

The Internet grew out of several projects dating back to the late nineteen sixties. In 1968, the Department of Defense released a proposal to link Stanford Research Institute (SRI), University of California at Los Angeles (UCLA), University of California at Santa Barbara (UCSB) and University of Utah. The following year, the four sites were networked together with several other sites to become the Advanced Research Project Agency Network (ARPAnet). In the seventies, protocols such as Ethernet and TCP/IP were developed and deployed.

The term "Internet" was used in an ARPAnet planning document in 1974 but was not officially used until the mid-eighties when other networks began, such as the DOD's MILNET and National Science Foundation's NSFnet. By 1988, NSF had several T-1 connections across the nation linking dozens of sites to supercomputer centers, research facilities and academic campuses. The 56 Kbps links of ARPAnet were replaced with the 1.5 Mbps links of the NSFnet. By December 1993, the links were upgraded to T-3 lines for a backbone speed of 45 Mbps. Collectively, these links became the basis for the First-Generation Internet.

4. Second-Generation Internet

Unlike the original Internet, the Abilene network is the Second-Generation Internet (Internet2). It is currently only available to academic and research institutions. The network is closely managed to overcome some of the problems of the First-Generation Internet. First, the network's hierarchy is closely managed with several trunk lines connecting to Gigabyte Points of Presence (GigaPops). The GigaPops are responsible for the dispersal of traffic within their region to secondary and tertiary members. In order to enforce this scheme, membership fees are commensurate with bandwidth and hierarchy status. The network is also currently peered with twenty-seven other high-speed networks such as very high performance Backbone Network Service (vBNS), NASA Research and Education Network (NREN) and Defense Research and Engineering Network (DREN).

The goal of Abilene is to provide for low latency, high reliability, and advanced functionality. Low latency is realized in the form of centralized routers with trunking lines and internal, regional networks. There is currently 10,000 miles of fiber optic cable with ten routing centers connected to Qwest's underlying network infrastructure. Hence, there are a minimum number of routers for a packet to traverse. Additionally, high reliability is created with centrally managed Network Operating Centers observing the networks on a separate monitoring network. Cisco 12008 GSR routers are utilized for upgradability, manageability and reliability. Finally, multicast, quality of service and IPv6 can be effectively supported.

E.   SUMMARY

Content distribution of streaming media is a technically broad area where proprietary and open-standard solutions compete for the market place. Most commercial products support the open-standards and then distinguish themselves by offering proprietary solutions evaluated as a better answer. In the end, computer networks need to support the same underlying protocols and topology in order to allow all solutions to interconnect. As new technologies emerge and are adopted by the Internet community, added capabilities ride upon the fundamental base created to internetwork computers. When the capabilities become too revolutionary to be supported by the original networks, new networks have been developed to continue the advancement of technology. The trend of higher-resolution streaming media across larger, more reliable transportation devices shall continue.

VI.   VIDEO TELECONFERENCE (VTC) CLASSROOM CASE STUDY

A.   PREFACE

The equipment located in the NPS Distance Education Center offered an example of current commercial solutions available for academic institutions. Observing two full-quarter classes, a number of observations about effective instructional technique in such an environment were realized. The use of equipment, body motion, voice, gestures, technical staff and more were evaluated to reduce the instructor's workload while increasing the effectiveness of knowledge transference. Additionally, multiple tools were explored in the pursuit of asynchronous education. This study is the result of six months of day-to-day classroom observation, examination, trial and experiment.

B.   BACKGROUND

The Naval Postgraduate School (NPS) currently has four 26-person classrooms equipped with Video Teleconferencing (VTC) equipment. The classrooms are accessible via multiple Integrated Services Digital Network, Basic Rate Interface (ISDN BRI) lines for two-way video, audio and computer generated data transfer. The classrooms utilize the commercially available PictureTel 4000 Video Conferencing System for maximum efficiency and standards-based usability. Any H.320-compatible video conferencing system may connect to the school's classroom sessions using Federal Telecommunications System (FTS2000) through AT&T, or commercial dial-up phone connections.

The PictureTel 4000 Video Conferencing System in each classroom provides two-way communication between the classroom and distant site. Each classroom contains VCRs, electronic whiteboards, document cameras, facsimile machines, and PC's for computer-generated presentations. The system is designed to allow the NPS to carry out its mission of educating military and federal employees from afar.

1. Overview

The Video Teleconferencing centers were utilized to allow the observation of several different classes conducted in a well-established commercial solution for distance education. The two classes observed were 3D Modeling with X3D/VRML and Advanced Physically Based Modeling. The dynamics of the classroom audience were recorded and improved to the maximum extent possible over a six-month period. While all classes had local audiences, some classes were distributed using the PictureTel 4000 system to PictureTel 4000 users at other military installations. Some classes were distributed utilizing the MBone toolset on the Internet's MBone virtual network. All classes were recorded to videotape utilizing local VCRs. Various software tools were used to explore multiple distribution techniques such as JMStudio.

The class audiences comprised mainly of native-English-speaking students and conducted in written and oral English. A small proportion of the students were from countries such as Greece and Singapore who had passed English proficiency exams and had spent at least one year at NPS. One class was an elective, while the other was a Modeling, Virtual Environments and Simulation (MOVES) curriculum requirement.

While the majority of 3D Modeling with VRML and X3D was instructed locally in a VTC classroom, one week of class was conducted with the instructor approximately 500 miles away using VTC equipment at Camp Pendleton, California. This class was used as an introduction to VTC controls and provided familiarity with the VTC equipment. It demonstrated two interaction models: one with the professor in the room sending information to a distant audience, and the second from the audience in remote locations receiving information from the instructor. While the two situations may appear to be opposite versions of a single model, classroom observations reflect how significant the differences are to warrant the distinction of two separate models. Many observations were also recorded during the Advanced Physically Based Modeling class in reference to the remote attendance from Naval Undersea Warfare Center (NUWC) at Newport, Rhode Island.

2.   Class Environment

Two different classrooms in Root Hall were utilized for observation, Root 258 and Root 262. The room layouts were slightly modified throughout the semester as attempted improvements to gauge the effect of the difference. Course content was presented in a variety of methods to maximize the number of presentation media observed. After two semesters, perhaps every conceivable method of classroom use was explored within the difficult limitations imposed by poorly designed equipment configurations. The remainder of this section explains these limitations in detail.

a.   *Ergonomics*

Each classroom was approximately thirty feet wide by forty feet deep. Four rows of tables and chairs filled the rear three-fourths of the room. The tables offered a continuous surface across the width of the rooms with three feet for their own width. The front of each room was filled with a podium, large desk, computer system, fax machine, two VCRs and the Polycom VS4000 conferencing system. The system contains a document camera, two wall mounted Sony cameras, camera controller and a rack mounted VS400 control cabinet. Internet connectivity was provided for classroom computers with approximately two additional drops for student and instructor use. Internet access was through the NPS's OC-3 (155 Mbps) fiber optic connection to the Defense Research and Engineering Network (DREN). The VS4000 system had its own dedicated BRI ISDN connection for video conferencing capability. Due to security implementations, network access had to be requested before each quarter's class began.
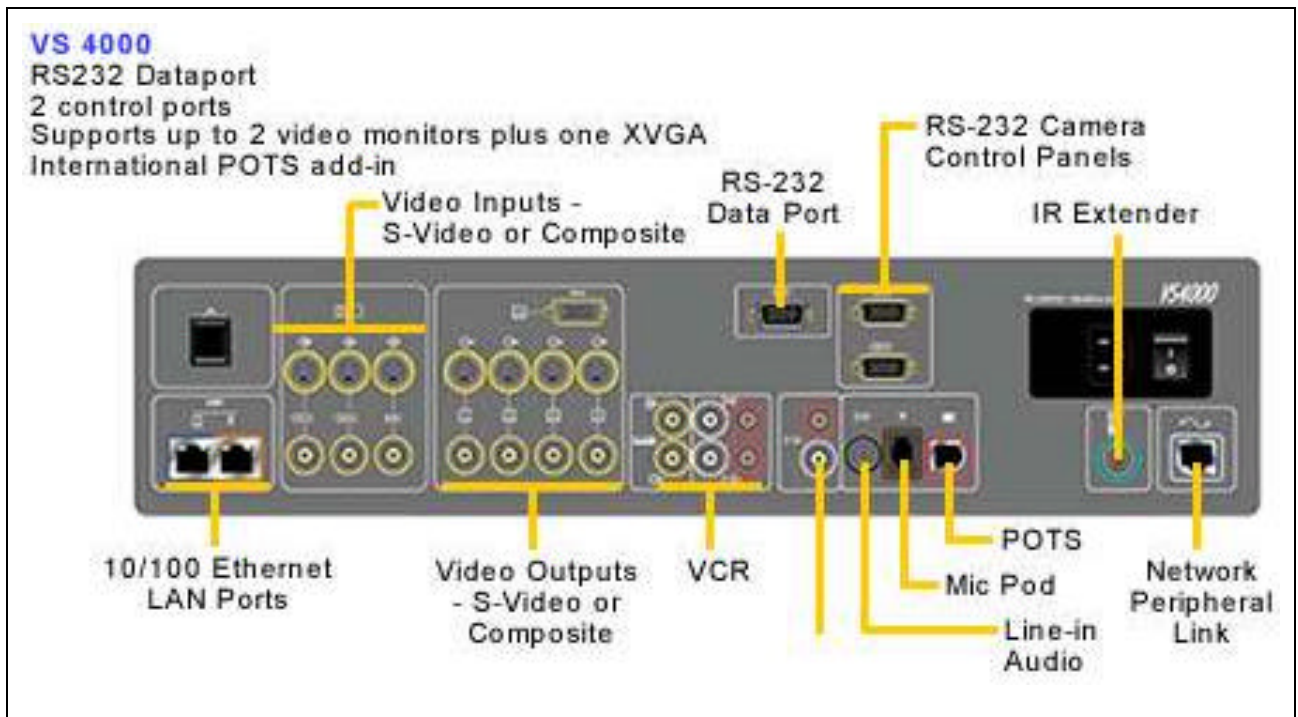
Figure VI-1:   PictureTel's Polycom VS 4000 rack mounted system connections [Picture from www.polycom.com].

       The system is designed to connect up to four sites at 384 kbps or three sites at 512 kbps using the H.320 or H.323 protocols. Preset camera positions to focus immediately upon the speaker. Full-motion video at 30 frames per second and broadcast quality beginning at 512 Kbps is possible. Administrator functions are eased with an address book, which records numbers frequently dialed, and an embedded Web server handling diagnostics and simple software upgrades over the Internet. The system can connect to two local cameras, one document camera, VCR or computer source and display the information on two local and two remote displays such as televisions. It is possible to scale the system to handle ten simultaneous sites in a single session [www.polycom.com/products/video_large.html].

Figure VI-2:   Root 258 Classroom Pictures.

Each classroom contained four televisions and multiple monitors in various layout configurations throughout the semesters. Room 262's television layout was not modified since it utilized a large (and essentially unmovable) container for two televisions in the front of the room. One sixty-inch rear-projection television was mounted directly above a second. Each shared a single plastic front sheet that extended

from the top of the container to the floor. There was no practical reason to disassemble the display system. Many of the "lessons learned" regarding classroom configuration was already gleaned from the class instructed from Root 258. Two large forty-eight inch flat-panel televisions were in the rear of the room. Based on requests, after several recording failures, a small five-inch television was attached to the VCRs to monitor the recording process. The small monitor was normally in front of the camera operator or instructor to provide situational awareness of what was being recorded. Finally, the video stream, which was sent to the distant site, was split to the classroom's local computer through an ATI-All-In-Wonder card. Hence the local computer was able to connect to the Internet and test multiple software applications in the Distance Education arena.

Root 258 was much more configurable than Root 262. There were two large televisions in the front of the room, with one suspended from the ceiling. Halfway back in the classroom was two televisions mounted to the ceiling, but in a back-to-back configuration. The positions of the televisions as well as the information they displayed were modified throughout the semester. For instance, when the class was instructed from Camp Pendleton, three monitors were placed across the back wall and the students were turned around. The students were now able to have one monitor on the instructor, one monitor on his computer display and another monitor to see what was broadcasted from the local classroom to the instructor. Since all three monitors were next to each other, it was a simple matter to take in all three data feeds and concentrate on the one that was most important at the time.

The PictureTel 4000 room control cabinet in the front of the room allowed the instructor to control which feed was displayed on the different televisions. The two choices were near and far. The near feed was from the local classroom and the far feed was from the distant site. The feeds can also be controlled from a remote technician room located several rooms away in Root Hall. Once the near or far feed was slated for each monitor. The local system controls can choose which camera to display for the near feed. The operator can choose between the camera in the front of the room which covered the students or a walking professor, camera in the back of the room which was the standard camera for the instructor in the front of the room or usage of the dry-

erase board, computer screen capture or document camera. The operator can also display each feed with an additional picture in picture display.

The remote operator also controlled audio. The operator can mute the near and far stream. Hence, if the instructor did not want the distant site to hear something, the audio stream can be disabled. Also, if there was a distracting noise at the distant site, the audio can be stopped. However, individual sound levels were controlled from their respective renderer. Each television had its own volume control. Also there was a set of speakers in the front of the room which was controlled by knobs located on the speakers.

Microphones were evenly spaced throughout the front of the room and on each table. The one in the front of the room was by the instructor while the others were in front of students. The gain was controlled by the PictureTel 4000 control cabinet and was preset by the school's technicians for maximum efficiency. The microphones were not configurable and always on, even if the sound that they were absorbing was not transmitted or saved.

The entire front wall of the room in Root 258 was covered with a whiteboard for use with dry-erase markers. The instructor was able to write notes and draw diagrams in an impromptu manner with almost forty square feet of writing space. Root 262, however, had a small three-foot by five-foot whiteboard while the rest of the wall was painted in light green to allow the use of Chroma Color video display. The smaller board size created some difficult situations that are covered later in this chapter under "lessons learned".

Both classrooms contained two rows of overhead florescent lights. There were only two light switches to control the two rows of lights in each room. One switch turned on or off an entire row. Hence, the only options were to have all of the lights on, half of the lights on either the right or left side of the room on or no lights on. The lack of light controllability lead to some situations where it was difficult to see certain items that were pointed out by the instructor from his computer and relayed to a television monitor.

Both classrooms also contained a single computer in the front of the room that had Internet access. The same network also had an Apple Airport wireless cell covering the classroom. The instructor used his own laptop for most of the presentations. It contained a wireless PCI card to connect to the Airport cell and merely needed to have a monitor cable connected to its HD-15 port. There was also a RJ-45 connector. An extra monitor cable connected to the near feed for this purpose. Some students also had laptop computers with IEEE 802.11b cards that allowed them to connect to the wireless network.

The instructor had an additional switch control under his desk to control which computer monitor was utilized for the near feed and where it was displayed. He had to manually choose between his laptop and the classroom workstation. Also, in Root 262, he had the added ability to display the computer monitor as a backdrop to the classroom camera. Using Chroma Color drivers, the monitor appears in the feed where light green was displayed on the classroom's front wall. The effect allowed the instructor to be composed within the image displayed by the computer.

The fax machine was also utilized to send handouts and impromptu drawings that were too small for the document camera. Although used as a reserved means of communicating handouts, it was an effective device. If person at other end cannot see document or needed a copy of a handout, the paper can be faxed to the student. The student can also fax in assignments if it is handwritten or not available in electronic format. The fax machine was also important as a back up means of communication if Internet access was not available for the distant student but a phone connection was possible.

A technical support staff of one to three people was typically available while the classrooms were in use as well as during other times throughout the day. If assistance was needed, the instructor would interrupt class to walk two to three rooms down the hall to gain their assistance. If there was a recurring problem or they were asked for assistance before class, at least one technician can stay in the classroom during the session. They also had the ability to control classroom and streaming settings for the PictureTel 4000 system from their offices.

Each room had environmental controls such as air conditioning configurable by the instructor. There was a remote control for the air conditioner that allowed the room to dramatically cool if desired. The air conditioner can be noisy but was not too distracting. Once the room was cooled, the air conditioner can be shut off until it heated up again. Turning the lights off also helped control the classroom temperature. The instructor was required to master the live adjustments, many variables and arcane procedures during instruction. Hence, room configuration and equipment usage presented multiple and significant teaching impediments. A summary of these problems is presented in Table VI-1.

• Small space	• No user manual for VTC equipment
• Non-intuitive equipment	• Equipment easily disabled by accidents
• Limited support staff	
• No production staff	• Limited digitalization capability
• Overtasked professor	• Proprietary MPEG-2 encoder
• Poor overhead light configurations	• Limited recorder monitoring
• Wrong ink color for whiteboard	

Table VI-1:   Teaching Impediments in VTC.

b.   Delivery Type

Content for the classes was through all media available. The instructor used handouts, handwritten notes, PowerPoint, HTML Web pages, 3D Applications, VRML enabled browsers, Internet resources, LAN resources, whiteboard, textbook references and verbal explanations with hand gestures. Each method of delivery was deliberately used and evaluated to elicit maximum understanding of the presented content. In many situations, more that one media was utilized to present a topic in the hope of increasing the students' comprehension.

c.   Student Body

The students that attended the classes were full-time graduate students with little to no prior background in the subject matter. They all were military personnel,

federal employees or contractors. The average student age was 28-years with a range from the low twenties to high forties. All students were proficient in English and were experienced computer programmers. The majority of the students were male with only one female attending each class. Finally, approximately a third of the students were foreign nationals who spoke English fluently as a second language.

2. Tools Utilized

Three video tools were extensively studied and one was briefly observed due to its proprietary nature. The three video tools were PictureTel 4000 hardware, MBone Toolset software and JMStudio software. A proprietary hardware-based MPEG encoder and transmitter was examined to look at remote location control issues. The diversity in these tools covered the spectrum of commercial, software, open-source Java implementation and hardware implementation possibilities.

a. PictureTel 4000

The strength of the PictureTel system is that it allowed a single operator to control camera systems and switch viewpoints within the classroom. In fact, this strength was capitalized upon with the other tools by having them accept the near view feed of the PictureTel system. Other camera schemes were possible, but since a system was already in place, there was no need to add extra hardware.

The PictureTel system fed images and audio directly into a session feed that includes ISDN lines, two SVHS VCRs and a video capture card location in the classroom computer. By having the system split its data to the multiple output resources, a class can be simultaneously streamed to another site, captured to tape for future playback and captured in a digital format for Video-On-Demand applications. In this manner, both synchronous and asynchronous sessions might be created. By encoding using SVHS tapes, a baseline for future content digitalization is also established. If a new codec is later developed or the school decides to work with a new format, there will be no further generation loss or digitalization artifacts in the creation of the new digital media. Also, since many current proprietary formats cannot be transferred to another proprietary format, these tapes provide a guarantee for future capabilities and capacity.

b.   *MBone Toolset*

Use of the MBone Toolset is covered extensively in Appendix B. This tool was used to test classroom connectivity to the MBone as well as transmission to a civilian academic campus. The tool was used to attempt connectivity with George Mason University. The intent was to allow instructors and students to view class presentations from the Naval Postgraduate School. The tools were not used for college credit but merely as a proof of concept. Also, several MBone connectivity-related issues were raised in the use of these tools. Unfortunately, only a few sessions were possible after the technological and pedagogical challenges were overcome. However, this remains a promising area for future work.
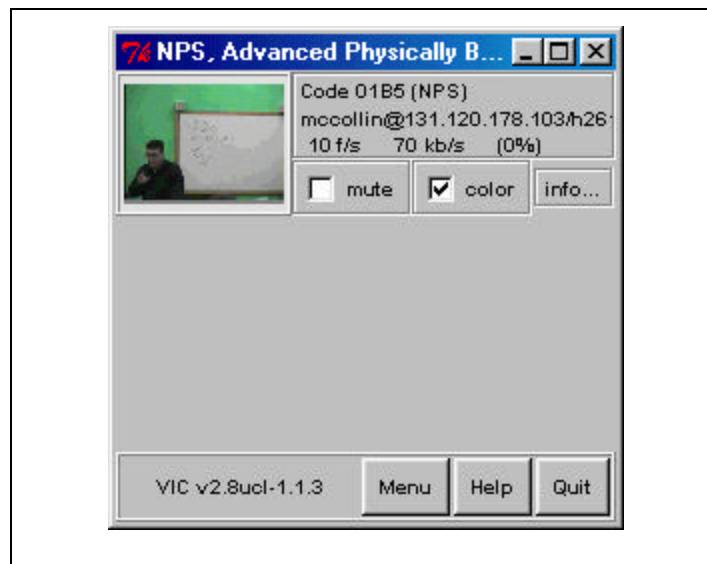
Figure VI-3:   Advanced Physically Based Modeling VIC video panel.

c.   *JMStudio*

JMStudio was used to explore capture and transmission techniques using an open-source software tool. This application was also a baseline to test different Java JMF applications, and became a litmus test to determine if experimental Java programs might work with new systems of hardware or software. The use of JMStudio is documented in Appendix A. JMStudio's ability to connect to another JMStudio client

was explored using the NPS LAN. Capture was performed using the classroom computer with an ATI All-In-Wonder video capture card. File size for different capture formats was further recorded.

d. MPEG encoder

The NPS distance-learning supervisors purchased a proprietary hardware MPEG-2 encoder that allowed control, capture and transmission of video feeds from the technical staff's office. Due to legal reasons, the company's name was not disclosed. Unfortunately, a proprietary software client was necessary to receive the video feed. The company's business model was to sell the hardware at a discounted price and then charge several hundred dollars for each software client despite prior claims by the company that any MPEG player might render the images. Extensive tests were performed to determine if the claim was valid. Only the company's player could render the feed. Hence, this alternative was rejected as a valid choice since the equipment is essentially useless without the expensive proprietary player. The MPEG-2 encoder company was unwilling to answer queries about the return and reimbursement of the improperly contracted equipment.

C. EDUCATION CYCLE

The *education cycle* is a term informally used by the author to describe the overall process of transferring information from an instructor to a student. A confirmation loop is used to close the process by using mechanisms to verify knowledge transference. Distance education is an extension of the traditional *education cycle*. As discussed in Chapter II, distance education needs to capitalize upon the strengths of the traditional education system with the new technology available. This study addressed how the technology affected methods of student assignment, how the assignments can be fulfilled, demonstration of knowledge, content-delivery methods, level of effort for producing different types of presentations, and finally a summary of the different types of distance education classes.

1. Methods for Student Assignment and Fulfillment

In the traditional classroom, written and oral communication is the primary means of assigning work to students. Most classes begin with a paper syllabus, outline and goals handed to the students in the classroom. Some instructors may elect to write this information on a chalkboard or dry-erase board. An instructor may choose to deliver the information orally, but studies have shown that this is one of the least effective methods of directing the class. Also, there is no definitive reference that can be used to guide the class's progress. Finally, the instructor may post assignments and solutions in a public location so that students may go to the physical bulletin board at their convenience.

Traditionally, students return completed assignments in a paper format as either handwritten, typed, word-processed or printed output from specialized applications. If a student needs clarification of an assignment, verbal questions appear to be a primary means of gaining extra instruction. The instructor may respond verbally immediately or with a clarifying written explanation during the next class. In a seminar class, an instructor may assign readings where the assignment is fulfilled by having students discuss the topics in an open forum.

With the extensive use of computers in the distance education environment, the traditional methods of work assignment and fulfillment may be adopted to new media. Such technology allows for spatial and temporal differences between the instructor and the students and speeds the flow of information. For instance, correspondence classes can utilize telephones and postal service. While these methods may work for a small number of students in a well-established curriculum, it does not allow rapid feedback and can delay the learning process. The Internet accelerates the rate and ease at which the information can flow from one location to another.

The main strengths of the Internet are the ability to transfer electronic data immediately from one location to another no matter where the involved sites are located in the world. The rate of retrieving data depends largely upon individual connection speed. However, most forms of Internet communication have been optimized for slow connections. Also, data may be placed in formats tuned for different bandwidth consumption so that users can select the format that is optimized for their connection.

When video is placed on the Internet, it is common to place it in formats acceptable to phone modems, cable/DSL modems, ISDN modems and local networks. This allows for synchronous and asynchronous viewing of lectures over the Internet.

Synchronous written communication can be realized in the form of text relay servers sometimes referred to as "Chat Servers". The servers allow for a large number of people to interactively comment and discuss class content. One class used the chat server as a background means of communication while the instructor lectured and answered classes verbally. The multiple channels of communication reduced distractions from the primary means of communication. Options exist for classmates to help each other as well as have the chat moderated by a Teacher Assistant (TA). The instructor can also review the chat during breaks or after class. Finally, a written record of the students' comments and questions can be maintained for future improvements in the course presentation and content.

The servers also provide asynchronous written communication in the form of electronic mail. Due to the ability to attach files to the correspondence, e-mail is a primary method for handing in assignments or clarifying information about an assignment. The teacher and students can view the correspondence when it is convenient for them. Handwritten assignments can be scanned or e-faxed. However, strict standards and procedures for digitizing the content need to be established since the quality may otherwise be intermittent and questionable. There is also a possible bias that people with better equipment might have a better presentation of their assignment. For this reason, fax machines may provide for instant transfer of paper reports with poor but usable display resolutions. However, serving reports via a web site allows high quality transference on demand.

Finally, a physical bulletin board may be replaced with an electronic bulletin board. Servers with modems may be used for small classes or servers with Internet connections allow for large classes to access information posted by the instructor. Web servers can contain class notes and assignments. These servers create an instant reference for the students. File Transfer Protocol (FTP) servers allow for the downloading and uploading of large files so that the class can exchange files with the instructor as well as

upload completed assignments. File uploading may also be accomplished using a Posting Acceptor that allows the client to use Hyper Text Transport Protocol (HTTP) and overcome some firewall restrictions. Since firewalls restrict many Ports except for Port 80 (HTTP), the Posting Acceptor might be the only solution for file uploading. Additionally, Newsgroup servers, Active Server Pages (ASP), Cold Fusion Pages (CFM), Java Server Pages (JSP) or Computer Gateway Interfaces (CGI) pages can allow students to post comments and follow the discussions in threads. Hence, the students may learn from each other's experiences and decrease the workload on the instructor. Since the instructor also has access to the threads, he may moderate the group.

2. Methods for Instructor Feedback

Traditional instructor feedback occurs in an asynchronous manner. After an assignment is handed in, the instructor normally corrects the papers during a time separate from class time. Depending upon the media used and the formality of the corrections, the instructor may use hand marks, a separate sheet containing comments, e-mail or even computer-generated remarks. One class covering expert computer systems had an expert computer system grade programming projects. Verbal comments may also be used but are normally reserved for the entire class as a summarization of the assignments as a whole.

In the distance-education arena, such comments usually need to be in an electronic format. An instructor may still print out the assignments, but then might increase everyone's workload by taking an electronic document, printing it, making hand-marked comments, re-digitizing the corrected papers and then transmitting them back to the student. There are numerous applications available that allow the entire correction process to proceed in a purely digital format. For instance, Microsoft Word allows documents to have sections highlighted, stroked-out or commented upon when placed in a review mode. In this sense, the Word document can have all of the hand marks placed in a digital format. Additional pages added to the end of the document by the instructor may be used to summarize the paper's grade and content. Furthermore, the word processor can aid the instructor by highlighting incorrectly spelled words, misused grammar and reaching a specific word count.

Verbal comments can still be utilized in the traditional manner by using video conferencing. However, new technology again lends itself to new and innovative uses. For instance, each assignment may have the professor create a digitized video of the assignment's demonstration and grading. Now, not only can the end results be displayed but the processes of how the results were created are also recorded. The added information can now be used for other forums such as instructing student teachers on the proper method of grading an assignment.

3.   Demonstration of Knowledge

The distance education environment also raises the challenge of how to gauge the level of knowledge gained by the students. Traditional classrooms may gather all of the students for a single test taken at the same time in the same location. These methods aid in reducing the number of variables that can cause differences in test scores, for instance, if one student had more time to study than another, or if one was in a more relaxed setting than another. Another method is to have individual students present reports or respond to questions in a verbal interview. By isolating the test to a single student, other students do not have the ability to copy others work and the instructor can concentrate on a more precise evaluation of the student's knowledge.

In distance education, the gathering of students in a single location for a test at the same time may run contradictory to the reason the class taken. One reason for the attendance of a distance-education class is the inability to travel to the place of instruction. Also, there may not be a single time where all students can gather together. Some may argue that distance education is supposed to overcome these restrictions for the normal class and, therefore, needs to overcome these same restrictions for the evaluation of the students.

This limitation may be overcome via several methods. First, computerized testing has gained much popularity in recent years. A school may purchase or develop their own testing software that allows students to cycle through multiple choice, short answer or essay tests at their convenience. Another alternative is when a third party is used for student testing. Companies such as Microsoft and Cisco have turned to third parties such as Sylvan Prometric [www.2test.com] to conduct testing for them. The third party

company may have thousands of testing centers around the world that are monitored with standardized equipment. By using a third party, the primary company or university ensures integrity in the testing process.

Second, if time constraints are not prohibitive, word processors or customized test-question database applications may be used to generate text documents that examine a battery of topics. This model follows the traditional "take home" test. Students are allotted so many days to complete the test and then hand it in. The "handing in" can be through electronic means and follow the examples illustrated in assignment fulfillment.

Finally, a hybrid combination of these methods may be used. For instance, it is quite likely that a distance-education class offered at a university has a local audience as well as a remote audience. In this situation, everyone may not perceive "fair" and "equal" assessments equally. If the local audience must gather together for an exam, they may desire that the remote audience attend the session locally on that day. If an exam is proctored remotely, the exam similarly needs to be proctored locally. This mixed environment usually leads to the acceptance of the most relaxed situation. If only one group can be proctored, then no groups should be proctored in order to promote a perceived equal environment and evaluation.

4.   Content-Delivery Methods

The distance-education environment leads to several questions about course delivery methods. "What content should be displayed for students?" "How can the professor use the new tools to create analogies?" "How can an instructor best display facts?" These questions all revolve around how to use the distance environment and tools to maximize the students' education.

Using different types of content depends on who steers the presentation of classroom material. For instance, a *lecture* only has the instructor talk about subject. *Interactive lessons* have students help steer conversation and teaching. *Demonstrative class* normally has the instructor demonstrate how to accomplish something and then followed by the students' imitations. By classifying the direction a class may take, the instructor will have a better idea of what information to display.

During a lecture, the instructor may communicate emphasis with hand gestures and intonations. If anything is written to a board, the image is normally transmitted to the students. In interactive lessons, the audience as well as the instructor will be displayed to the remote audience. There may be the possibility of showing the remote student to all students as he or she asks a question. The extra information such as facial and hand expressions can aid in answering the question. Finally, classes using demonstrations need the ability to show the instructor's computer monitor or device he is using for the demonstration, as well as have a feedback mechanism for the students attempting to imitate the feat. The demonstration model is the most bandwidth-intensive and technically difficult approach to achieve.

Naturally, the above models may depend upon the human presence involved in the Distance Learning Environment. So far, the models have assumed that there is unlimited personnel and technical knowledge. In reality, the teaching model may have to be modified to meet actual limitations. Three models can show the granularity of the real world limitations.

First case: the professor runs everything. This is the worse case scenario where the instructor must not only be proficient at his specialty but also an expert at using the distance-education equipment. The challenge may seem daunting and lead many professors to avoid the distance-education arena. The instructor will need to be a master of multitasking but, even so, may still reach a point of task saturation. While attempting to answer questions, provide a lecture and maintain the flow of the class, the transmission equipment may be distracting and always present on instructor's mind. While it is possible to conduct a class in this manner, a decrease in educational quality is easily noticed.

Second case: the professor runs nothing. While this situation may seem optimal, inexperienced or untrained personnel may find it almost impossible to track what the professor wants to display or think is important. Even when the professor is not operating the controls, he may still need to give verbal or visual clues as to where the cameras should point and what monitors need to be displayed. Eventually, the operators may learn the instructor's patterns or follow previously created scripts to allow the

maximum transfer of information with the least amount of strain upon the professor. This approach can be very effective, even when technical difficulties are encountered.

The final model is a cross between the two previous models. The session is conducted by using a Teacher Assistant (TA) with some knowledge of the course but not as much as the instructor. They can specialize in using the equipment while allowing their comprehension of the material to guide them in displaying what is important. The TA can assist camera usage, illustrations on board, switch between document points of view, answer individual questions and help keep the instructor synchronized with the class. The assistant model reduces the instructor's workload and allows them to concentrate on content and the answering of difficult questions.

5. Level of Effort for Presentation

The instructor's workload is also directly affected by the location of the class audience. The traditional classroom provides for a local audience with synchronous learning. This model is the baseline against which all other models are compared. It can be argued that the traditional classroom model is the easiest to prepare for since there is no need to operate special equipment or change one's presentation style. This model may also contain the smallest number of students.

The workload begins to increase as the information is presented to more students, across farther distances, and over different time periods. More students equates to more variables. By increasing the number of students, the probability of having a student that does not understand the material increases. However, once a class reaches a certain size, the additional students may be absorbed within the normal question and answer period allotted for the class. The instructor must communicate to a traditional class within the physical dimensions of the classroom. Once the instructor communicates with students outside of the classroom walls, new issues arise such as camera and microphone usage. Also, communication channels and bandwidth must be constantly evaluated to assure the educational channels remain open. Finally, the ability to preserve class content and provide access offers a strain on resources and effort. The instructor must review recorded content to insure it is in a presentable manner.
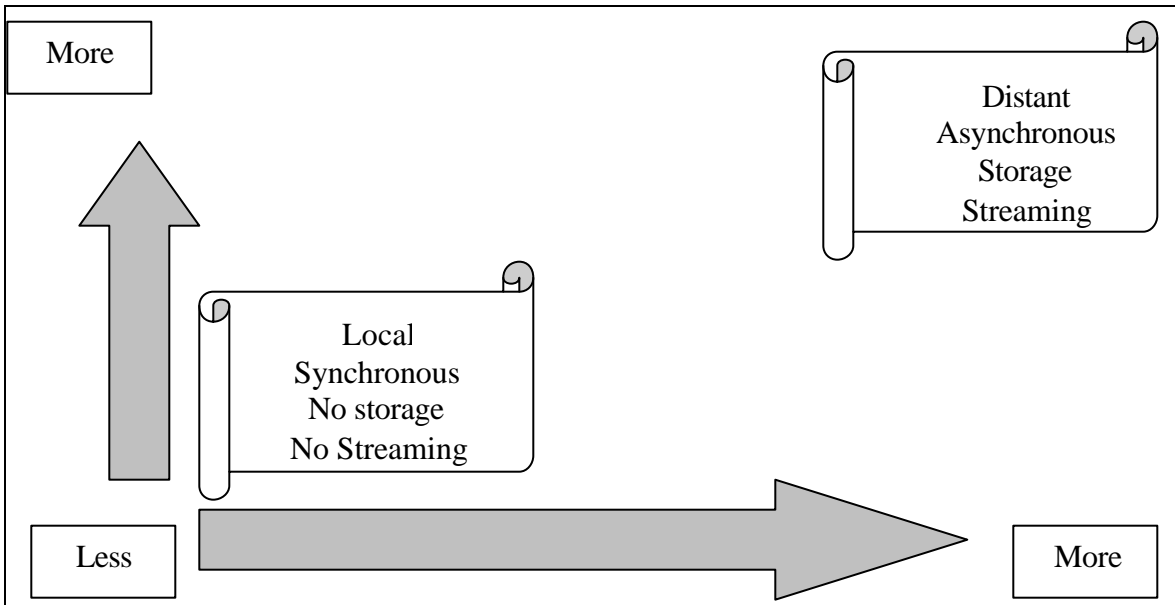
Figure VI-4:   Presentation Effort.

6.   Types of Distance-Education Classes

There are two ways to differentiate the types of distance-education classes. One method is to categorize the class by physical characteristics. The physical characteristics depend on where the instructor is located and where the students are located, as well as the type of information conveyed to the students. The second category pertains to the credit received for the class.

In the first method, distance-education classes can be divided into lectures to different locations, lectures from different locations, lectures to the same location with storage for asynchronous learning, and lectures to different locations with storage for asynchronous learning. Conceivably, the information may be conveyed by neither video nor sound in a class utilizing a real-time chat server or e-mail. Another category is sound but no video, as in a phone conference or Voice Over IP (VOIP) situation. A third category might be video and sound of just the instructor in what can be termed a "talking head." A fourth method is video, sound and demonstrative materials that include multiple types of content. Finally, no real-time connection but usage of asynchronous methods such as e-mail or mail correspondence is a viable model.

The credit received for the class can also distinguish the category of distance education. First, self-improvement classes are normally asynchronous with no recognized credit or separate skill evaluation. Companies such as book retailers may offer these classes to aid in the sale of books. Also, some introductory courses may follow this format to give students a venue for refreshing skills before moving into a more complex arena. In this manner, minimum resources are required for the distance education. An example of this type of instruction is Sun's Java University [http://www.sun.com/java-university/] where participants work through tutorials to gain an understanding of the material.

A second distinction can be realized in the form of certificate classes. Certificate classes have a means of tracking attendance and student progress. Certificates do not normally assign a grade but rather provide the distinction of completion. There is a certain amount of implied knowledge from the attendance and completion of a certificate class that may not be easily quantifiable. Professional certificate classes, however, normally assign a grade or assessment of knowledge attained. For instance, Cisco certification on a variety of topics can be achieved by attending classes to gain the prerequisite knowledge, but then students need to take a test for assessment of the knowledge gained. Even though students have attended the class, they do not necessarily achieve the required test grade without sufficient study and practical experience.

The final distinction is where distance education is used to achieve traditional course credit and educational degrees. The classes are held to the same standards as a traditional class and must meet the same accreditation in order to be offered. Some requirements are instructors who have achieved certain levels of education, board of supervisor approval, university association requirements, and more. Many of the academic institutions that offer distance education still offer "brick and mortar" traditional classes.

D. LESSONS LEARNED

After six months of observing multiple teaching styles and techniques, there were many practical lessons learned regarding what does and does not work in a distance-education environment. Since the NPS DLE system was a commercial product with a

funded support staff, there was an assumption that this system had the fewest number of deficiencies. However, through observation and experimentation, it was shown that there were a great many improvements that needed to be made in order to increase the system's efficiency and effectiveness. The lessons learned were divided into classroom ergonomics, integration of third-party tools, presentation techniques, chroma-color usage, and instructional techniques.

1. Ergonomics

Even though the distance education environment can transcend space and time, the location where it is created is still restricted to the normal laws of physics. Issues such as distance to objects, time to reach or display objects, number of personnel involved in content development and teaching aides available all affect the final product and how the educational experience is perceived.

First, in order to transmit written comments from the whiteboard through the PictureTel 4000 and have the information legible, the instructor needed to use at least four inch letters. If smaller writing was used, the camera can zoom in to the individual letters, but context was lost. The four-inch letter rule allows the viewer to see the full sentence and distinguish the individual letters. The size of the board needed to be at least five feet by five feet. A smaller board did not allow enough room for the instructor to write sentences in a coherent manner. A larger board can be displayed but required the camera operator to follow the instructor through the lesson and display the portions of the board that were of interest.

The control of the cameras, switching displays, changing backgrounds and managing video feeds are manpower-intensive jobs. The PictureTel allowed for a single operator to manage most of the production through a single hand controller. Using this single controller, a student, instructor or dedicated production personnel can manage how the class is projected. Unfortunately, the PictureTel controller used a weak infrared signal to control most of the equipment and unless the controller was pointed directly at the access unit, nothing was affected. Since there was no ability to bounce off of walls, whenever the line-of-sight path was blocked, control was also blocked. This became an issue in Root 262 where the access unit was in the back of the class. It was ideally

94

situated if the instructor was operating the equipment but inadequate for student operators in the audience. The audience usually blocked the signal, resulting in an unsatisfactory situation.

The only aspects requiring control solely by the instructor's station were recording, chroma-color fading and deciding which monitor was transmitted under the monitor display. The tape recording was accomplished by the use of the VCRs just below the instructor's workstation. While it was easy to load tapes, start and stop the recording process, it was also easy for an instructor to accidentally hit the controls with their knees and interrupt the recording process. The adjusting of the chroma-color was best left to the technical staff since it appeared to be more art than science. At one point, it took approximately fifteen minutes to adjust the color when a guest lecturer was wearing a shirt that was too close to the keyed color. Finally, yet another switch box in series with two other switch boxes, was used to decide which monitor was used as the input for the display. There were too many switches with independent failure modes that inevitably led to frequent class interruptions. It was obvious that the there was no human factors evaluation attempted by the local design supervisors.

2. Tools

a. MPEG-2 Encoder

The MPEG-2 encoder purchased for the digital encoding of the PictureTel 4000 classroom streams had multiple negative traits that outweighed the positive traits of the system. The main strengths of the system was the fidelity at which content can be captured, small disk and bandwidth size, and remote controls that allowed central operation of the capture process. With all of these benefits, the only drawback was the proprietary nature of this particular MPEG codec, which meant that it could only be displayed using a proprietary client. While other clients are normally free of charge, the encoder's client-cost prohibited large-scale purchases for general use. Also, Internet streaming servers were not able to use the content for Video-On-Demand (VOD) applications. The proprietary encoder was only able to stream one video feed at a time. In summary, the system demonstrated great benefits but its proprietary nature limited the

95

number of users who might utilize its strengths. Unfortunately, there were only limited attempts to recoup the costs of this unusable system.

b.   JMStudio

Appendix A explains how to utilize JMStudio. JMStudio was mainly used as a capture tool during this project. Since JMStudio was written with Java and distributed in multiple performance packs, it is theoretically able to run on any platform that supports the Java Virtual Machine (JVM). In the beginning, there were multiple hardware and Operating System configurations incompatible with the video-capture capabilities of JMStudio. Once these issues were resolved, JMStudio worked extremely well. Other benefits: JMStudio supports multiple capture formats including AVI, MOV, H.263, a multitude of RTP and more. Also, video size and frame rate were highly configurable. In fact, JMStudio queried the different capture devices to ascertain the format, frame rate and video dimensions that are supported by the device. The software's ability to be so configurable allowed maximum optimization of bandwidth, quality and compatibility.

Another strength of JMStudio is that there are a multitude of independent software clients that can play the formats stored by JMStudio. Apple's QuickTime Player and Microsoft's Media Player can render uncompressed QuickTime formats. Also, JMStudio can transmit the session using RFC-compliant Real-Time Transport Protocol (RTP) headers [RFC 1889]. Hence, RealPlayer can render a session transmitted by JMStudio. Finally, many streaming media servers can deliver the content created by JMStudio. This interoperability is very powerful.

c.   MBone Tools

Appendix B explains how to utilize the Multicast Backbone (MBone) software tools for audio, video and whiteboard control. Use of the tools revealed several interesting facts. First, it was observed that there was almost no or little gain in visual information by increasing frames per second. The added bandwidth required for extra frames was not necessary. Also, when the video tool (VAT) was set to transmit thirty frames per second, the tool automatically down sampled the number of frames per second to match the bandwidth settings. For instance, when the transmission was set for 30

frames per second, it still transmitted about 4 frames per second at 50 Kbps, 12 fps at 100 Kbps, 21 fps at 200 Kbps and 28fps at 300 Kbps for 320x240 image. Also, quick movements were completely lost and added to the jumpiness of the pictures at lower bandwidths. Knowledge of these traits leads instructors to use more deliberate gestures; maintaining effectiveness for both students and Web delivery.

For the classes, the text editor and whiteboard tools were not used. They might have added to the value of the lecture but were not feasible since the instructor was already "task saturated" with instructing and operating the other distance education equipment. Furthermore, PowerPoint slides were displayed from the instructor's laptop computer for this need, and students did not have local laptops for viewing the whiteboard. Also, due to the reduced video fidelity of the MBone tools, reading the classes dry-erase board through the video picture was not possible. If the video image was the only way to display the notes, much meaning may be lost. Also, much student concentration and time is lost attempting to discern words and their context. Overall, training may actually have a negative learning impact if slides are not utilized.

The use of the MBone tools also had several overhead requirements. First, all session participants need to have the tools pre-installed on their computer plus reliable access to the MBone. While connectivity to the MBone may seem a trivial matter, experiences with Internet Service Providers (ISPs) have shown that few yet endorse or support multicast traffic. Hence, if there is no direct access to the MBone, router tunneling may be employed or Internet Service Providers (ISP) may be convinced to enable multicast traffic on their routers [RFC 1112, RFC 1458, and RFC 1075]. While convincing ISPs to enable multicast traffic may be a wasted endeavor, most routers support multicast tunneling. With multicast tunneling enabled, multicast traffic is placed within a unicast TCP/IP wrapper and directed to another router known to have multicast enabled. Using the wrapper, allows the traffic to travel through several non-multicast ISP domains.
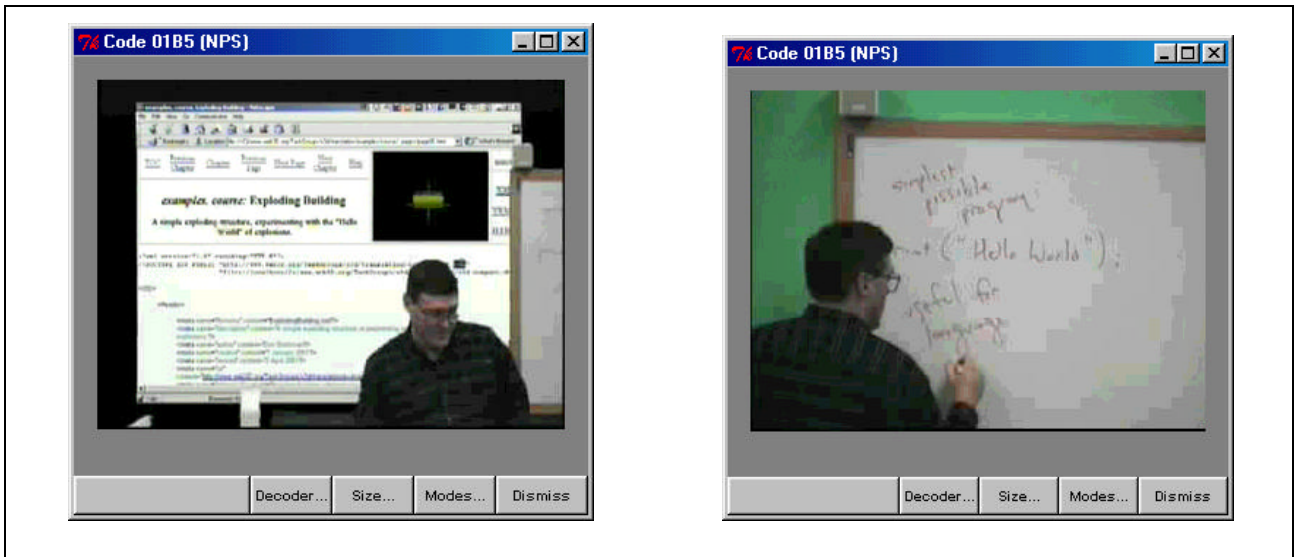
Figure VI-5:   MBone Delivery Tools display of VRML/X3D class.

3.   Presentation

Usage of the PictureTel system demonstrated its strengths and weaknesses in presenting a lecture.   First, while the use of chroma-key color allowed the instructor's image to be superimposed on the computer display, there were many situations where this generated a negative learning experience. By having the instructor occupy too much of the screen, text and computer generated images were blocked from view. Also, if the content was displayed from non-optimized, computer generated pages such as web pages, the text was difficult to distinguish. By keeping the instructor's image small and in the lower right-hand corner of the screen minimized obstruction problems and provided ongoing context for the presented material. Instructor practice is needed for effective gestures. Local students can look at the local monitor to interpret the gestures.

a.   *Display Resolution*

Some applications displayed on the instructor's computer were not easy to see. Items designed to occupy small areas of the desktop such as tool bars were difficult to see even on large-screen class televisions. HDTV may be a solution but it is extremely expensive. The expense is not in just acquiring HDTV televisions, but also compatible cameras, servers, bandwidth and clients. The distance-education experience is also limited to other distance education centers with equivalent resources, which is still a

98

rarity. A more cost-effective method is to have repeating computer monitors throughout the classroom. While the minimum requirement is one monitor for every two students, more monitors and computers are desirable. For instance, a Microsoft Class in San Jose for Microsoft's Cluster Server seated each student with two computers and three monitors. The third monitor was a repeater of the main presentation screen, while the two computers allowed students to display and take electronic notes as well as work through software exercises to gain familiarity with the software.

b. Display Size

If a distance learner is utilizing a computer screen for the reception of the video feed, the image may be too small to see. This is especially the case when the computer is being utilized for demonstrational software and note-recording software. A separate computer or notebook is usually needed to make notes. In this situation, it is difficult to balance small screen with Microsoft Word and the video picture. The addition of software such as X3D-Edit authoring tool and a VRML-enabled browser demonstrated the need for large or multiple video displays. Additionally, student laptop computers proved to be of great value during the class and as a presentation review tool after the class. If there needs to be choices because of the video displays, audio was the most important component of the instructor's presentation. While the visual input added information, unless the instructor was making gestures or pointing out data in the display, there was little additional information added to the presentation.

---

1. Visualization Tools: X3D-Edit application, Browsers with plug-ins.

2. Slides: PowerPoint, Lotus Freelance, and HTML.

3. Audio: Understandable with limited noise, includes student questions.

4. Video: May be small but large enough to convey gestures.

Table VI-2:   Priority of Media Value for Learning Effectiveness.

c. *Classroom Distractions*

The PictureTel VTC wireless infrared controller can only work if directly aimed at the sensor. Since the system was optimized for the instructor, when a student used the controls, the controller had to be held in an awkward position of backwards and off to the side. People were still blocking the path of the beam at times since the beam cannot bounce off of the forward wall. This position also reverses the button map for all users except the instructor, which can be confusing and lead to class interruptions. Also, there was no natural mapping of the buttons to their functions and controls. Hence, the session operator was tied to a position almost directly in front of the instructor with the controls in plain view of the video feed. At times, operator activities distracted from the lecture. This problem is easily fixed through stronger IR beams or a second sensor at the opposite end of the room.

Other distractions were noted during the Advanced Physically Based Modeling class from the distant audience. For instance, there was no technical staff at the distant site. Thus, volume and static generated from the remote audience was self-adjusted or mal-adjusted. Because of this constraint, sounds from the distant site kept coming through in a cyclical pattern until the instructor was able to talk the student through the sound adjustment process. Additionally, there was only a single-channel (half-duplex) for audio. So, when one person talks, the other person's audio is instantly silenced. This can be very frustrating. Also, the connection to Rhode Island incurred a perceived two-second delay in audio communication due primarily to hardware and firewall issues rather than transmission rate. Since both sites are connected directly to DREN, there should have been less than ten milliseconds delay for every time zone.

The camera usage from the remote site was also self-guided. In the case of the Advance Physically Based Modeling class, the camera was centered on one individual. Since the system was designed to display an audience of approximately twenty people on a front and back television, the distant site's picture was of one person's face watching the screen. Since the front television was approximately sixty inches, that one student took a more prominent role than the instructor. In fact, the student was scaled to be approximately three times larger than the instructor. Since he was so large

and the local audience was looking at the television directly under his display, it was difficult to concentrate on the lecture when the distant student moves or make noise. Zooming in or zooming out (depending on the active speaker) is a useful technique.

There were a few instances that were extremely distracting. At one point, the remote student stood up and left while closing the door behind him. Since his picture was one of two big screens in front of the room, it took everyone's attention. A few minutes later, he came back with food for lunch. He prepared the food on the side of the camera view and then tucked in his shirt. He even scratched his hindquarters, which caused the local audience to start laughing. Apparently, he was not aware that his image was in the front of the room at three sizes larger than life and must have had the sound muted on his side of the feed. In this situation, the distant audience's actions became extremely disruptive for the local audience and embarrassing for the remote audience as well.

The opposite situation can also occur in the distance education environment. When the X3D class was instructed from Camp Pendleton, the instructor had difficulty distinguishing who was speaking from a twenty-person audience. The camera view needed to be so wide, in order to see the entire audience, that it was difficult for the camera operator to find the person talking in a timely manner. The instructor also had difficulty recognizing facial signals from students to see if they comprehended the material. This was a situation where too little information was displayed from the distant audience to the instructor.

d.   *Chroma-color and Limited Work Area*

Chroma-color is when two video images are captured, composed and displayed as a single image. The color, chroma, in one video image is keyed for replacement with the full pallet of the other image so as to provide the appearance of a single scene. If an object in the view (such as a computer screen) is not of the keyed color, it occludes the final camera view and blocks readability. Since there was so much loosely connected distance-education equipment in the confined space of a normal classroom, physical items often disrupted the flow of the lecture. For instance, the document camera was blocking the left portion of the instructor's presentation area.

There was also an unneeded hole in the back wall that was a different color than the green backdrop. When the instructor used chroma-key, the hole was extremely noticeable. Also, the whiteboard was to the right of the instructor's presentation area and limited the usable area for the chroma-key overlay to an even smaller area. Clearly, the personnel responsible for the room design and maintenance do not use it as instructors. Repeated requests and suggestions met with only limited and cosmetic attempts at corrections.
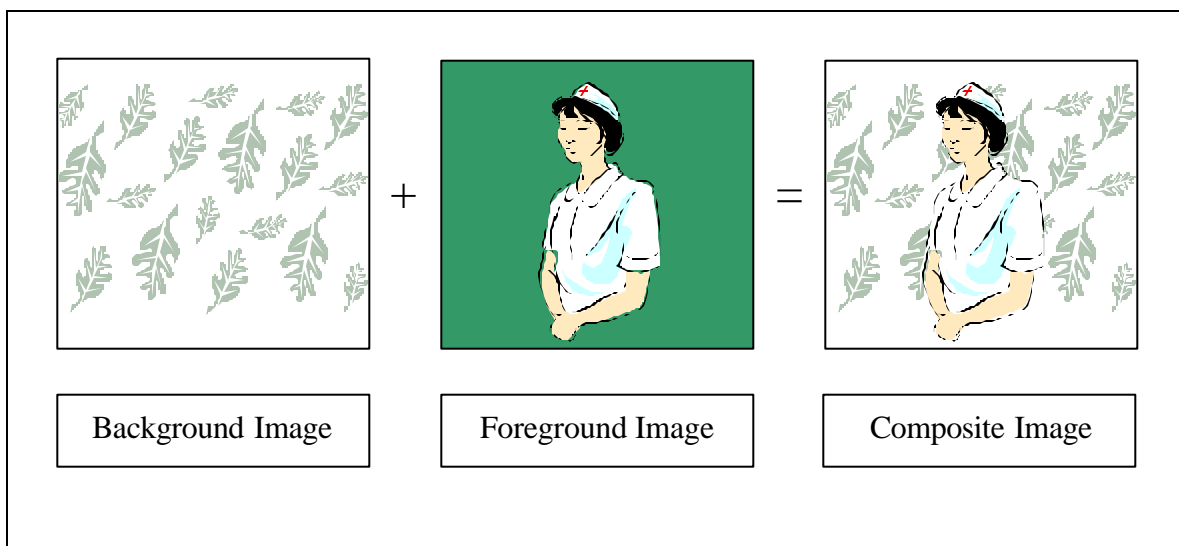
Figure VI-6:   Chroma-Key Effect: Speaker Superimposed on the Background.

The chroma-key capability is dramatically affected by various classroom factors separate from the camera equipment. First, the classroom lighting changed the chroma-key quality substantially. If all of the lights were turned off for the local audience's viewing of the classroom televisions, the full background was eliminated and chroma-key did not work. If one row of lights was turned off, the display showed tearing with horizontal streaks through the picture. Second, shirt color of the lecturer affected chroma-color in an unpredictable manner. One person with a horizontally striped shirt, revealed an image with tearing and partial disappearance of the instructor. When there were multiple demonstrators, perhaps with poor color choice in clothing, the chroma-key had to be re-indexed. This event required technical support and cost time from class. It

can be an extremely disruptive process. Significant improvement can be gained by rewiring light fixtures from two to four switched banks.

Besides limiting the chroma-key area, many of the VTC hardware objects were in the way of the instructor. The PictureTel controls are in a large black cabinet next to the dry-erase board. This cabinet prevented right- handed instructor from writing on the board and then allow room for stepping out of the way. He was forced to stay in front of the board and half turn or exaggerate a step to the right in order to allow the students to see what was just written. The cramped presentation space forced the instructor to be highly conscious of his body and camera location. Once again, requests for improvements resulted in no constructive changes.

This cramped position also places the lecturer next to the session recording controls. The controls on the VCRs were optimized to be in the most readily convenient location. Hence, whenever the instructor moved around his work area, there was a high probability of brushing the controls and interrupting the capture portion of the class. Merely turning around was enough action for the instructor's knee to hit the on/off switch on the VCR and silently cancel the recording session. On several occasions, the class was presented a second time to an empty classroom to restore the unrecorded lecture. These situations were lessened in frequency by using a small monitor that was connected to the VCRs and handed to the camera operator as a primary means of quality control. Also, a small plastic control guard or relocation of the VCR might prevent someone from inadvertently hitting the controls.

e.   *Instructor Movement Limitations*

The instructor was tied to a one-foot square in the front of the room by various factors. If he was operating the camera and feed controls, the small reach of the control's umbilical cord limited him. Even when the camera control was handed over to another, there was still some verbal control of the view by the instructor. In order to overcome this limitation, there needs to be a remote camera operator or an instructor tracking system. A second system also needs to control the instructor's slides and display control. A wireless mouse or equivalent technology may be necessary to allow the instructor to move without being encumbered. A free utility called Pebbles allows the

instructor to control his PowerPoint lecture through a wireless network from a Palm Pilot [www-2.cs.cmu.edu/afs/cs.cmu.edu/project/pebbles/www/slideshow/palm]. However, this method still restricts use of hand motions and board writing.

Additionally, the use of chroma-key further restricted the instructor by limiting the area where he is visible in the composed image. Since the system was composed to a certain shade of green painted on the wall. There was no other area within the classroom where the effect can work. It was also disorienting to reposition the camera's viewpoint when chroma-key was in use because the background remained centered within the camera but only displayed on the green paint. Holes in the wall, poor document camera placement and other easily corrected design flaws almost negated the value of this expensive technology.

f.   Third-Party Distractions

The technology utilized in the distant learning center can be used as a major distraction by third parties with no connection to the class or school. An example was the classroom fax machine that was used to send handwritten notes to the Rhode Island site. On several occasions, companies "spammed" the fax machine by offering unsolicited advertisements such as CCNA certification or Sprint Wireless Phones. Since the phone number was not publicly available, a "Warez" software program that uses a computer's modem to dial a series of sequential numbers may have discovered the number for the fax machine. If a fax machine answers the phone, the number is recorded in a database for advertising firms. Since the fax machine is at the front of the room on the instructor's desk, loud beeps, blips and printing noises caused everyone to stop what they were doing. Everyone, including the instructor, was waiting to see what was coming through the fax. When advertisements were revealed, the entire class laughed and proceeded. The fax machine had the potential to become a daily disrupter to the classroom. Changing the fax machine phone number at regular intervals is recommended.

g.   Organizational Resistance

The support staff was very willing to help but was not equipped for success. Insufficient funding was first among several factors leading to difficulties in the

VTC. Although offered from PictureTel for a fee, the staff was not sent to classes offered by PictureTel on how to optimize the school's system [http://www.picturetel.com/home.asp]. Also, classroom schematics and user manuals were unavailable. New users had to learn the system by trial and error. If the staff was not available, a new instructor can receive a verbal overview of the system's usage and features. If there was criticism of the system or its administration, it was treated as a hostile attack instead of an improvement opportunity. Finally, the system's technical expert was tasked with other functions not related to the VTC. He was normally at a remote location and hard to reach. Additionally, none of the designers or support staff were instructors and had never used to the system to deliver content to remote locations. The designers and supervisors approached every problem from a technological point of view and not from a content delivery point of view. If an item made the system easier to administer but removed an instructor's capability, the new item was incorporated into the system. Professors or classroom users never reviewed changes to the classrooms. So, modification often continued in an untested fashion. Verbal and written recommendations by faculty and students were typically ignored.

4. Instructional Technique

The instructional technique for distance learning is different from traditional instruction in several areas. First, there is the added overhead of class production. The instructor needs to be conscious of camera views and how information is displayed across the allotted bandwidth. Distance learning definitively has a larger overhead than normal teaching in presentation, delivery and technology familiarity. Since there is so much overhead, the instructor may have to ask some hard questions to temper the environment for what is necessary to have an effective session. She may have to ask, "What is the value added?"   "Is there a need for a "talking head"? "Is voice enough or are the instructor's facial and hand expressions necessary to convey ideas?" All of these questions were explored in the NPS Distance Learning Center. Instructors new to the technology should plan on extensive preparation and training.

a.   *Video Feed*

One may wonder if a display of the instructor through a video feed is necessary for the conveyance of information. It was discovered that the students felt a sense of familiarity by being able to see the instructor. The facial expressions and hand gestures added extra meaning to the verbal seminar. Also, the instructor can point to a particular section of a slide or underline what he was talking about with hand gestures. Even so, hand gestures are difficult due to mirror-imaging and loosely coordinated video composition. The viewing of the instructor offered some value but the least amount of added information to the lectures.

However, the value of seeing the instructor may go beyond a pure transfer of information. The students begin to establish a certain report with the instructor by being able to see his or her face. By placing a face and human actions behind a lecture, the student gains more information about their learning environment. Human nature makes some people change their interaction with the class if they can actually see an instructor and know that the instructor can actually see them. For instance, it is much easier to miss a class if the student does not associate the text and tutorials as being associated with a person. However, if the person has to see a disappointed instructor when they do not attend class, their psychology may force them to take the extra step to attend the class and not have to deal with a disapproving human countenance. Facial expressions are a powerful motivator.

The video feed, however, acquired a new meaning when viewed from the instructor's point of view. For the distance instructor, it became critical for the instructor to see the audience's reaction to his words. It was necessary for the instructor to gauge the effect of his words, in order to tailor the class to the learning speed and style of the current audience. In an ideal world, all students understand every word spoken by the teacher. Since education is very personal, a conscientious instructor constantly evaluates the effect of their teaching style on the students. A blank stare can express more information than a student who pursues question after question about a topic. The video feed to the instructor becomes an important link to the flow of information and may be the deciding factor if a teacher disconnects from their class.

The usage of the video feeds was also dramatically different depending on whether the instructor was in the classroom with the audience, or else projecting from a distant location. When the instructor was in the classroom, the placement of local monitors and display of the instructor were not important and the main use of the televisions was to see a repeater of the computer monitors used by the lecturer. However, from a distant location, it was extremely easy to disconnect students from the instructor by showing the wrong feeds. Students quickly became frustrated if the instructor was talking about a slide that the students could not see. Students immediately informed the instructor if he was showing a software demonstration but the class was not able to see his mouse movements. The correct usage of the multiple feeds was extremely important for the class's attitude, engagement and learning.

Multiple feeds also became important for the transfer of knowledge. A simple test was performed where the televisions in the front of the room were disabled. Even though the audio feed was working, the students had a degraded sense of what the instructor was discussing. Adding the video feed of the instructor during a demonstration also provided more information necessary to convey the specific operations performed using the software. Finally, audio with video of the computer monitor running the demonstration software delivered an acceptable level of information so that students might learn how to perform the tasks. Adding a video of the instructor aided in creating a sense of presence in the virtual environment, even though it was not strictly necessary for the information. After much experimentation, it was discovered that the maximum amount of information can be conveyed with a full audio track and video feed of the moment's most important sources of information. Determination of information importance was based upon anyone asking a question, anyone answering a question which did not require use of the computer, showing the computer monitor when software was being demonstrated, the instructor's computer-generated slides, and the instructor's impressions as he verbally answers a question.

When taping the classes for future viewing, the need for visual feedback to the camera operator became essential. There are so many variables associated with the system that a display of what is being recorded needs to be set aside and rendered for the system operator. Some variables are as simple as whether a tape is consumed before the

107

lecture ends, or as complicated as having the system place the wrong signal feeds into the VCRs. It is conceivable to have taped a class for a full session, only to discover that the entire tape consists of the audience at the far end of the feeds watching what the instructor is doing. Such lectures need to be re-taped or lost completely. The loss of class feedback on the tape and the overall time involved is very frustrating and occasionally unbearable. Staffing priorities need to be adjusted to provide regular support during recording endeavors.

The visual feedback is also extremely important for instructors since they are attempting to stand within the camera's view. If a teacher is attempting to visually demonstrate something with his hands, it certainly does not help on tape if the instructor steps outside of the view of the cameras. The instructor can greatly help to insure the quality and flow of the presentation by monitoring the video feeds. Additionally, when the MBone video tool (VAT) was used on the classroom computer, the session operator can see if the information was transferring into the digital media and how degraded the presentation was. Throughout the two semesters, there were different times where each method of observing the display was used to catch an incorrect display configuration. For maximum effectiveness and reliability, the use of multiple monitoring methods needs to be normalized and incorporated as part of a standard session routine.

b.   *Written Media*

Another primary method of transferring information is through written media. The instructor typically generates the content before class in the form of a printed handout or PowerPoint slides. Alternatively, the instructor might generate the notes during the class lecture. Most classes appear to have a mixture of the two styles. For the classes taught in the DLC, most notes were generated before class and either e-mailed to the students or placed on a Web server for on-demand download. PowerPoint slides were used extensively during the second class and they were also available for download. Notes written on the board were digitized using the camera.

The instructor used the dry-erase-pen whiteboard to draw notes and images. While a variety of ink colors were chosen, the use of green ink was almost unreadable over VTC due to the color matching the chroma-key green. When sent over

the MBone tools, it was completely unreadable. After the instructor removed the detrimental, green markers from the classroom, they were normally replaced with more green markers. Differences in color often were not distinguished on the MBone tools using 15 frames per second, 320x240 and 300Kb feed. Less than 15 frames per second became jerky and hard to follow the video which was no longer matched to the sound. The MBone tools also contained an excellent whiteboard tool but then, unfortunately due to classroom configuration, the local audience could not see the display that well. The tool was equivalent to a scaled down drawing program, which is good for simple annotation of prepared diagrams or simple preparation of basic line diagrams.

For the Physically Based Modeling class, the instructor experimented with using PowerPoint as a whiteboard. The use of this application had a unique effect since the people on the other side can now see notes more clearly. It was typed with a format that was extremely readable and could be e-mailed for even finer fidelity.   The participants on the distant side of the video feed had difficulty reading it in draft mode but could easily read it in Slide Show view. Also, the notes could now be added to course content and websites. It made for a permanent, more understandable media than the standard website. The use of PowerPoint in this manner was also unique in that the instructor notes, which are normally erased during the class, were preserved. The instructor kept them for current and future class downloads. It improved the quality of the notes and provided a validated permanent record.

However, there was a learning curve involved for the use of pictures and animation. It required a certain degree of expert familiarity to allow the instructor to begin drawing math equations or generating animated sequences. Once the teacher learns how to perform these tasks, the use of PowerPoint becomes significantly easier. The instructor compensated for the extra time needed to create equations by generating the math symbols before class began.

c.   *Movements and Gestures*

The instructor's delivery of words and body gestures almost always needs to be slowed down for the capture equipment. In the traditional classroom, an instructor may hold up a piece of paper for a second and make a comment about it. In the distance

education classroom, the same gesture may never allow the capture of the displayed paper. A general rule for deliberate gesturing is to show slides or notes for at least five seconds. This rule helped to insure that the class remained synchronized with the instructor. Naturally, words needed to be enunciated correctly and gestures needed to be bold and then held. Fortunately, this approach is more dramatic than artificial or unnatural. These techniques tend to emphasize good speaking practices and do not distract from the experience of local students.

5.   Inadequate Campus Infrastructure Support

The campus infrastructure was not properly arranged to support the storage, bandwidth or distribution of the lectures taped and transmitted through the VTC. There was no coordination between the VTC product and school's network infrastructure. For instance, there are no servers available for storage of large digitized video files. There are no servers with streaming media capabilities for the delivery of video on demand. Until it was made a product of this thesis, there was no open-source or instructor controlled method of capturing digitized media. Tape capture was the only method available. Additionally, a firewall blocked and limited access to the school's OC-3 connection attached to the Defense Research and Engineering Network (DREN). Streaming media was not available to/from other academic institutions or students except through dial-in modem connections. The limited bandwidth of these connections makes the lectures almost ineffectual. Finally, many streaming formats, such as Real's `.ra`, displayed jerkiness as it was processed through the school's firewall.

- Limited multimedia digitalization ability

- Poorly used resources

- Firewall limitation of sending streaming video

- Firewall limitation of receiving streaming video

- Improperly trained support staff

- Lack of diagrams and documentation

- No large-scale computer data storage capability

- No multimedia streaming servers

- Bureaucratic resistance to network improvements

Table VI-3   NPS Distance-Learning Classroom Deficiencies.

E.   CONCLUSIONS

The study of two distance-education classes in a well-established commercial solution revealed many interesting facts. The combination of available options revealed an almost staggering number of configurations available for the educator who desires to participate in this arena. However, while this area of education is exciting and on the technological edge, there are many solutions which are not easy to use and require much training. Furthermore, institutions need to be vigilant of unfounded claims by companies providing proprietary hardware and software. There now appears to be a number of open-source and open-standard systems available in order to avoid this pitfall. Finally, the initial resources involved in generating distance education content are substantial when compared to the standard classroom. An instructor needs to be aware of the added work involved in a DLE so the number of necessary tasks does not overwhelm them. A well-trained technical support staff is easily justified in such an environment.

THIS PAGE INTENTIONALLY LEFT BLANK

VII.   DR. FRED BROOKS TURING LECTURE CASE STUDY

A.   INTRODUCTION

The Fred Brooks case study was designed to explore multiple digitization methods available for a workstation computer. The requirements for utilizing a standard commercial-off-the shelf (COTS) computer allow the results to be highly portable, repeatable and minimize costs. Capture cards and devices attached to the computers were commercial devices representing a cross section of low-to-mid range capture hardware. Higher-end devices perform most of the capturing and transcoding using hardware solutions within the devices themselves. Since most hardware solutions are not software configurable, they are excluded from this discussion.

The primary goal of this study was to digitize a historic fifty-minute lecture presented by Dr. Fred Brooks during the Siggraph 2000 Conference, after he accepted the most distinguished award in computer science, the Turing Award. Multiple software codecs were compared to discover any hidden qualities as well as decide which ones might be optimal for different employments. Additionally, web pages were designed by Juan Gril and Dr. Don Brutzman to wrap the multiple videos in a user-friendly environment. The web pages also contain the original PowerPoint slides used for the lecture called Design of Design, which were graciously provided by Dr. Brooks. Different form factors were explored to evaluate the utility of the web pages' controls. With the trials completed, Dr. Brooks' lecture is available on the Internet as well as Compact Disk (CD) in multiple formats with the lecture's slides.

A second goal for this study was to lay the foundation for the capture of 250 hours of content presented at Siggraph 2001 in Los Angeles, CA. Using a single lecture and digitizing it in multiple formats developed optimized solutions developed for the week in Los Angeles. Approximately sixty-five software codecs were explored for their strengths and weaknesses during the Fred Brooks case study. This information became crucial for the final techniques used at Siggraph.

B. CAPTURE SOLUTIONS

Many types of capture software and devices were explored for this study. The hardware consisted of AverMedia's TV98 Capture Card, Winnov Videum 1000 Capture Card, D-Link DSB-C300 USB Web Camera, Logitech Quick Cam USB Web Camera and Belkin F5U208 USB Capture Cable. The software consisted of AverMedia's TV98 Proprietary Software, Ravisant Technologies DVR Software, GNU AVI2MPEG, Microsoft's Media Encoder and JMStudio. Although each software was exhaustively tested with each piece of hardware, a solid cross-section of results were obtained by mixing the software and hardware solutions. In order to maintain as much consistency as possible, captured content was attempted at thirty frames per second with stereo 44.1Khz audio sampling. Any deviations from these settings are noted through out the chapter.

1. AverMedia

AverMedia's TV98 Capture Card was installed in a Dell Dimension 4100 WorkStation with a 1GHZ PIII CPU and Windows 98 Operating System. The card was originally placed inside a Windows 2000 Professional machine, but the software drivers downloaded from the company's website were still "beta" versions and repeatedly caused the system to stop functioning. By utilizing a Windows 98 machine, the card installed more stably as a Plug and Play device. Also, the included software installed without any problems or additional configurations.

The card included multiple ports on the back for sound and video input. There was a port for Coaxial RF cable, S-video connector and composite video and sound. After learning how to adjust record levels through the Microsoft Operating System's sound control panel, it was discovered that the capture card did not actually capture sound but merely relayed it through a port on the card into the system's sound card through a second cable. In order to reduce distortion, the sound feed bypassed the capture card and was connected directly to the system's sound card.

However, once the sound card received a direct connection from the sound source, the captured sound was distorted. While the sound was loud, the voice became

distorted as if the speaker was talking through a wall. After much experimentation, it was discovered that the sound signal was actually too hot. The sinusoidal wave normally associated with sound was becoming squared off due to the maximum capabilities of the digitizing system. Once the sound recording level was reduced through Microsoft's sound panel, speech quality returned to a normally acceptable range.

The software provided with the capture card offered minimum functionality but presented a few interesting concepts used to evaluate other capture applications. The software, for example, only captured uncompressed data. The file size for `.avi` and `.mov` formats was exactly the same with a 10-minute, 320x240 image occupying 1.33GB. Using the open-source C code from the GNU project that was released with the MPEG-1 [www.mpeg.org] specification, the file was compressed to 88.1 MB (15:1 compression ratio).

This method presented two main concerns. The first issue was that the size of the uncompressed intermediate files grew too rapidly. There appears to be an undocumented file size limitation on the FAT32 file system where any single file exceeding 2GB caused the file to become unreadable. In fact, a Property display of the file indicated a negative file size. This leads one to believe that a signed integer was used to index the file size and that when the file grew too large; the integer reset to its negative value and became unusable. The bug was reported to Microsoft with no reply.

The second concern was the length of time it took to convert the `.avi` or `.mov` file to the MPEG-1 specification. The compression had a 7:1 ratio of time for conversion to complete. For example, a one-hour movie consumed 1 hour for capture and another 7 hours for conversion. It is easy to conclude that this is an unacceptable process for a large number of captures. Since there was no batch or remote utility, all work had to be supervised and conducted by human operators.

2. Ravisant Technologies

Ravisant Technologies DVR software was the first commercial solution utilized to overcome the file-size limitation and time limitation of the previous attempts. Since the capture card was already installed and confirmed to work, installation of Ravisant's DVR was believed to be trivial. Unfortunately, the card did not natively support any of

Ravisant's compression codecs in the original configuration. After contacting the company by e-mail, a suitable patch was eventually discovered to allow the software to work with the video card's software drivers.

After successfully installing the software, tests were undertaken to measure file size and bandwidth concerns with the new software. Table VII-1 is based upon a 10-minute truncated video of Fred Brook's Turing Award Lecture using AverMedia's TV98 Capture Card, Ravisant DVR software on a Dell Dimension 4100 WorkStation with a 1GHZ PIII CPU and Windows 98 Operating System. The captured file sizes were created to represent the range of codecs that the software is capable of capturing.

	1 Hour	10 Minutes	1 Minute	1 Second
Web Server Emphasis				
28.8kb/s	9.00 MB	1.50 MB	150 KB	2.50 KB
28.8kb/s w/ audio emphasis	13.92 MB	2.32 MB	232 KB	3.87 KB
56kb/s	15.24 MB	2.54 MB	254 KB	4.23 KB
multibit 28.8, 56, 100kb/s	64.20 MB	10.7 MB	1.07 MB	17.83 KB
ISDN(128kb/s)	67.20 MB	11.2 MB	1.12 MB	18.67 KB
Video Emphasis				
28.8kb/s (160x120)	15.24 MB	2.54 MB	254 KB	4.23 KB
56kb/s (172x144)	23.22 MB	3.87 MB	387 KB	6.45 KB
100kb/s (320x240)	44.34 MB	7.39 MB	739 KB	12.32 KB
256kb/s (320x240)	97.80 MB	16.3 MB	1.63 MB	27.17 KB
384kb/s (320x240)	150.60 MB	25.1 MB	2.51 MB	41.83 KB
768kb/s (320x240)	303.00 MB	50.5 MB	5.05 MB	84.17 KB

Table VII-1:   ASF Video Codec File Storage Sizes

The software was also capable of capturing content in the MPEG-1 and MPEG-2 open standard formats. There is a much larger file size for MPEG-1 when compared to ASF, which is due to MPEG being an older standard that cannot achieve the highest compression ratio. MPEG-2 is the largest file size since it bases its capture quality on the size of the bitstream that it must render. MPEG-2 was designed for television broadcast and must conform to additional limitations that its web-based codecs do not have to

116

adhere. However, all media players were able to render the MPEG-1 and MPEG-2 format.

	1 Hour	10 Minutes	1 Minute	1 Second
CDx1				
Low (160x120)	550.20 MB	91.7 MB	9.17 MB	152.83 KB
Medium (352x240)	550.20 MB	91.7 MB	9.17 MB	152.83 KB
High (160x120)	550.20 MB	91.7 MB	9.17 MB	152.83 KB
High (320x240)	550.20 MB	91.7 MB	9.17 MB	152.83 KB
High (352x240)	550.20 MB	91.7 MB	9.17 MB	152.83 KB
CDx2				
Low (160x120)	1.026 GB	171 MB	17.1 MB	285.0 KB
Low (320x240)	1.026 GB	171 MB	17.1 MB	285.0 KB
Medium (352x240)	1.026 GB	171 MB	17.1 MB	285.0 KB
High (352x240)	1.026 GB	171 MB	17.1 MB	285.0 KB
CDVideo				
Low (352x240)	598.8 MB	99.8 MB	9.98 MB	166.33 KB
Medium (352x240)	598.8 MB	99.8 MB	9.98 MB	166.33 KB
High (352x240)	598.8 MB	99.8 MB	9.98 MB	166.33 KB

Table VII-2:   MPEG-1 Video Codec File Storage Sizes

By studying the file sizes in Tables VII-2 and 3, a few previously unrecognized facts were discovered. In Ravisant's MPEG-1 controls, a low, medium or high setting can be set. These settings did not adjust the quality or size of the images. Second, for both MPEG-1 and MPEG-2, changing the video dimensions did not affect the file size. For instance, if a file was captured at 160x120, 320x240, or 352x240 in MPEG-1 CDx2 format, it still grew at 285 KB per second. This can be explained by use of the multiple compression methods discussed in Chapter III, Section B of this paper. Hence, if a larger image is desired, there are no additional system-storage requirements.

	1 Hour	10 Minutes	1 Minute	1 Second
1500kb/sec (160x120)	786.0 MB	131 MB	13.1 MB	218.33 KB
1500kb/sec (320x240)	786.0 MB	131 MB	13.1 MB	218.33 KB
3000kb/sec (320x240)	1.500 GB	250 MB	25.0 MB	416.67 KB
4000kb/sec (320x240)	1.974 GB	329 MB	32.9 MB	548.33 KB

Table VII-3:   MPEG-2 Video Codec File Storage Sizes

3.   Media Encoder

The Microsoft Media Encoder is a free application offered by Microsoft for the encoding and capturing of multimedia content on Windows Machines. The current release is version 7. The encoder only encodes to Microsoft-proprietary formats that can only be served and rendered with Microsoft products. There are currently no products available that can take a proprietary format of Microsoft's and convert it to an open standard codec. This restriction has many pros and cons that are evaluated next.

First, if an organization strictly uses Microsoft software, there are minimal configurations issues involved in capturing, streaming and rendering video for internal use. For instance, utilizing Microsoft's Media Encoder installed on a computer with a capture card and attached to a video camera, a broadcast session was created through the use of a wizard in less than one minute. Other computers in the lab and on campus were able to connect to the session simply by using the Uniform Resource Locator (URL) of mms://computer. The word "computer" is substituted with either the Internet Protocol (IP) Address or computer name is name servers and protocols were enabled. Hence, if a computer broadcasting the session had an IP address of 131.129.7.3, the clients open Internet Explorer and type mms://131.129.7.3. The client machine then automatically launches Microsoft's Media Player and begins to render the live stream. Since the lab where this experiment was performed was equipped with strictly Microsoft Operating Systems and application software, the time involved was minimal. In less than five minutes, a session was established and over thirty clients were connected to the session viewing the same images and listening to the same audio. Unfortunately, this approach does not support arbitrarily different clients connected via the Internet.

118

For Video-On-Demand (VOD), a server must be utilized to stream the content. A File Transport Protocol (FTP) server allows simple file to transfer but the user then has to wait for the entire contents to be downloaded. Another alternative is to place the captured files on a Microsoft server and use the mms:// protocol, where upon the file will begin to stream from the server as soon as a connection is made. Finally, a new project in the streaming server market is the Jakarta Project for the Apache Server [jakarta.apache.org]. By installing the Tom Cat application, streaming media can be served on an Apache Server.

It is now a matter of matching the correct encoding to the bandwidth available. Table VII-4 shows a few of the typical streaming solutions with their related file sizes. It also worth noting that the 28.8 and 56 Kb/sec quality streams do not achieve the full thirty frames per second and stereo 44.1Khz audio sampling.

	1 Hour	10 Minutes	1 Minute	1 Second
28.8 Kb/sec	9.252 MB	1.54 MB	154.2 KB	2.57 KB
56 Kb/sec	13.46 MB	2.24 MB	224.4 KB	3.74 KB
64 Kb/sec ISDN	22.50 MB	3.75 MB	375.0 KB	6.25 KB
128 Kb/sec Dual ISDN	43.00 MB	7.50 MB	750.0 KB	12.5 KB
256 Kb/sec Broadband	102.0 MB	16.7 MB	1.670 MB	27.8 KB
384 Kb/sec Broadband	145.0 MB	24.2 MB	2.418 MB	40.3 KB
768 Kb/sec Broadband	290.0 MB	48.4 MB	4.836 MB	80.6 KB

Table VII-4:   Media Encoder Video .wmv File Storage Sizes

If an organization or website supports open standards in addition to other products besides Microsoft, there are further acceptable solutions that allows them to render the streaming multimedia. This capability allows organizations relying on Unix, Linux, Beos or Apple systems to utilize the multiple compression schemes and diverse quality of these codecs. Hence, an organization now has many possible solutions for its multimedia needs.

4.   JMStudio

JMStudio is an excellent example of Java Media Framework's capture capability. It became a test-bench program to shorten development time using JMF for a custom

capture application. If there was a problem with the JMF capture application, one of the first troubleshooting methods was to test JMStudio in a similar situation on the same machine with the same hardware. The beta capture stations were tested using JMStudio to prove that capture was possible with the hardware combinations created for the capture stations. The tests were restricted to machines running Microsoft Operating systems since the capture cards purchased for the tests only came with Microsoft capture drivers. The three operating systems utilized were Windows 2000 Server, Windows 2000 Professional and Windows 98. Machines were Dell Dimension 4100 Workstations, piecewise assembled multiprocessor Pentium-based computers, and Silicon Graphics Visual 320 PCs.

One capture device was tested on the Windows 2000 Server. A D-link USB web camera was used as the video source while a microphone connected to a Creative Labs Sound Blaster Live! X Gamer card was used as the audio source. The computer was a Dual 500MHZ Celeron Processor hand assembled computer using an Ultra 66 hard drive. JMStudio was able to capture, store and transmit the video camera's images and sound without any change in the default settings. The images can be captured in dimensions of 80x60, 160x120 or 320x240. The frame rate for capture can be specified up to thirty frames per second. In actuality, the camera was only able to generate five frames per second within these constraints. The sound, however, was captured without any delay or pauses.

Two capture devices were tested on Windows 2000 Professional installed on two Dell Dimension 4100 Workstations. The capture devices were the Winnov Videum 1000 and USB Belkin F5U208. The only modification necessary for the Winnov was adjusting the capture source by pausing the player and adjusting properties under the player menu item. Exact procedures are specified in Appendix A. Also, there was a need to set the proper sound drivers and sound sources under Microsoft's volume properties since the Winnov capture card had its own set of sound capture and speaker ports. When the settings are correct, the Winnov 1000 can capture a variety of dimensions and frame rates up to 640x480 at thirty frames per second.

The Belkin cord required no modifications to the default settings but only allowed a 320x240 video capture. The frame rate was also limited to approximately five frames per second. Even when a higher capture rate was specified in JMStudio, the camera still only produced five frames per second. As new drivers are released for this product, its compatibility and usefulness may increase.

On Windows 98, the AverMedia TV98 capture card worked with no modifications necessary. The AverMedia Card was the lowest-priced card (with a price below fifty dollars) but had robust capabilities. It was able to capture a variety of formats with dimensions from 80x60 to 320x240. Frame rates can be captured from one up to thirty frames per second. The sound was captured using the system's built-in Sound Blaster audio card.

JMStudio was originally tested on seven SGI Visual 320 PCs running Windows 2000 Professional. The video was captured with no difficulties but error messages indicated that the sound drivers could not be initialized. Both DirectSound and JavaSound capture was attempted but met with failure. The built-in sound-capture chip did not have software drivers compatible with JMStudio. A solution is to purchase an add-on PCI sound card such as a Creative Labs Sound Blaster and attempted capture with the new device. Future updates to SGI drivers might fix the problem. The bug was reported to SGI and JMF but no solution was reported in return.

Also, there were differences notices when related to capture devices and the choice of JavaSound or DirectSound capture device. The USB device was having a stop and go (i.e.halting) capture of sound using DirectSound but captured in a continuous manner for JavaSound. The PCI cards, however, performed satisfactorily no matter which device was used for capture.

	1 Hour	10 Minutes	1 Minute	1 Second
H.263 GSM (176x144)	102.0 MB	17.0 MB	1.70 MB	28.34 KB
H.263 IMA4 (176x144)	234.0 MB	39.0 MB	3.9 MB	65.0 KB
RGB IMA4 (160x120)	2.40 GB	400 MB	40.0 MB	666.67 KB
RGB IMA4 (320x240)	7.98 GB	1.33 GB	133 MB	2.216 MB

Table VII-5: JMStudio Video Codec File Sizes

5. Apple's QuickTime Pro

Although it was not used for capturing, the QuickTime Pro application was used for transcoding. It allows a large format file such as RGB IMA4 (320x240) to be converted to an acceptable streaming format that was optimized for an Internet connection. Table VII-6 summarizes the file size reduction of Dr. Fred Brooks' Lecture. It is also worth noting that Apple's QuickTime Pro was not able to correctly transcode an MPEG video since the resulting file did not contain an audio track.

	1 Hour	10 Minutes	1 Minute	1 Second
20Kb/sec	16.02 MB	2.67 MB	267 KB	4.45 KB
40Kb/sec	32.49 MB	5.49 MB	549 KB	9.15 KB
100Kb/sec	65.6 MB	10.8 MB	1.08 MB	17.93 KB

Table VII-6:   Apple QuickTime Pro Video Codec File Sizes

C.   EXEMPLAR CONTENT

The fundamental goal of this case study was to create an example of a finished lecture captured using low-cost equipment and storable on a single compact disk. The tradeoffs in video quality, size and ability to be rendered were all factors in the final choices of video formats. The CD's storage limitation of 650MB placed a final throttle on content size and diversity. Since the time of the original capture experiments, 700 MB CDs have become commonplace and might therefore enable alternate formats to be stored on the disk. Also, DVD-R has entered the commercial markets and is experiencing rapid cost reductions similar to the CD-R a year earlier. If the trend continues, the current price of a DVD-R recorder may drop to less than $100 and the price of disks may become less than a third of a dollar. Once this commercial phenomenon occurs, CD-Rs will disappear and consumers will use the larger format disks. The current commercial DVD-R disks can store 4.7 GB of data but cost

approximately fifteen to thirty dollars. For this case study, therefore, an upper limitation of 650 MB was imposed.

1.   Video

The video capture process is designed to achieve the best quality in the smallest amount of time. The main consumer of time was the digitalization of the SVHS tape. Every hour captured required a human operator to be available at the beginning and end of the capture process. Also, if there is switching between sources, the human operator needs to view the entire session and switch the sources at appropriate times. The digitizing of the content is where the most operator intervention is incorporated. Instead of having a digitalization session for each format desired, an intelligent solution is to digitize the multimedia in the highest quality desired, and then down sample the digitized videos into alternate multiple desired formats. Down sampling is an automated task and is also well suited for hardware-based server solutions.

With this approach in mind, the Brook's Turing Lecture was captured into an MPEG-1 file. MPEG-1 was selected since it created a high-resolution representation of the video at 30 frames per second and 320x240 dimensions. Additionally, the format only occupies two-thirds of the CD, which allows for additional formats and web pages to increase the usability of the content. MPEG-2 format might have been selected but even the lowest quality generated a file larger than the 700MB CDs can store. Although MPEG-1 showed more artifacts and less fidelity than MPEG-2, MPEG-1 was the correct choice for the CD size limitation.

JMStudio might have conceivably been used to create the initial capture, but the overhead of interpreted Java reduced the video quality well below that of MPEG-1. For instance, the JMF capture process, using the same hardware, could not capture more than twenty frames per second for a RGB format. Also, the RGB format had to be reduced to a dimension of 160x120 and was restricted to less than fifty minutes so as not to exceed the 2 GB file size limitation. Alternatively, capturing directly to the H.263 format in JMStudio did not have the file size limitation due to compression but displayed a large number of artifacts for a sub-standard picture. Twenty frames per second was also the highest achievable frame rate for this format as well using JMStudio. The format also

only supported a 177x144 image dimension. Hence, JMStudio was not an ideal first digitalization program.

However, once the lecture is available in a digitized format that can be manipulated by the computer, it is easily transcoded with a number of applications. Microsoft's Media Encoder was able to take the video and convert it into all formats up to 768 Kbps optimized streaming media. Apple was unable to correctly convert the MPEG-1 but was able to transcode it once the source was converted to a `.mov` format. JMStudio was the main application for transcoding the MPEG-1 into a usable format for Quicktime. The final process consisted of capturing the initial video in MPEG-1 with Ravisant's DVR, transferring the file to another computer which has JMStudio installed, converting the file to H.263 format, running Apple's QuickTime Pro to convert the file to multiple streaming formats and then place all of the files within a single directory. The same process can be used for `.wmv` formats but Microsoft's Encoder can be used as soon as the MPEG-1 file is created.
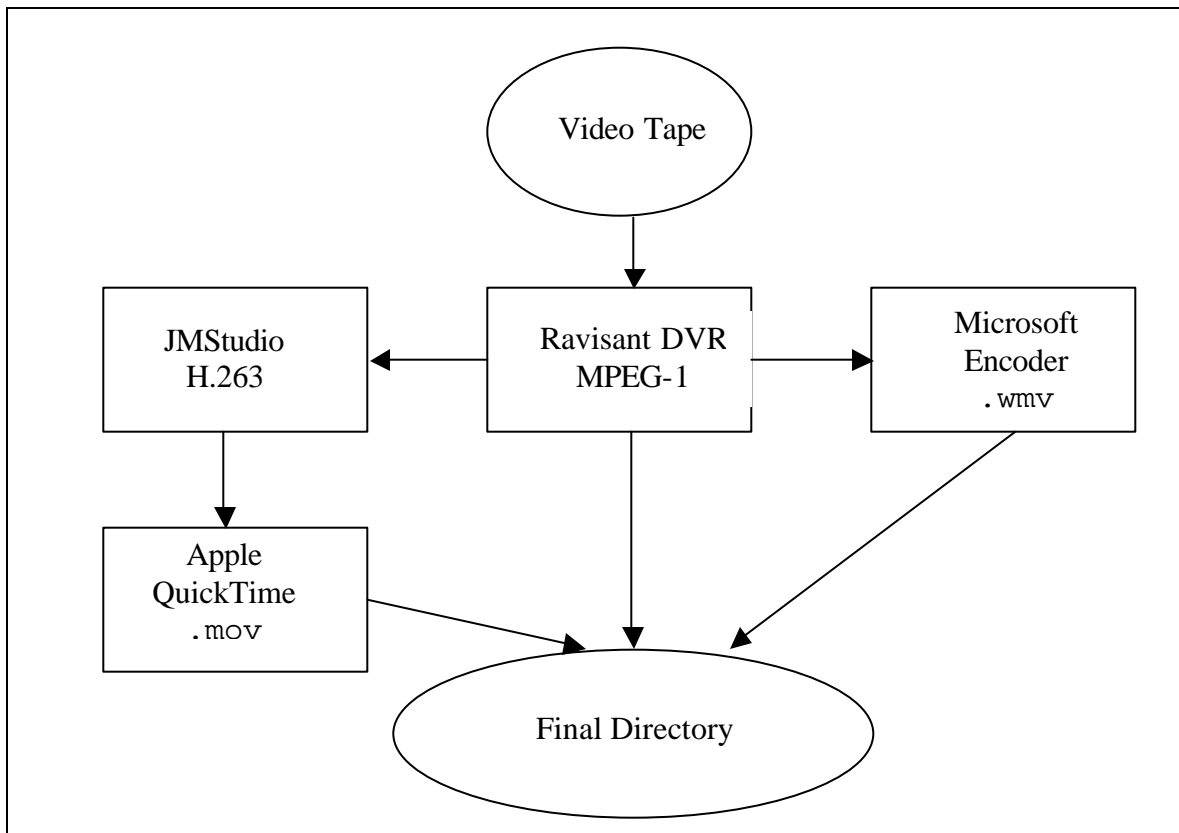
Figure VII-1: Best File Transcoding Process

Once all of the desired formats are created and placed into a single directory, additional information is placed into the directory as well. The slides presented during the lecture were converted to GIF images. For this lecture, the slides were presented in PowerPoint, and so PowerPoint was used to convert the slides into web pages. It was not required to create an index or additional web pages to wrap the slides since the only desired product is the slides as GIF images optimized for web pages. There are many programs freely available that can convert the images to GIFs. Once converted, the autogenerated web pages (described in the next section) are customized for the presentation. Thus the same solution works both for web delivery, file copying and CD redistributions. The directory is now copied to a CD-R and distributed.

2.   Siggraph Online Autogenerated Web Pages

While the video displays the sights and sounds of the presentation, a large portion of the information presented is not readily available to the viewer. For instance, slides presented during the lecture may not be optimized for the small display of the captured format. Biographic and contact information may never be presented during the lecture. Also, legal copy writes and permission for use may never be incorporated into the presentation. These limitations may be overcome with additional information provided by web pages, which present the content.

For this CD, Juan Gril, a volunteer for the Siggraph 2001 Online Committee, created web pages that match the style and functionality desired by the Siggraph Committee for the presentation. The final look and design was called BroadCaster. It used a combination of HTML and JavaScript to allow users to gain a general overview of the lecture, select video streaming size, download additional information such as biography and permission for use, and scroll through the presentation slides. The JavaScript also advanced the individual slides based upon predefined times. The times were recorded by viewing the lecture and then writing down when the times when the slides were changed. A better solution is to have a program that allows the user to press a button every time a slide changes and then have the program write down the time. This process is still prone to error.
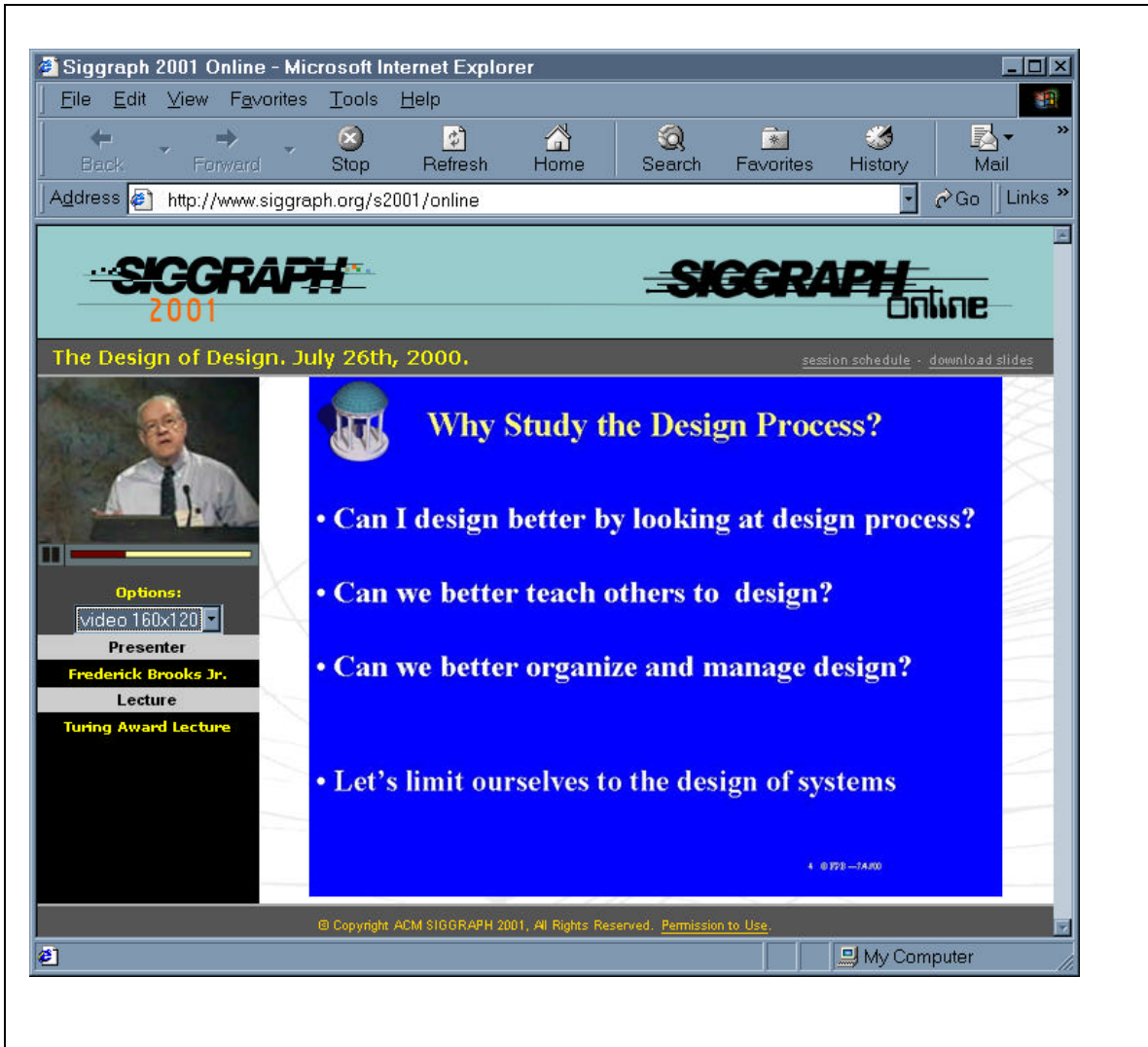
Figure VII-2: Turing Award Presentation.

While BroadCaster provides a solid template for the creation of additional presentation CDs, it is still extremely user intensive to customize. Each lecture will contain unique information such as the timing of slides, name of presentation, name of the lecturer and more. With this limitation in mind, it provides a comprehensive wrapper for the delivery of any presentation.

3. Final CD

Since there were multiple requests for the lecture with QuickTime formats, Table VII-7 displays the contents of the final CD that was created for this case study. The inclusion of the QuickTime formats can easily be replaced with Microsoft's WMV files. The MPEG-1 file is not suitable for streaming across the Internet but is appropriate for a LAN delivery or viewed directly from the CD. This format allows for the maximum amount of information to be delivered to the user in the highest fidelity.

The BroadCaster HTML pages were used to present the lecture and allow optimized user interface include the original slides presented by Dr. Brooks during the lecture. The slides were converted using PowerPoint to an HTML format, which is then advanced using JavaScript to maintain timing with the presentation. The final CD can be copied to a web server for delivery across an intranet and the Internet.

	Item	Image Dimensions	Frames/Sec	File Size
Video				
	MPEG-1	320x240	30	458 MB
	H.263/GSM MOV	177x144	30	85 MB
	QuickTime 100Kb/s	177x144	20	54 MB
	QuickTime 40Kb/s	177x144	10	40 MB
BroadCaster with Slides				1 MB
			Total:	648 MB

Table VII-7　　Dr. Fred Brooks Compact Disk Content

D. SUMMARY

The Dr. Fred Brooks case study demonstrated the ability to use minimal resources to capture and distribute a multimedia lecture. Until recently, the final CD created for this study needed specialized hardware and cost thousands of dollars. With the current speed of today's personal computers and the availability of multimedia capture devices, the capability to distribute multimedia content in a multitude of formats is readily

accomplished. By choosing formats and evaluating their trade offs, optimized content distribution is now possible. The lecture is now packaged in a media that will allow it to be easily distributed and benefit future generations with Dr. Brooks' presentation of <u>The Design of Design</u>.

VIII. DR. RICHARD HAMMING CLASS CASE STUDY

A.   INTRODUCTION

The Dr. Richard Hamming Class Case Study makes the assumption that an organization may have already videotaped or digitized content considered optimized for its own purposes. In the case of the Hamming Lectures, the presentations are from the man who created the concepts named after himself. It stands to reason that the world's greatest expert on Hamming Parity Code is Richard Hamming. By already having his lectures videotaped, the only challenge remaining is to repackage the content in a Web-enabled format that can be disseminated in a distance-learning setting. Hence, this section concentrates on the creation of tools using JMF. The tools are able to digitize and repackage the original media into an optimized format for today's technology. Multiple applications were created for this endeavor with both success and failure. The results are well documented for future work.

B.   FORTE FOR JAVA

The goal of the Hamming Class case study was to use the Java Media Framework API to create a program capable of capturing the previously recorded lecture series delivered by Dr. Richard Hamming, so that the class can be offered in a Distance Learning Environment. This study built upon previous work explored in the Dr. Fred Brooks and the Naval Postgraduate VTC case studies to optimize the content for a Distance Learning Environment.

Since the previous studies concentrated on the presentation of material and the proper employment of video formats, this study focused on the development of software that can take the place of the commercial software previously used. A series of programs were created using the free Forte for Java CE Integrated Development Environment (IDE) that can be downloaded from http://www.sun.com/forte/ffj/. Figure VIII-1 shows the IDE with a standard number of windows displayed. There are multiple windows that can be opened so that the environment is configured for the programmer's optimized use.

The main strength of Forte is its rapid graphical user interface (GUI) integrated development environment (IDE). By choosing the *absolute layout* mode from the layout manager, components such as buttons and panels can be visually positioned within the form. Other layouts, such as *grid* or *grid-bag* layouts, allow components to be snapped to a grid and automatically resized for display environment. The ability to visually position Java Swing and Abstract Window Tools (AWT) dramatically reduces development time.

Additionally, development time is reduced with built in automation. By clicking on a button in the graphical editor, Java methods are automatically created that will listen to events generated by a user. Code can be placed within these functions to make an application event driven. For instance, no action occurs until the user clicks on a button. Once the button is clicked, the code is executed. The IDE allows for the customization of which events the program will respond to. Some events are *Mouse Click*, *Mouse Over* and *Status Changed*. Hence, the development of a program-shell is extremely rapid using Forte.

Two tools included in Forte that aid in the packaging of the application are the *jar packagizer* and *javadoc* creation tools. The jar packagizer can take multiple Java classes, compile them and place them into a single executable file. By including "Main-Class: filename" in the program's manifest, the application will execute when a user clicks on the final jar file. The javadoc creation tool is designed for the programmer community where documentation needs to be standardized and proliferated. The javadoc allows future programmers to reuse the code.

However, the main weakness of Forte is that it "write-protects" much of the code that it creates. Because of "write-protection", it is not easy or intuitive to change constructor code or modify default component initialization. In order to over come these restrictions, the *code generation* tab under the *component inspector* properties can be toggled to change the code. Unfortunately, this text area now available to change the code does not have an automatic completion ability or program checking capabilities. After typing in the code, the programmer must exit out of two windows before checking if the code compiles. If the code does not work, the programmer must navigate back

through the previous tabs and change the code. The process repeats until the modified code can compile.

The normal editing area allows certain portions of the program to be modified. This window will automatically complete any code typed as long as a new library is registered in the program's database. In order to register a library, the jar file must be imported into the project. The file is then "right clicked" using a mouse to reveal tools that expose the option to update the parser database. Once updated, the IDE can now show methods available for variables and classes.
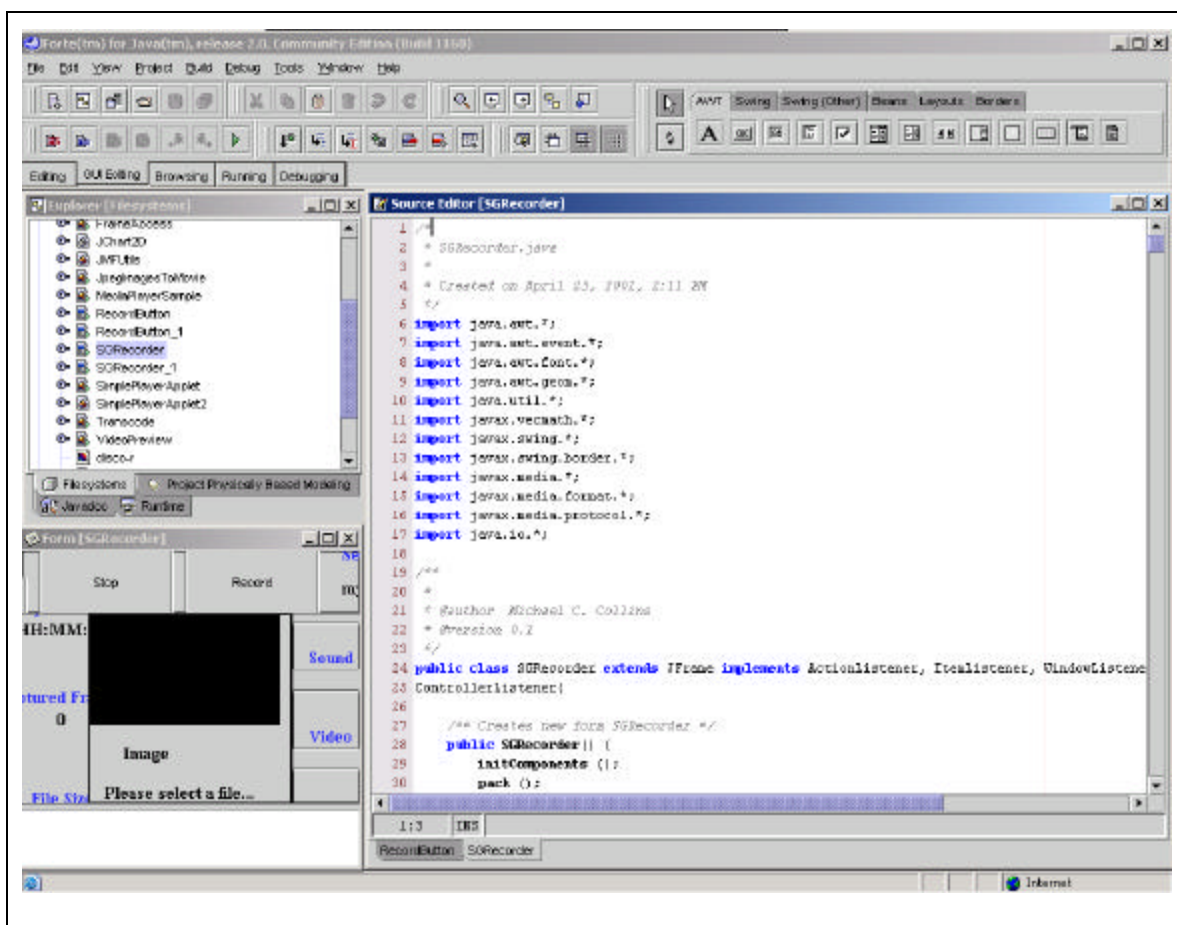
Figure VIII-1: Forte For Java Integrated Development Environment.

Due to the restrictive development environment, once the GUI was created in Forte, the java file was saved and then opened in Borland's JBuilder. Enterprise Version 4 and Personal Version 5 were used throughout this development process. JBuilder

131

allowed full control over the code and automatically added classes and class paths through the auto-complete function. With the knowledge that everything was added to the database, human error did not interfere with optimized use of the IDE. Also, earlier versions of JBuilder impeded the programmer from finding a pre-created file not in the current project. The newer versions allows the user to browse the computer's file directory with ease.

Hence, by using the best of both IDEs, applications were rapidly created with Swing interfaces and JMF functionality. A few tricks were required to overcome concurrent use of heavyweight components, such as Swing and JMF, to work within the same program. However, the SGRecorder example shows how the heavyweight components can eventually saturate a computer to the point of failure. When the programs were complete, custom applications that can capture, play and transcode video were developed to capitalize upon the work of the previous case studies. In the end, the previously recorded lectures were digitized in a simulated production environment where minimizing human labor was a goal.

C. CUSTOM APPLICATIONS

Several custom applications were created for this study. They can be categorized under TestButton, SGRecorder, RecordButton and SGTranscoder. There were actually dozens of programs with multiple variations used to explore the capabilities of Swing and JMF. These four programs cover the range of programming explored.

These four programs represent significant milestones or learning points used in the final arena of digitizing the lecture series. TestButton is unique in that there were many programs called TestButton used to explore specific Swing functions. SGRecorder was a single program that resulted from combining multiple JMF and Swing methods until the program eventually failed do to overuse of computer resources. RecordButton was the result of scaling back SGRecorder into an optimal, minimally configurable capture and play program. SGTranscoder was developed to overcome the lack of batch transcoding software.

1.   TestButton

TestButton was based upon the concept of having a jFrame containing a single button testing a specific characteristic of the programming language. Some representational code is a test of launching a separate executable application from within a program, moving data into and out of ListBoxes and browsing the computer's file directory. All of these functions and more were eventually used in the more complex programs employing JMF.

a.   *Application Launch*

The launching of an executable from within a java application served a two-fold purpose. First, since JMStudio was used as test benches in several situations, the ability to launch the program from the application at any time saved numerous hours of searching for the program. Also, if the custom application did not work on new hardware and needed a more complex or robust program, someone trained to use JMStudio can continue the capture process. If there was time, this robustness could be built into the custom application as an expert mode, but it was not necessary for the final product.

Figure VIII-2 shows how simple it is to launch an executable from a Java application. The method was created for a jButton's event of ActionPerformed. When the program user clicks on the button and the event signals the jButton1ActionPerformed method. The name was automatically generated using Forte. The code simply comprises of a string pointing to the filename with its directory structure location. Then, within a try-catch routine, the program is executed. If the program does not exist or cannot be executed for any reason, the catch statement notifies the user.

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
     //begin jButton1ActionPerformed

     String program = "C:/Program Files/JMF2.1.1_beta3/bin/jmstudio.exe";

     try{

          Runtime.getRuntime().exec(program);

     }

     catch (Throwable t) {

          System.out.println("Unable to Load and Execute Program " + program);

     }

  }//end jButton1ActionPerformed
```

Figure VIII-2: Launching a Program From Within A Java Application.

b.   ListBox

The use of listBoxes allows the program to provide sets of information to the user. By filling listBoxes with relevant information, the user can quickly select revelent settings to customize the program's usage. A quick method of filling a listBox is to use a Vector. By importing the `util` library, a program can use a Vector to store dissimilar types of components. By loading a Vector with strings and adding the Vector to the listBox, the listBox is immediately filled with useful information. Using `listBox1.setText(Vect)` accomplishs this. The Vector is filled with `Vect.add("a", "b", "c")` method. All Vector methods can be used to modify the text such as selecting a string by index number, jumping to the first string or jumping to the last string. Vector information is also available such as querying the number of components in the Vector.

Naturally, listBox has its own methods of adding and removing individual fields. The add and remove commands will manipulate the fields. Also, the updateUI method should be employed to insure the GUI display reflects the latest information. The strength of returning a field selected by the user and then removing it, is that the user now has the ability to rapidly remove information. When the user clicks on a field in the listBox, its index number is returned from the getSelectedIndex method. The text located in this index is now removed using the remove function. Hence, data can be rapidly added and selectively removed from a listBox.

```
int index = jList7.getSelectedIndex();
jList7.remove(index);
jList7.remove(jList7.getSelectedIndex());
jList7.updateUI();
```

Figure VIII-3: Removing An Item From A ListBox

c.   *Directory Browsing*

Browsing the file system and capturing the name of a file from the directory saves a tremendous amount of time for the program's user. First, by allowing the user to browse for a file, typographical mistakes can be avoided. Hours of typing and re-typing file names and locations are easily dismissed. Second, by setting up a file structure and naming scheme before a large scale capture session, allows the user to simply browse to the correct file name and begin the capturing process. Hence, the user can operate the program with minimal training.

In Figure VIII-4, the name and directory structure is placed within a string called filename. JList7 has a `setText(Vect)` method so that the filename is added in the first position of the Vector and displayed in the listBox. The `setSelectedIndex` moves the focus to the newly added filename. Finally, if there is an error, the error message is placed in a separate listBox using the same Vector technique.

135

```
try {
    FileDialog fd = new FileDialog(this, "Select File",
            FileDialog.LOAD);
    fd.show();

    String filename = fd.getDirectory() + fd.getFile();


    Vect.add(0,filename);
    jList7.updateUI();
    jList7.setSelectedIndex(0);

}
catch (Exception e) {
    Vect2.add(e.toString());
    jList1.updateUI();
}
```

Figure VIII-4: Browsing Directory to Select a Filename.

2.   SGRecorder

For the purposes of SGRecorder, a minimally configurable Processor Model was employed. The model is used to create a Processor that is capable of capturing live audio and video data, encode the data in formats such as IMA4 and RGB tracks, multiplex the tracks and save the combined media stream to a QuickTime file. By using the Processor Model and specifying a valid track format, the Processor is automatically connected to a capture device which can support the formats. If there is no device to support the formats, the processor cannot be created and error events are generated.

While there are different methods of creating a Realized Processor, the method used in this example uses a "best attempt" method. This means the computer will connect to any device supporting a format similar to the one specified. Hence, the less information provided about the desired format, the better chance the computer has of selecting a device that will work.

Additionally, the application used a large number of Swing components to provide the user with the maximum amount of information. Buttons as well as current date and time were placed across the top. The buttons allowed the user to quickly activate the program. Information about the captured program is placed down the left column. Capture setting modifications are placed down the right column. In the middle

136

was a panel to allow videos to be seen as well as controls to adjust sound and display properties. There is a listBox that contains the name of the file to be captured or rendered. Finally, across the bottom, is a status window containing a running log of program events and errors.
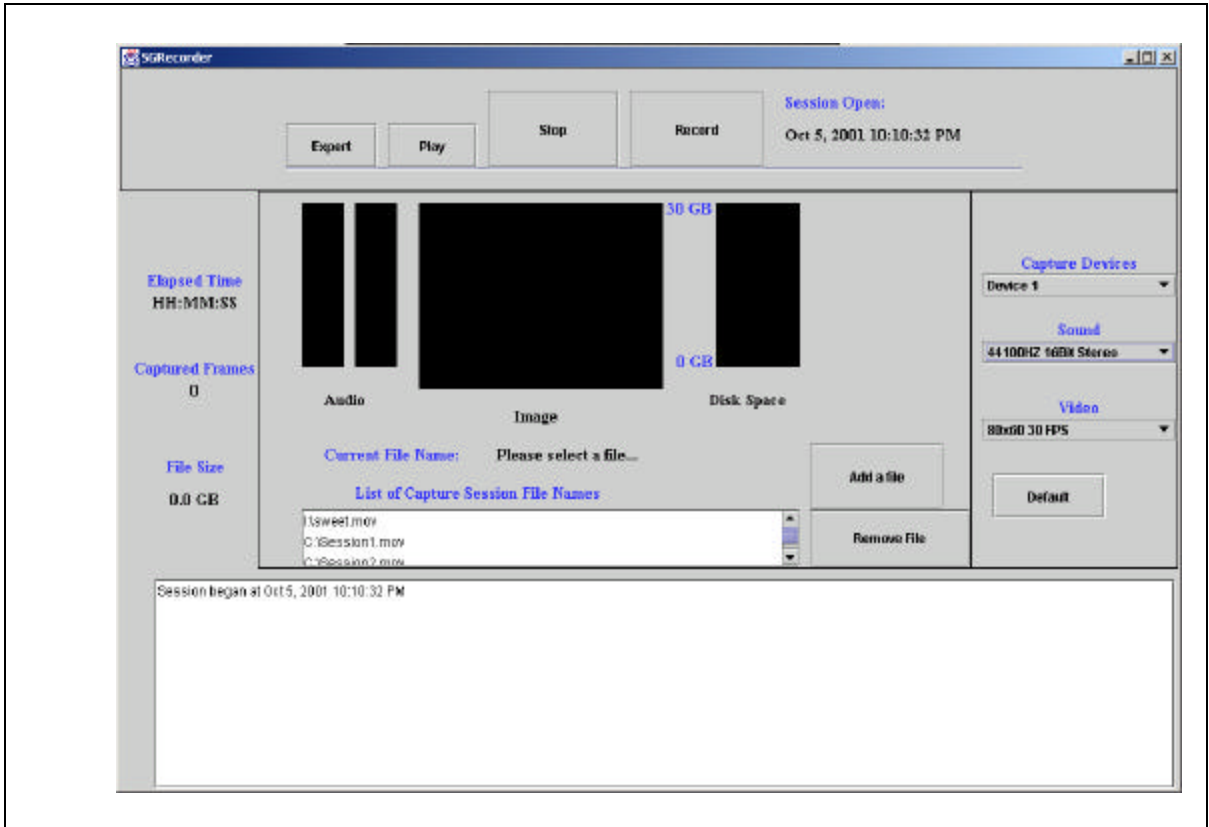
Figure VIII-5: SGRecorder.

Since the program overloaded a Pentium III 1 GHz computer, it was not a viable tool for capturing video. However, the code which was generated for the capture and play process provided a solid base for future programs. Additionally, since there is multiple ways of specifying a track format as well as creating a Processor, this program used the simplest methods available. With these techniques exhaustively explored, future programs contained the more complicated methods of specifying these objects. Hence, the future programs were able to be more configurable and robust.

a.   *Libraries*

The libraries needed for this application can be divided into three groups. The first group allows for the abstract window tools to be utilized for listening to program generated events. The second group allows for the use of Swing components. The reason swing components are desirable is that they are optimized in several ways. For instance, images drawn to a panel, a close equivalent to AWT's canvas, are automatically drawn using a double buffer. The performance gain and rendering improvement is noticeable. Finally, the last group of libraries allows for the capture, rendering and saving of multimedia streams. The jmf.jar file must be included in project-required libraries or the program will not compile.

```
import java.awt.*;
import java.awt.event.*;
import java.awt.font.*;
import java.awt.geom.*;
import java.util.*;
import javax.swing.*;
import javax.swing.border.*;
import javax.media.*;
import javax.media.format.*;
import javax.media.protocol.*;
import java.io.*;
```

Figure VIII-6: SGRecorder Import Libraries.

b.   *Variables*

The program contained a large number of variables. However, Figure VIII-7 contains several variables of interest. First, Vectors are used extensively as the main structure for holding multiple components. In order to discover the capture devices supporting particular formats, the devices need to be registered and stored in a Vector. If the programmer wants to find a device that supports `VideoFormat.RGB`, the CaptureDeviceManager can be queried and have the results placed inside a Vector.

Second, multiple utility classes were employed to give the program added functionality. Unfortunately, these classed only exaggerated the exhaustion of resources caused by SGRecorder. Most of these classed did not add to the overall usability of the

program or provide valuable information. For instance, while the Clock2 class was useful and ran in its own thread, the added information of time slowed the program to a near frozen state. For the resources it utilized, it should have been immediately removed from the program.

Finally, the program used a Player to render the videos and a Processor to capture the multimedia. A Processor can render the images but the program attempted to use a maximum number of JMF components. Components were also used to add visual components to panels. While this method worked, a better method of using MonitorControls was utilized in the RecordButton program. The Processor's captured data was fed into a dataSink that was a file.

```
        private Vector  vectorDevices = null;
        private Vector  vectorAudioDevices = null;
        private Vector  vectorVideoDevices = null;
        private Format  arrFormats [];
        private Component comp;
        private Vector Vect2 = new Vector();
        private Vector Vect = new Vector();
        private Clock2 myClock = new Clock2();
        private Player dualPlayer;
        Vector v = CaptureDeviceManager.getDeviceList(new
                    VideoFormat(VideoFormat.RGB));

        Processor p = null;
        DataSink filewriter = null;
```

Figure VIII-7: SGRecorder Variables.

c.   *Processor*

A quick method for creating a Processor is to specify the formats desired and place them within an array of formats. A format can be specified with only an encoding type such as `AudioFormat.IMA4` or can be as complicated as specifying number of channels, signed, big or little endian, sampling rate and more. The only caution with specifying a more precise format is that the capture device may not support the more exact specification. For instance, if thirty frames per second are specified and the device can only generate twenty, a failed to create processor error shall result. One method to avoid this trap is to only specify the encoder type. A more configurable

139

method is to use all specifications in the format but only specify the characteristics required. This can be accomplished with the AudioFormat or

`VideoFormat.NOT_SPECIFIED` constant. With this constant, the computer has the freedom to choose the first format matched.

Once the formats are created, the createRealizedProcessor method can be utilized by using the formats and a FileTypeDescriptor. The FileTypeDescriptor signals the Processor the type of content created. In this program, the QUICKTIME descriptor was used since the processor will be sent to a MOV file structure. Now, within a try-catch routine, the `Manager.createRealizedProcessor` method will go through the states of Unrealized, Realizing to Realized.   The Processor will attempt to connect to the first capture devices supporting the requested formats. If the Processor is in any of the states before Realized, it will finish the state and immediately move into the Realized state. The Processor has all of the information needed to connect to the data sources, but it does not yet have the ability to place the data anywhere.

```
 Dimension d = new Dimension();

d.setSize(160,120);
 Format formats[] = new Format[2];
 formats[0] = new AudioFormat(AudioFormat.IMA4);
 formats[1] = new VideoFormat(VideoFormat.RGB, d,
             VideoFormat.NOT_SPECIFIED,
             VideoFormat.byteArray,15.0f);
FileTypeDescriptor outputType =
              new FileTypeDescriptor(FileTypeDescriptor.QUICKTIME);

 try {

   p = Manager.createRealizedProcessor(new ProcessorModel(formats,
             outputType));

 } catch (IOException e) {
     System.exit(-1);
 } catch (NoProcessorException e) {
     System.exit(-1);
 } catch (CannotRealizeException e) {
     System.exit(-1);

 }
```

Figure VIII-8: Creating a Processor.

140

d. *DataSink*

The dataSink needs to be created to allow the storage of multimedia streams. The `java.io.*` library needs to be included to allow access to Java's file writing capability. The dataSink consists of a source and destination. In the case of this program, the source is the output of the Processor just created. The destination is a MediaLocator pointing to a file. The dataSink named filewriter becomes the one created by the Manager. The dataSink is then opened to insure it is ready to receive data.

```java
DataSource source = p.getDataOutput();

// create a File protocol MediaLocator with the location
// of the file to which bits are to be written

MediaLocator dest = new
    MediaLocator((String)jList7.getSelectedValue());

// create a datasink to do the file writing & open the
// sink

try {

    filewriter = Manager.createDataSink(source, dest);

    filewriter.open();

} catch (NoDataSinkException e) {
    System.exit(-1);
} catch (IOException e) {
    System.exit(-1);
} catch (SecurityException e) {
    System.exit(-1);
}
```

Figure VIII-9: Creating a DataSink.

With all of the components in place, the dataSink and Processor are started. Since the Processor was left in the Realized state, it may take a few seconds for the processor and dataSink to begin working. Once they do start, multimedia streams flow from the capture devices to the Processor in the pre-selected format. The stream then flows from the Processor to the dataSink and is stored on the disk. The raw data is

stored in a file ending with "nonstreamable" extension. Once the file is stopped, the data

is transferred from the nonstreamable version to the QuickTime version.

```
       // now start the filewriter and processor
           try {

               filewriter.start();

           } catch (IOException e) {

               System.exit(-1);
           }

           p.start();
```

Figure VIII-10:   Start DataSink and Processor.

e.   *Player*

The Player was used to render the visual and audio components of the

video file. The Player is created from a previously recorded file represented with a

MediaLocator. The addControllerListener enables the program to listen to events

generated by the Player and act upon them in the controllerUpdate method. The Player is

started and it multimedia streams are now accessible.

```
       try {

               String filename = jLabel24.getText();
               dualPlayer = Manager.createPlayer(new MediaLocator("file:///" +
                       filename));

               dualPlayer.addControllerListener(this);

               dualPlayer.start();
           }
           catch (Exception e) {
               System.out.println(e.toString());
           }
```

Figure VIII-11:   Create and Start Player.

One of the simplest methods to access the multimedia stream is to listen for events generated from the Player. Using a series of if-then statements, multiple actions can be evaluated and acted upon. For this program, when the Player generates RealizedComplete, the method generates a component equal to the Player's Visual Component and Control Panel Component. These components are added to a Swing panel for display. The Control Panel Component allows the program user to control the playing of the video.

```
public synchronized void controllerUpdate(ControllerEvent event) {
        System.out.println(event.toString());

        if (event instanceof RealizeCompleteEvent) {

            if ((comp = dualPlayer.getVisualComponent()) != null)

                jPanel9.add(comp);


            if ((comp = dualPlayer.getControlPanelComponent()) != null)

                jPanel9.add(comp);
            validate();
        }
    }
```

Figure VIII-12:   Displaying Player's Visual Components.

f.   *Freeing Resources*

When the user is done capturing or rendering the resources must be freed. Under the stop button located in SGRecorder, all resources are released. A simple test determines if the Player was rendering. If it was in use, it is stopped and its resources are deallocated. Additionally, the Visual Component and Control Panel Component are removed from the display panel. However, if the player was not in use the processor and filewriter are stopped and closed. Resources have been released and the program is able to perform another function.

```
            if (dualPlayer!=null) {

                dualPlayer.stop();
                dualPlayer.deallocate();
                jPanel9.remove(dualPlayer.getVisualComponent());
                jPanel9.remove(dualPlayer.getControlPanelComponent());

            }
            p.stop();
            p.close();

            try{
                filewriter.stop();
                filewriter.close();

            }
            catch (Exception e) {
                System.out.println(e.toString());
            }
```

Figure VIII-13:   Stopping FileWriter, Player and Processor.

.

The initial creation of SGRecorder worked without a problem. The Processor and Player models successfully created, stored and rendered multimedia streams. Unfortunately, as system monitoring was increased, the program eventually overwhelmed the computer resources. When enough resources were monitored, the program could not dispatch Swing component events and freezes the system. The system freezes forced the computer to be rebooted before it was again usable.

3.   RecordButton

With simplicity in mind, the RecordButton application was created. The button has minimal functionality with maximum usability. Figure VIII-14 shows the initial interface developed for RecordButton. The simple design records a file as designated in the text box as a RGB, 160x120, IMA4 audio. The other parameters were left for the computer to decide. Initially, the format was only designated as RGB and the computer defaulted to a 320x240 file size. Unfortunately, the file size becomes too large after fifteen minutes of recording.

The program displayed the current status of the program's state. If it was recording, the program displays a "Status: Recording" message. If the program was stopped, a "Status: Stopped" message was displayed. However, the program did not have

144

any fault tolerance. So, if the user hit the record button twice, the program attempted to begin a second recording session while the first was already running. At best, a failed to create processor message was generated. At worst, the machine needed to be rebooted before it was again usable. There was also no video monitoring capability.
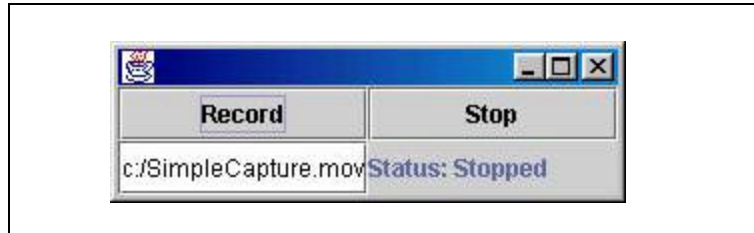
Figure VIII-14:   RecordButton Version 1.

The second iteration of RecordButton overcame the limitations of the first program. First, multimedia streams were recorded in compressed formats of H.263 and GSM. These compressed codecs allowed for extremely small files. The 2 GB file limitation now allowed for the recording of three-hour sessions using the compression. The default dimension for H.263 was 176x144. Also, the capture equipment was unable to achieve full thirty frames per second of capture. It was best to leave this value unspecified so that the computer can capture at the highest rate available. The laboratory equipment normally captured around twenty-eight frames per second.

When a session was recording, the user had the ability to monitor the video and audio of the file being created. The MonitorControls were searched for and identified within the program. The MonitorControls can be displayed under their own frame or as part of the program. They were placed within the viewable panel just below the control buttons. The Controls were check boxes to enable or disable the images and audio tracks of the video. If the boxes were checked, input audio and video were rendered in the panel and through the speakers. The ability to monitor the inputs proved to be extremely important. There were numerous occasions where the wrong audio or video port was configured for the capture and the monitors allowed the operator to realize there was a problem.

145

The last two buttons also provided the final control over the capture and playing processes of the program. The Stop button ends the player or processor and attempts to finish the file writing. Additionally, the Monitor Controls are removed from the panel. After a file was captured, it was trivial to press the Play button and see the multimedia file presented within the viewable panel area. The component system developed in the SGRecorder program was used to render the file. The Stop button removes the components after stopping the Player.

Finally, a system of Boolean flags was utilized to prevent non-available buttons from working. Hence, if the session was recording, the user can press the Record and Play button multiple times and nothing happens. However, if the user clicked on the Stop button, the program now released resources and finished writing the file. The flags prevented the buttons from executing the code capable of crashing the program and cause unpredictable results. Hence, the flag system prevented minimally trained personnel from hitting the wrong buttons and made the program extremely robust.

The main point of failure, despite this robustness, was in the fact that the file writing was not an immediate or real-time process. JMF captures files into a raw format labeled as "nonstreamable". The data in this file is uncompressed and raw. The computer now finishes it's writing by moving the data from the nonstreamable file through the appropriate encoder and into the QuickTime file. A one-hour capture may take up to five minutes to finish the transfer process. This characteristic was also noted under the JMStudio application. An easy method to determine if the program is done is by noting the program's unfrozen appearance or watching the computer's hard disk light and noting that it is no longer blinking in a rapid manner. The final transfer is extremely intensive for the disks. Hence, it is extremely important to not close the program until after the conversion process is complete. Otherwise, both files will be rendered unusable and the capture will need to occur again.
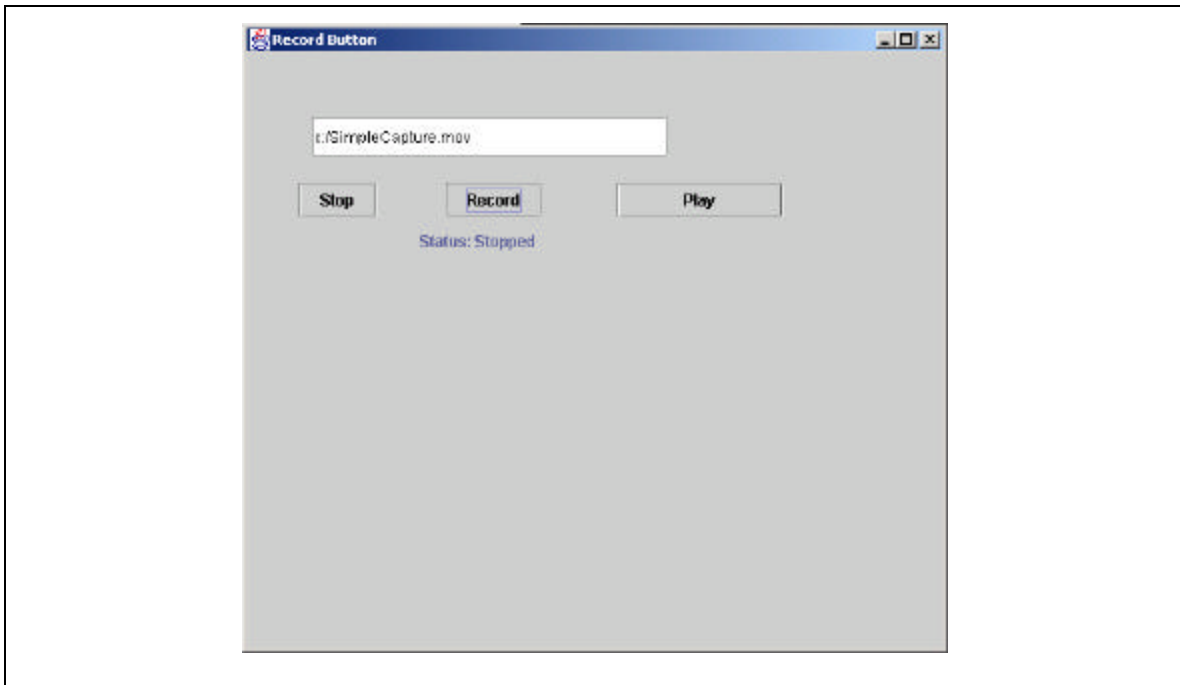
Figure VIII-15:   RecordButton Version 2.

4.   SigTrans

SigTrans was developed for a post capture environment. The program capitalized upon code freely available from Sun Microsystems which was a line command transcoder. By adding the GUI component to the code, users were able to easily transcode a single file into three separate formats and saved as three separate files. The Sun code was used since it had multiple error checking conditions that might have taken weeks of planning and programming to replicate. The graphical interface also allowed for elimination of half of the Transcode class since it was dedicated to parsing the line commands.

The part of the program which remained, provided generic methods of accepting a file name as an input MediaLocator, file name as an output MediaLocator, formats for the output file and times which can be used to splice the video at pre-determined locations. For the SigTrans program, the time splices were not used and therefore set to "–1" so as not to crop the video size. The file names were generated from the form made visible when SigTrans is first ran.

147
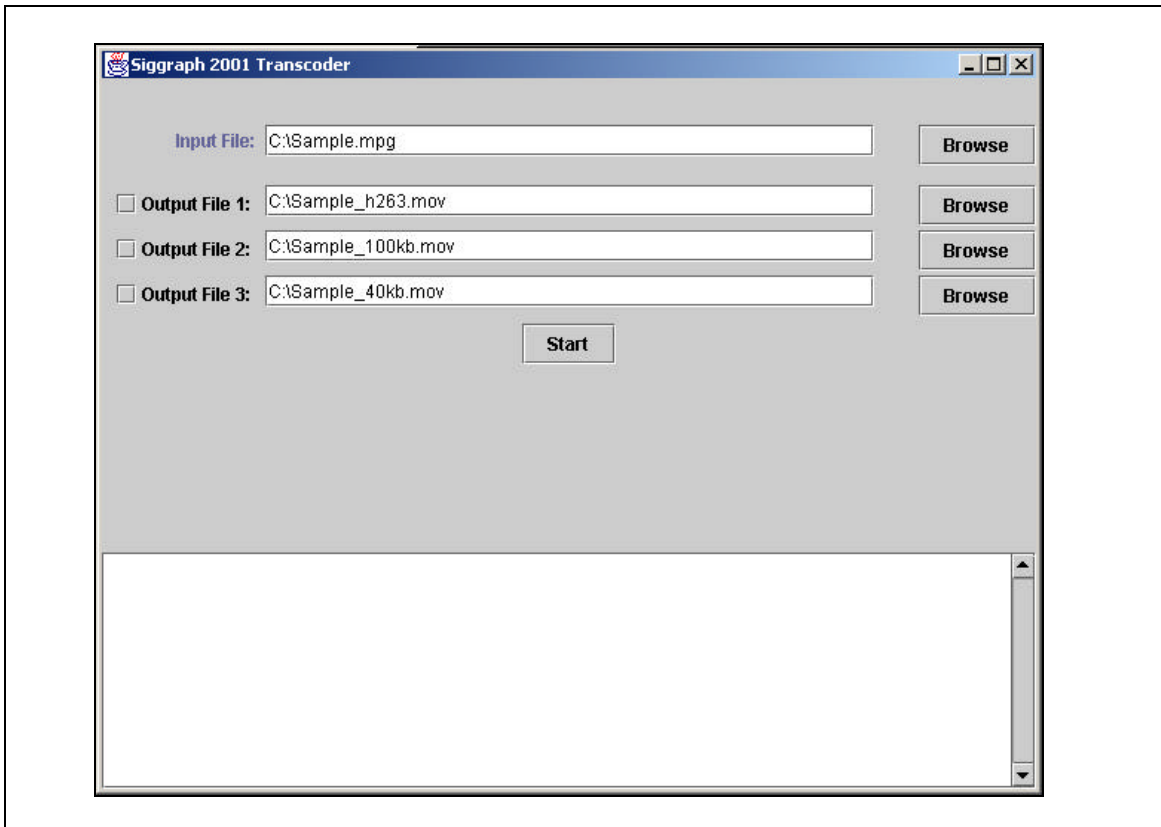
Figure VIII-16:   SigTrans.

Additionally, the SigTrans program used the predefined settings of Apple's QuickTime Player Pro. The default settings for the player's transcoding to 100 Kbps and 40 Kbps high motion and voice are the H.263 video with PureVoice Audio. JMF does not allow encoding into PureVoice, but does allow conversion to GSM. Cellular Phone and Voice over IP companies have used GSM for years. Additionally, the H.263 codec is optimized for the different bit streams by controlling the number of frames per second. For a 100 Kbps stream, ten frames per second are used. For a 40 Kbps stream, six frames per second are used. The default size of 176x144 is also used. These settings can be modified but are the results of extensive testing.

D.   CONVERSION

As was learned in the Dr. Fred Brooks case study, the easiest method to create digitized content of multiple formats is to initially capture the video with the highest fidelity and then convert the program to the lower quality codecs which are optimized for

streaming multimedia content across the Internet. Until now, a program captured in the MPEG-1 format had to be re-encoded by hand three times. The first time was to make it into a H.263 video track with GSM audio. The second time was to run the H.263 video through Apple's QuickTime Pro to optimize it for 100Kb/s stream. The final time was to run it through Apple's QuickTime Pro to optimize it for the 40Kb/s stream. A one-hour video takes approximately half of an hour for each conversion. Hence, a human operator needs to be present approximately two and a half hours for every hour of capture.

The use of SGTrans allows the initial video to be captured in the required formats without any user intervention after the initial setup. An operator can now set up multiple machines to transcode at the same time without having to constantly checking to see if a conversion has already occurred and then configuring the next capture. The reduction in workload and freeing of human resources allows a large-scale capture operation to become a reality with one person.

Naturally, the ability to convert one file to three different formats is only a start. The program can be modified and expanded so that it can convert multiple files to the required formats with a few lines of code. Additionally, the program can be network enabled to allow control from a central location. The central control makes a transcoding farm a reality. A single computer sending the transcoding to separate machines within the farm can control the transcoding of multiple files.

However, care must be taken when capturing or transcoding a file across a network connection. For an unexplained reason, when a JMF application attempts to capture and store a file across a computer network, only half of the information is captured. Hence, the saved video and audio files are halting and jerky. The same equipment under the same environment capturing to a local disk does not experience this problem.

E. SUMMARY

In conclusion, the Hamming Lecture Series presented an opportunity to explore JMF's labor saving functions. The programs developed for this study capitalized upon functionality explored in previous work and then automated the tools so they were applicable to an enterprise environment. Many traps and challenges were realized and

overcome to make the applications as robust as possible. However, certain limitations discovered in capturing directly to a network share, or using codecs not supported by the capture devices, cannot be overcome without customizing the JMF modules. Additionally, since the capture process is extremely resource intensive, additional threads should be kept to a minimum or the computer will crash do to resource starvation. Once these limitations are accounted for, JMF provides the means to capture, render and transcode multiple files with a minimum of human labor.

IX. SIGGRAPH 2001 ONLINE CASE STUDY

A. INTRODUCTION

The capstone of this thesis, Siggraph 2001, was designed as a fully encompassing capture, digitalization and delivery project providing hundreds of hours of content to the Internet for no public charge. The concepts, hardware and software previously explored were used and extended for this study. The number of hours, number of workers and untold technical experience was employed in a monolithic effort never before seen by the computer industry at Siggraph. Teams of volunteers, with accomplished professional credits in the computer industry, lead teams of student volunteers in a process of preserving approximately 250 hours of the world's computer graphics experts. The lectures by these professionals contained a de facto summary of the knowledge encompassing the computer graphics industry with presentations on the current cutting edge techniques and technologies used in the field.

This case study explores the multiple stages of developing the project, working through the project, postproduction and finally the delivery of the content. While it is easy to gloss over the rough edges or omit the parts that did not work, this study will reveal many difficulties that were eventually overcome. With the positive and negative ever present, a goal of this case study is to provide enough documentation so that Siggraph 2002, as well as any large venue, can be captured and delivered to the Internet with a minimum of effort.

B. SIGGRAPH ONLINE PLANNING

Planning for the Siggraph Online Committee began almost two years prior to the event. The committee is structured by placing the following year's committee as the current committee's Assistant Chair. Hence, the committee Chair will have attended two yearlong cycles of planning before becoming responsible for the execution of the current event. In the first year, the Assistant Chair is instrumental in organizing teams, adjusting and justifying budgets, and more. Hopefully, this person can learn a majority of the pitfalls from the current Chair who is finishing their two years in the cycle. The current Chair is responsible for guiding the committee with the experience and ideas gained from

the previous year of work. The cycle continues with a committee always having an experienced leader. The Committee for 2001 consisted of Don Brutzman, Stephen. Matsuba, Mike Collins, Allen Dutton, Juan Gril, Mike Hunsberger, Jerry Isdale, Shane Nicklaust, Nick Polys, Jeff Weekley and Jane Wu.

1.   Location

The Siggraph 2001 Convention was held in Los Angeles, California at the Staples Convention Center. Staples Center contains 900,000 square feet for sports events, entertainment events and conferences. The sports facilities allow seating for 20,000 fans. The facilities contain 160 suites, 32 party suites and 2,500 club seats. Five concourses complete the center.

Additionally, Staples Center is in the heart of Los Angeles. The location allows for easy access to highways, public transportation and the airport. There are also several hotels in the immediate vicinity to lodge the staff, presenters, volunteers and conference attendees. With an estimated 35,000 conference attendees, location and facilities were an important consideration.

In order to gain further situational awareness, the Online Committee met with the STV Committee Chair (Brad Lawrence) and media event coordinators at the Convention Center. The entire committee had two days to inspect the facilities, ask questions and review the planning documents. The meeting in Los Angeles aided everyone in gaining a general overview and formed a small core cadre of team leaders familiar with the area and center.
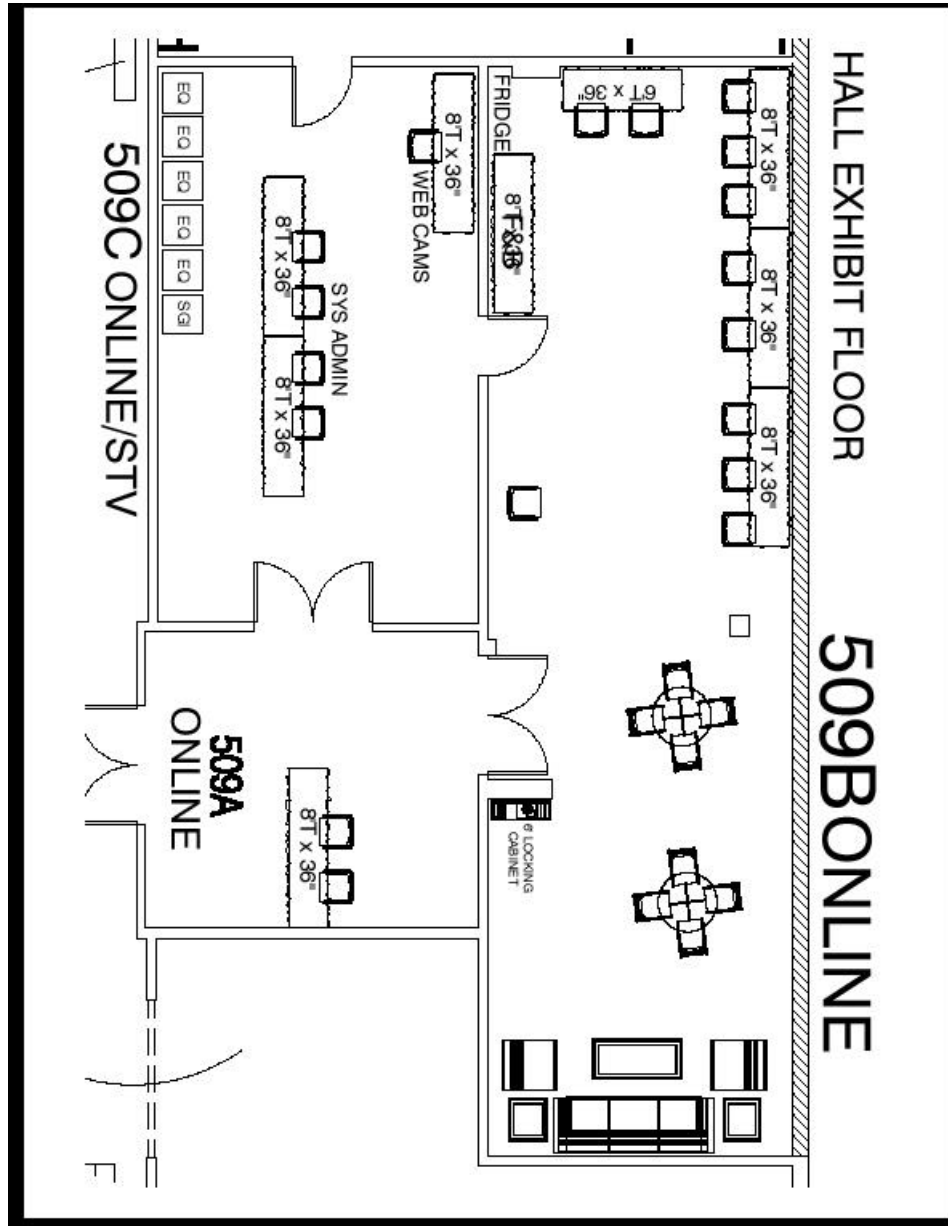
Figure IX-1:   S2001 Los Angeles Convention Center Layout.

2.   Servers and Network

One of the most difficult planning aspects was the use of computers and network connectivity. Most of the equipment was donated for onsite use. Additionally, some volunteers added their personal computers for specific projects, such as web site design

or image synthesis. It was agreed that a minimum of seven computers would be needed. Four computers for capture, one for storage and two for postproduction work. While more computers could be used, five was the minimum. Additionally, the central storage computer would have server software installed and would become the primary host for the web site containing the captured content.

The server was built prior to the event using the parts listed in Appendix G. Microsoft Windows 2000 Server was installed as the operating system. The motherboard showed some signs of having a degraded RAID controller. However, the system still performed to an acceptable level. Additionally, there was one instance where the computer became unbootable due to the faulty disk controller. Reinstalling the operating system fixed the problem.

It was further discussed that one server would not be an optimal situation. First, if the server failed (it was showing strange behavior) the data would be lost. Second, if there were a large number of users, one server would not be able to handle the workload. If there was an expected onslaught of users, one server may be able to handle a dozen to a hundred steaming connections. Hundreds or thousands of connections needs a server farm that is, ideally, geographically distributed. Tens of thousands of connections needs a company such as Akamai, MCI or Sprint to handle the scale of requirements for this level of usage. The computers can be set up using a technique called *Round Robin DNS* to provide load balancing. *Round Robin DNS* is a technique where the domain name servers for the domain name contain multiple IP addresses for the same name. The server will cycle through all of the IP addresses in a Round Robin fashion to distribute the requests. Fortunately, NPS has a Class B range of IP addresses and can allot an address for each server in such a farm. Table IX-1 illustrates some representational network speeds and the number of connections possible.

Connection	Bandwidth	Modem Connections	DSL/Cable Connections
DSL	128kb	2	1
	384kb	7	3
T1	1.5Mb	30	15
T3	45Mb	900	450
OC3	155Mb	3000	1500

Table IX-1:   Number of Simultaneous Streaming Server Connections

An additional concept discussed was the scheduled delivery of select shows on a pre-set time line. The delivery would utilize multicast to limit bandwidth consumption. There would also be fewer computers and less processing power required. While this concept solved several issues, it did not maintain the initial vision of allowing Video-On-Demand (VOD) whenever the user desired. This idea was abandoned for the ability to have users select their download whenever preferred.

3.   Software

With only one computer as a known quantity, software was the second hardest planning area. Certain operating systems would not work with some hardware. Also, the committed assumed all software would need to be installed on site. This placed the burden on committee members to carry all desired software to the location. Utility programs such as Microsoft's Office and Adobe's Photoshop were donated by their parent companies and were provided at the event.

The greatest unknown, however, was the interoperability of the custom Java capture programs and the provided hardware. If the software and hardware were compatible, the media would be digitized using the JMF programs developed in the Hamming Case Study. The XML site builder was also still under construction but would be used with its current DOM model. If all software worked correctly, the server would have a complete site with captured media ready for network connection the day the conference ended.

C.   SIGGRAPH ONLINE EXECUTION

Execution does not always match planning. In the case of Siggraph 2001 Online, almost every technical challenge presented itself. The teams worked long hours to

overcome many limitations and unseen obstacles. In the end, the online site was not ready at the conclusion of the week but approximately two months after the conference ended. Due to hardware failures, software incompatibilities and other challenges, a majority of the conference was captured and digitized but not placed into its final format and location until the postproduction phase. These issues are discussed in Section D.

1. Teams

The team members kept in contact for months prior to the event through the Siggraph Online List Server. Messages sent to the list were distributed to all members. Correspondence between the committee members was primarily through this list. Also, the list was used to insure everyone on the team was familiar with the issues and plans. Anyone could use the list server to ask questions and distribute personal information such as flight times and hotel arrangements. The flow of information was crucial since there were so many people arriving from so many disperse locations.

There were a total of five teams consisting of two committee members and three volunteers. The teams were responsible for the capture, digitalization and storage of scheduled events. Committee members were responsible for creating a work schedule, meeting the volunteers, volunteer orientation, transportation and general organization of the event.   Additionally, an office was employed as a single point of contact and central meeting place. Message boards, telephones and Internet access were utilized in this office.

A majority of the volunteers were college students from around the world. Students came from Brazil, Venezuela, England, France, Italy, New Zealand, Norway and the United States.   Some were PHD students or had just finished work at the graduate level. All of the volunteers were affiliated with computer graphics or information technology. Additionally, each student was required to commit a minimum of thirty hours to qualify for a scholarship covering his or her room, board and admission to the conference. The committee members arrived Friday, August 10 and left on Saturday, August 18. The student volunteers arrived on Saturday, August 11 to meet in the committee office of Room 509 at the Staples Center. The orientation schedule for Saturday is contained in Table IX-2.

- welcome and introductions (Don Brutzman and committee)

- what are we doing

- access, policies, passes, badges, administration items

- teams and schedules

- student volunteer specific items

- internationalization (i18n) support

- questions & answers

- site walkthrough: classrooms and big rooms, speaker prep

- lunch break provided at LACC

- review daily routine, further Q+A

- verify hotel/phone/departures on team spreadsheet

- handout & discuss content preparation checklist

- recording software demo, working with AVW

- Brooks exemplar and site builder demo

- getting ftp materials/slidesets from speaker prep archive

- network access and email

- content-preparation checklist review

- equipment, logistics, insurance review, shipping procedures

- room 509 office procedures, security, access

- online/STV celebration and party planning

- hello from STV, Pathfinders (Brad Lawrence, Mary Nichols, Barb Helfer)

- hello from SIGGRAPH 2001 chair (Lynn Pocock with assistant Larry Vliet)

- preparing committee's people page: pictures, story, links

- registration

- team time to practice software, discuss, walk around, etc.

- regroup before dinner: "Are we ready for sunday courses?"

Table IX-2:   Siggraph 2001 team orientation.

2. Capture Process

Teams were responsible to capture the courses listed on the schedule contained in Table IX-3. Sunday's courses were only half days. The times were listed for the classes online and in the central office. Monday through Friday courses and papers/panels occurred between 8:30 and 6 PM. Teams needed to cover all sessions occurring in their assigned rooms. Team member schedules were assigned within the individual teams. Every person assigned to a team was not expected to be available from 8 PM - 6 AM everyday. Team leaders were responsible to allow members to see sessions they were interested in and have a good time at the conference. This scheduling process worked quite well since team members generally captured sessions they were interested in and spent the rest of the time exploring the demonstration floor or seeing Los Angeles.

	Sunday 12 Aug	Monday 13 Aug	Tuesday 14 Aug	Wednesday 15 Aug	Thursday 16 Aug	Friday 17 Aug
Team A	Training / Augment	515A	WHA	Hall A	Hall D	Office
Team B	Training / Augment	WHA	Office	Hall B (1/2 day)	Hall B (3/4 day)	Hall A
Team C	PHC 1:30- 5PM	Office	515A	Hall C	Hall A	Hall B (1/2 day)
Team D	WHA 1:30-5 PM	Training / Augment	PHD	Hall D	Office	Hall C
Team E	WHB 1:30-5 PM	PHC	Training / Augment	Office	Hall C	Hall D (1/4 day)
Office Mgr	Kat Polys	Kat Polys	Kat Polys	Kat Polys	Kat Polys	Kat Polys
	Katie (when avail)	Katie (when avail)	Katie (when avail)	Katie (when avail)	Katie (when avail)	Katie (when avail)

TEAM ASSIGNMENTS

	Committee	Committee	Committee	SV	SV	SV
Team A	Alan Dutton	Mike Collins	Jane Wu	Emmanuel Turner	Marcio Calixto Cabral	Boris Mansencal
Team B	Shane Nicklaus	Jeff Weekley		Boris Mansencal	Alexis Paljic	Francesca Odone
Team C	Nick Polys	Jerry Isdale (Tues-Thurs)		Zoran Constantinescu	Antonio Pacheco	Onofrio Petruzzella
Team D	Stephen Matsuba	Juan Gril		Danilo Liendro	Luis Jimenez Navarro	Ana Gabriela Marquez
Team E	Mike Hunsberger	Doug Horner		Isaiah Bellais	David Bornfriend	Catherine Mullan
Possible Augmentees	Fred Zyda	Hans Lee				

Table IX-3:   Siggraph 2001 team assignments and schedule.

Once the teams were acclimated to the convention center, they were exposed to the capture process. The originally planned software proved to be incompatible with the capture hardware. Several hours of troubleshooting led to the abandonment of the Java capture programs. The computers used for the capture process were donated or rented with old drivers, old Java Runtime Environments and multiple operating systems. Drivers could not be downloaded from the Internet since access was not available until the following day. Even with the diverse computer knowledge available onsite, the task proved to be not viable. Instead, Windows 2000 and Adobe Premier were installed on the computers for the capture process. Even with a professional program such as Premier, the captured session soon lost sound and video synchronization. After consulting with the Adobe team, the problem was isolated to the capture cards and to the fact they were not certified by Adobe. The solution was to decrease the frame capture rate so video frames would not be dropped and lose synchronization with the audio.

Once the capture process was finalized with the available hardware and software, students were ready to begin the capture process. The streams were captured in an uncompressed AVI file with a dimension of 320x240. Once the file was transferred to the central server, it was transcoded by Adobe Premier. Figure IX-2 demonstrates the capture process.
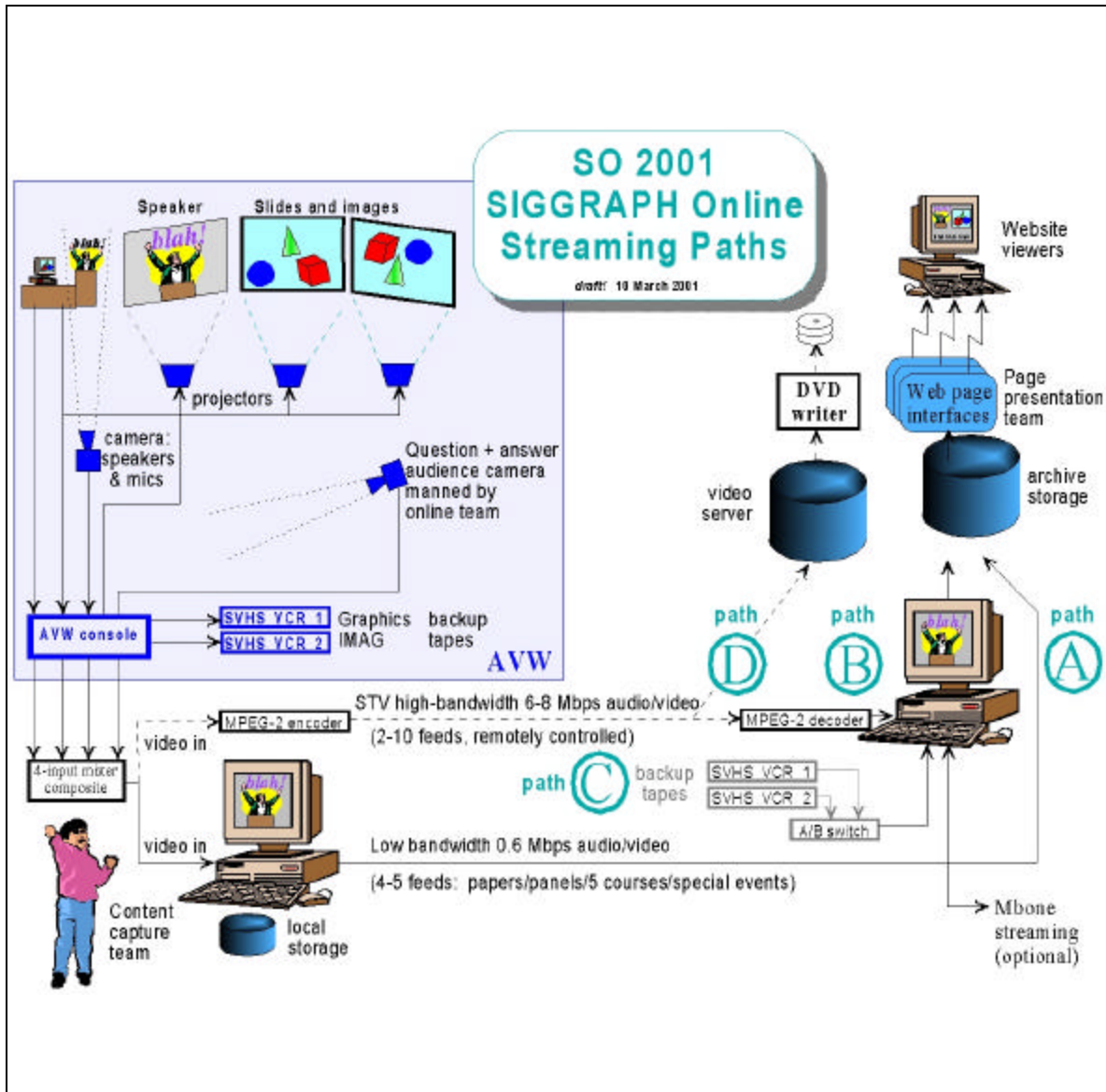
Figure IX-2:   S2001 capture diagram.

3.   Lessons Learned

a.   File Size

Many of the lessons learned were a direct result of the JMF solution not being implemented. If the capture cards were compatible with JMF, files would have been captured in a compressed H.263 format, 10 frames per second, 160x120 dimension and GSM audio. This format was the agreed upon acceptable size for streaming content. By capturing the content at this resolution, it can be directly placed in the correct

160

directory on the main server. There would not have been issues related to the capture of multi-gigabyte files, which was required with the ad hoc solution utilized at Siggraph.

Challenges created by the large file size were storage, transference and postproduction. Since the server, aptly named "Beast", was the only machine with a RAID configuration large enough to store 250 hours of data captured at the rate of one gigabyte per hour, all files needed to be stored on this machine. Local computers would capture a session and store the data locally. During the lunch break or at the end of the day, the multimedia files were transferred to Beast and the drives were cleared to make space for the next round of sessions.

b. Network Utilization

The network would become saturated when four machines would attempt to simultaneously transfer data to the one machine. Data loss and network non-usability plagued this situation. Additionally, when files were accessed from the server for postproduction, network saturation and delays resurfaced. The postproduction also became a problem since there was a .33 to 1 ratio of processing time to content time. This meant that ninety minutes of video required thirty minutes to process. Since there were only two machines available for postproduction, batch conversion jobs would run through the night and barely finish by the next round of capture sessions.

These challenges would have been overcome with the JMF solution. However, if a JMF solution is not available, the problem could have been solved with additional software. The file transfer issue can be solved with removable hard drives and additional storage on local machines. Additional processing machines would have increased capabilities and allowed for the transcoding of multiple formats. Adobe Premier has a batch transcoding capability allowing it to divide jobs amongst a server farm. Additional licenses are not required and each computer increases the speed of the overall conversion system. This system might require a gigabyte switch attached to the central machine with the RAID drive.

c. Standardized Configuration

The JMF incompatibility issues stemmed from the reliance upon donations. With the committee accepting any and all computer configurations and not

having the ability to test software ahead of time with the new hardware, incompatibles were resultant. Having a standardized software and hardware solution proven before the event would have eased concerns and troubleshooting during the actual affair. Rented or purchased computers and capture cards could have provided this capability. If Adobe Premier is used in the future, Adobe Certified capture cards must be utilized.

Additionally, the volunteers can become familiar with the software before the conference. The problems experienced at Siggraph 2001 placed the committee behind the capture process and removed confidence in the newly agreed upon solutions. Finally, a standardized software configuration could be imaged and mirrored on computers of similar hardware. Hence, each system would operate exactly alike.

During the 2001 conference, the original machines were loaded with Windows 98. The capture cards did not work under this operating system. Also, there were old drivers on the systems. Since there was no Internet access for the day before the conference, Windows 2000 was loaded onto each machine from a single copy located on a laptop computer. There was no means to create a CD-R to accelerate the installation process. Hence, there was much stress from the beginning.

d.   Backup

The reliance upon one machine containing a RAID array, which had been acting strangely, heightened the risk to an unacceptable level. With only one copy available after several hundred volunteer hours, the risk was too elevated. DVD-RAM or additional RAID arrays would have reduced tension and the margin for failure. The RAID controller eventually failed after all of the capture and ninety percent of the post processing was completed. Through the patience of Jimmy Liberato, the Moves Institute Network Administrator, and several others, proprietary utility programs were obtained from the RAID manufacturer and the array was reconstructed on a similar motherboard without loss of data. If the utilities did not work, the only options left were to use a data recovery company with a starting price tag of $15,000 or to begin re-digitizing the entire effort. The entire process, of contacting the company and reconstructing the drive, consumed more than a month of time.

Once the array was reconstructed, it was immediately copied to another 320-gigabyte array attached to a more robust PCI RAID controller from Promise Technologies. The original RAID controller was built into the motherboard and not able to support the removal of the array and attaching it to a similar array on the same type of motherboard. The Promise controller allowed the array to be moved from computer to computer with minimal driver installation of the controller. The new controller provided a mobile and more fault tolerant solution.

Additionally, the NPS Information Technology division was in the midst of installing a large tape array backup system. The system was not available for use until approximately three weeks after the second copy of the Siggraph files was created. However, once the Tivoli system was in place, 290 gigabytes of data was backed up across the school's network in less than twenty hours. With three complete copies, the work continued by modifying and improving content on the original array.

e. *Dedicated Administrative Support*

It is imperative to have dedicated administration support during the conference. First, a system administrator is needed to supervise and provide a single view of network configurations. By having a single person in charge, configurations are standardized and there is less confusion. Other committees also have a single point of contact since all committees use network technology. Additionally, the Online Committee and STV committee were so interlocked by the similarity of technologies, both system administrators could trade roles or assist the other during emergencies.

Second, a single person overlooking the teams to insure work was equitably divided is beneficial. Work division was accomplished in a fair manner, but if the teams become larger, it may become more of an issue. Ideally, there should be a minimum of six members on a team. Now, the teams can be split in half for a capture or transcoding session. Also, committee personnel should be dedicated to high priority or difficult problems and not assigned to teams.

The human resource oversight person can also handle student daily check in and generate a report of hours worked. Since the teams were working independently, it did not make sense to have a daily check in at the same time for everyone. Some teams

worked late or early, but everyone still had to be present at the same place and time. A team muster may appear more equitable to the volunteers.

This person can also intercept the public since many people were looking for the Internet access rooms close to the office space. It was difficult to accomplish the tasks at hand without also having to constantly help the public find the nearby rooms. Signs would have also alleviated the problem by aiding the public with directions.

f.   *Increased Postproduction Environment*

The regeneration of content form videotape, while possible, is never a desired solution. Every effort should be taken to insure a session is captured correctly the first time. Even though capture machines did not crash during capture sessions, it is a real possibility. Also, every capture application explored for Siggraph had a small failure rate, which would require the rebooting of the computer. The instant a program fails, the entire session needs to be recaptured from tape at a later time. A better solution is to have multiple tape and digital capture systems in place. If there is no failure, extra copies can be deleted. If there is a failure, the second or third system can still perform the capture during the first take.

Additionally, it is naïve to believe that there will never be a need to perform postproduction editing on digitized content. No tool was ever identified for postproduction until the conference had already begun. An application should be agreed upon early in the development process and users should be familiar with it prior to the event. There also needs computing power commensurate with requirements. At minimum, three computers are necessary for each capture session. One for captures, one for editing and postproduction and one for web development site building. Purchasing capture cards with hardware compression built in to insure a streamable format on the first attempt can eliminate some effort.

Ideally, having web sites ready with slides and content, but missing the captured files is the best prepared solution. Unfortunately, this amount of preparation relies too much upon the presenters. Some presenters do not have finished slides until immediately before they walk onto the stage. Others modify the slides during the

164

lecture. Also, many are hesitant to handout their slides or sign permission-of-use agreements.

D.   SIGGRAPH ONLINE POSTPRODUCTION

Unfortunately, due to hardware failure, file conversion and site building was not completed during the Siggraph Conference. Additionally, multiple hardware failures after the event forced delays two months after the conference was completed. With patience and perseverance, the effort was redoubled once the hardware difficulties were overcome.

1.   Additional Capture

While the servers were crashed and awaiting components, several lectures, which were not scheduled for the initial capture, were converted using the software and programs developed using the Java Media Framework. *Sensapalooza* and *Monte Carlo Ray Tracing* were converted into multiple formats in the first week. These two sessions were used to exemplify what could have been accomplished if there was not a hardware/software conflict during the conference.   The sessions were captured with the highest resolution possible and then transcoded into multiple formats.   The laboratory, where the digitalization occurred, had thirty networked computers capable of running transcoding software. By farming out multiple conversion tasks to the full array of computers, it was a trivial matter to create content in dozens of formats. In fact, it became difficult to keep track of the many versions and copies of media available.   Disk space was quickly consumed and backup space became an issue. All finished media was eventually co-located on Beast.

2.   Conversion

It was discovered that some Adobe projects were prepared but never processed. Adobe Premiere was installed on the computer and the projects were processed. Since there was such a long period of inactivity before the server became operable, many of the volunteers had dispersed and become unreachable. This disconnection left many hours of searching the directory structure looking for half finished or misplaced work. Since there

were so many people involved in the creation of the project, better documentation could have prevented the lapse. Checklists and quality checking would have insured the work was not performed in vain.

3.   Final Configuration

After its arrival and repair at NPS, the Siggraph RAID array was attached to a dual 800 PIII Dell Workstation. A request was made and granted to make the computer's Port 80 visible through the school's Raptor firewall.   Apache Web Server software was installed on the machine and the files were made publicly available to the committee members. As discrepancies were reported and corrected, the files were eventually made available to the whole of Siggraph. When usability and reliability can be confirmed, multiple web servers shall be configured, DNS servers shall be modified and point the public to the computer. Finally, a press release will announce the existence of Siggraph Online.

E.   SUMMARY

The Siggraph Online case study demonstrated how the best-laid plans could go awry. With hundreds of components and human workers involved in this single project, small problems and discrepancies soon grew to insurmountable challenges. Quick thinking and a large depth of computer expertise constantly pulled the project back into a workable arena. Eventually, multiple hardware failures corroded confidence in the project and delayed the effort for an unacceptable length of time. Funding for a similar effort for the following year's conference was contested while there still were no products to demonstrate for the current year. Despite heavy criticism and declarations of failure from "armchair quarterbacks" the work was continued to completion and the world now has access to some of the most profound knowledge in computer graphics.

X. CONCLUSIONS AND RECOMMENDATIONS

A. MULTIPLE SOLUTIONS

Distance education lends itself to a variety of solutions. Unfortunately, many of these solutions are expensive, proprietary and difficult to use. Increased capability is normally associated with increased cost and increased complexity. If an organization has the resources and desire to pursue large, customized solutions, the end product normally requires a maintenance agreement to support its product cycle. Smaller and less complex solutions may also have hidden costs increasing the total cost of ownership. Some of these solutions lock the purchaser into required hardware and software that can only be purchased from the one provider. Proprietary solutions may appear attractive at first, but normally end in rising costs and the inability to change solutions at a later time.

Hence, many organizations choose less expensive and less proprietary solutions. They may purchase individual components standardized to interoperate with other components. For instance, users are first divided by operating systems utilized. "Will the consumer use Apple, Microsoft, Sun, IBM, Beos or Linux operating systems?" Once the operating systems and computer equipment is decided upon, capture hardware and software is chosen. Once again, proprietary components may be chosen. However, now the organization may choose a particular technology. If they use Microsoft products, they may continue in this venue. The same can occur for any operating system provider. In the end, most organizations pursue the technologies of which their people have been trained. The costs are not as immense as full, proprietary solutions but are still substantial.

A final solution is to pursue open-source components and insure that they interoperate sufficiently to provide the required functionality of the organization. Until now, open-source tools have been restricted to few uses with limited large-scale applicability. Some are written in older computer languages requiring months of study and limited code reusability. This paper has shown that current standards of Java extended by multiple APIs is more than able to handle the requirements of distance education and improve upon its deliverance of knowledge. Additionally, programs

created in Java can transcend the normal limitations of multiple operating systems and dissimilar hardware. Finally, the multiple open-source libraries and large number of Java programmers exponentially increases the usability and extensibility of programs developed for the Distance Learning Environment using the Java language.

B. VALUE ADDED

The use of the Java programming language allowed for multiple advantages and generated an unforeseen synergistic effect in the capturing, transcoding, storage and transmission of multimedia content. With the MBone tool set as the only open-source tools available to send and receive multimedia content, the use of JMF allows a programmer to mimic and exceed the current level of information transfer.

The main value in using JMF to create distance learning tools is that multiple hooks into other Java APIs increases the multiple uses of the JMF tools. Java's Speech API can allow words to be transcribed to text, text transcribed to speech and voice control over DLE components. Java's 3D API allows natural mapping of tools as well as innovative display methods of previously non-imagined display methods. Java's K Virtual Machine allows these interfaces to run on small, portable handheld devices. The list continues and grows. However, the use of Java in Distance Learning Environments insures that it will be expanded and improved as computer technology increases. JMF allows for proprietary, expensive solutions to be duplicated and improved for a fraction of the original cost and reduction in complexity.

C. FUTURE WORK

This paper may have generated more questions than it has answered. The following four sections deal with the main areas left open by the explorations of this thesis. The ability to finally control the capturing and modification of video and sound open a plethora of possibilities. Areas such as automating this process are still in their fledgling state and offer exciting future possibilities. Also, JMF has the ability to become an editing suite on the scale of the most powerful suites available today. The ability to use and create filters and plug-ins gives JMF great flexibility. Finally, the interaction of humans, video and sound create an interesting test bench with Java Media Frameworks and streaming content.

1. Tomcat/ Linux Transcoding Server

The work in this paper was stored and delivered from a Microsoft Windows 2000 Server. While this approach was effective and overcame many compatibility issues related to hardware, other approaches have similar merit. As part of the open-source effort of this paper, a Linux server with Apache or Tomcat can be utilized to deliver the captured content.

An interesting concept presented by the JMF team at JavaOne 2001 in San Francisco, California was to use one computer to capture, digitize and stream a particular feed. Then, one or many computers are subscribed to the multicast group that is supposed to receive the video and audio stream can save or transcode the stream in one or more formats. Other machines can display the current capture in real-time for quality monitoring or an instantaneous audience presentation. Since the stream is using the Real Time Protocol (RTP), it can send the data to any or all computers that are connected by a multicast enabled route. After the streams are transcoded to multiple formats, information is generated from the new files such as size and the extraction of a single frame into a Joint Photographics Expert Group (JPEG) format. All of this information and transcoded files are placed on a Tomcat Server.

This approach offers numerous advantages. As new formats and codecs become popular in the market place, the software can be upgraded with only slight modification. The example at JavaOne allowed for the transcoding of the capture video into a Quicktime format of two video and audio sizes as well as a MPEG-4 encoding solution. The small size of the MPEG-4 stream allowed the video stream to be displayed on a Personal Digital Assistant as well as a Web Enabled Cellular Phone. All of these technologies warrant the effort of a thesis.

2. Modifying Video and Audio using JMF

Java Media Frameworks offers many interfaces to allow a programmer to grab, modify or transcode video and audio tracks. Some of the strengths of JMF are its ability to decouple multiplexed tracks, modify the individual tracks and re-combine them into a single stream. More advanced programming is available where JMF can actually grab

individual frames from the video or audio buffer and change its characteristics. For instance, a neon filter can be added to music video or sound levels may be raised during a quiet or poor audio capture.

JMF provides for these capabilities using codecs, effects and stream buffers. Although documented in the API, more advanced examples can be pursued to create a multimedia-editing studio capable of splicing, editing and combining multiple tracks of audio and video.

3. Human Components Interface (HCI)

This thesis is the basis for many avenues of discovery in the HCI field. The ability to tightly capture, modify and display video using the technology presented in this paper offers the ability to measure human tolerance and augment daily reality. The human interaction with video and audio can be tested and modified for hundreds of scenarios.

a. *Determine Quality Perception and Tolerance Levels*

Determining the differences in human perception is both time consuming and exacting. The codecs available in JMF allow videos to be created and transcoded to multiple formats. The granularity of control offered by this API allow for the creation of exact testing parameters for human perception of audio and video. Tests can determine how much noise is tolerable in both tracks. Viewing tolerances can also be accessed for distance learning as opposed to entertainment environments. The question needs to be asked of "How low can the quality be for effective learning to occur." The reduction in audio and video quality directly impacts bandwidth considerations. These considerations will affect cost for delivery and end user requirements for streaming media.

b. *Secondary Data Track Overlay*

Information from a secondary data track can be either overlaid or superimposed over the original video and audio feed. Example sources can include Geographic Information System (GIS) feeds such as geographic coordinates or compass information. Audio from a second member in a group or expert navigator can be added to the audio track to increase situational awareness. A closed captioning system can also

be utilized to clarify indistinct verbal directions. The extensible nature of JMF makes all of these scenarios and more quite possible.

c.   Augmented Reality Studies

With the decreasing size of computer hardware and increased wireless bandwidth, augmented reality studies may become feasible in the near future. It is foreseeable to have cameras co-located on the subject as they walk through a store or college campus. The cameras capture the images, have a computer program augment the images and then display the newly modified images to the person wearing a small heads-up-display (HUD). The same effect can be accomplished for sound. The person is now able to see his and hear surroundings, as they are accustomed. In fact, stereo-optics is available with the use of two cameras mounted just in front of the user's eyes. Stereo hearing is possible with two microphones on either side of the head. A small computer on the person can perform augmentation tasks from a local database or a larger computer can remotely handle difficult or computationally intensive tasks.

Almost all human endeavors can be amplified and made more efficient by the use of augmented audio and video. Mechanics working on cars or aircraft can have schematics available as they are looking at an engine. Consumers can instantly compare prices from a dozen sources simply by looking at a product. Extra sensors can be integrated such as geo-positioning or temperature. Heart monitors and breathing rates can provide biofeedback. The possibilities are almost endless.

4.   Camera and Studio Automation

The Distance Education and Siggraph Case studies demonstrated the large number of people necessary for the production of a multimedia capture event. The Naval Post Graduate School was highly automated and still required at least two people to capture the class. One to operate the cameras and viewpoints while another was needed to control display item and highlight important data in the presentation. Siggraph required two trained camera operators, a large number of set-up personnel, a production booth staffed with trained personnel and several volunteers for managing the switching of feeds as well as running capture software. Future work can reduce the number of people necessary by automating many of the processes. For instance, switching feeds can be

automated based upon time, lecturer switching slides, speech recognition or a set algorithm. Cameras can move based upon a pre-defined pattern or pointed based upon a transmitter located on the speaker. Additionally, the software created for capture in JMF can be augmented to include even larger batch processing or central network control to provide a larger selection of media without an increase in human workload.

5.   Replacing MBone ToolSet

Currently, the only open-source widely available distance learning program set is in the form of the MBone tools. Once a programmer is familiar with JMF, it is a relatively straightforward endeavor to replace the tools with Java implementations. Once the tools are replaced with similarly functioning Java applications, the Java MBone tools can be hooked into multiple Java applications. Additionally, code re-utilization can allow the programs to gain added functionality. For instance, new user interfaces such as three-dimensional maps can allow users to have a more natural and intuitive mapping of where video feeds are originating. Finally, the skills of 2.5 million Java programmers can now be applied to the challenges associated with distance education.

6.   Wireless LAN Implementation

The network requirements for distance education can be exacting and precise. The conversion to a truly mobile environment such as a wireless campus LAN increases these requirements to a greater magnitude. Additionally, less bandwidth is traditionally available in a wireless network as compared to a hardwired system. The protocols and codecs utilized for this environment need optimization for the restrictive setting. The use of mobile devices with multiple indoor and outdoor access points extends the distance learning tools into an entirely new venue.

7.   Advanced Distributed Learning (ADL) Compliance

The Advanced Distributed Learning (ADL) initiative is a collaborative effort to standardize the availability of high-quality distance education tools. The goal is to create a common technical framework for large-scale, rapid development of reusable distance learning content. ADL desires to create usable standards, develop towards future network technologies, promote collaboration, and use commercial-off-the-shelf products. A primary contribution has been the development of the Sharable Content Object

Reference Model (SCORM). It is a solid standard for the education and training community [http://www.adlnet.org/].

Future work would include the development and publication of NPS's ADL participation and initiatives. Several hundred organizations including NPS are partnered with ADL, an active pursuit of documenting and developing contributions to the process is appropriate. By contributing to ADL, NPS may develop a more efficient and cost-effective distance-learning environment.

D. REVIEW

This paper began with the large-scale vision of evaluating the current state of distance education, cataloging tools available for the dissemination of classroom lectures and attempting to improve upon distance education's current status. One goal was to evaluate the techniques to convey knowledge from an instructor to a student over geographic and temporal distances. Another goal was to minimize costs. A final goal was to use open-source tools to accomplish the task.

By using Java Media Frameworks and low-end video and audio capture devices, experiments proved that inexpensive distance learning can be created with multimedia streams. By understanding the Distance Learning Environment, multiple streams can be optimized to insure that a maximum amount of knowledge can be conveyed to a geographically disperse audience. By using the JMF extension of Java, open-source tools can be ran on any platform with a Java Virtual Machine; including platforms yet to come. Also, JMF is firmly hooked into the other Java APIs to allow integration of truly complex and exciting products.

It should also be noted that the more complex and expensive learning systems were not a perfect solution. Some of the most expensive software and proprietary hardware still faced limitations. Mostly because of commercial enterprises inability or unwillingness to participate in open development markets, the products shall retain defects that can only be discovered by the scrutiny of a large and open community.

The proprietary system incorporated at NPS, although robust, still faced several challenges larger than the system itself. Table VI-3 summarizes many of the difficulties

that need to be resolved to improve the system. As the VTC case study illustrated, a few simple corrections could drastically improve the usability of the system incorporated by the entire United States Navy.

The programs created for this thesis and the distance education evaluations provide a solid baseline of knowledge for the open community. This work is capable of creating and sustaining a distance learning enterprise. The multimedia captured for this thesis will allow access to knowledge previously reserved for a few. The words and thoughts of Dr. Fred Brooks, Dr. Richard W. Hamming and two hundred cutting edge computer graphic professionals is now available for the world to learn from and enjoy. With this baseline in place, future applications can only be more exciting and robust as knowledge is captured, stored and transferred for future generations.

LIST OF REFERENCES

Afonso, Francisco, "Virtual Rality Transfer Protocol (VRTP): Implementing a Monitor Application for the Real-Time Transport Protocol (RTP) Using the Java Media Framework (JMF)," Naval Post Graduate School, September 1999 available at: http://www.web3d.org/WorkingGroups/vrtp/rtpMonitor/

Allan, Gabriel and Evan Schumann, "*Broadband to Go*," Business Week Online, July 2001 available at *http://adsections.businessweek.com/broadband/future/bbhome.html*.

Apache http server, The Apache Software Foundation, 2001. Information available at: http://www.apache.org/.

Bureau of Census, "Availability and Use of Selected Teaching Resources: 1996," *Statistical Abstract of the U.S.: 1999*, Congressional Information Services, Washington, D.C., 2000.

Carnevale, Dan, "A Distance-Education Student Puts His Lessons to Work on a Farm," *The Chronicle of Higher Education*, Washington, June 1, 2001.

Carswell, Linda and David Benyon, "An Adventure Game Approach to Multimedia Distance Education*," Proceedings of the Conference on Integrating Technology Into Computer Science Education*, Barcelona, Spain, June 1996.

Chen, Herng-Yow, Yen-Tsung Chia, Gin-Yi Chen and Jen-Shin Hong, "*An RTP-based Synchronized Hypermedia Live Lecture System for Distance Education*," Proceedings of the Seventh ACM International Conference on Multimedia, Orlando, Florida, Novermber 1999.

Chery, Y., "Bringing the Common Whiteboard into the Digital Age," *IEEE Multimedia*, Boston, Massachussetts, April-June 2000.

Chute, A., L. Balthazar, and C. Poston., "Learning from Teletraining," *Readings in Distance Learning and Instruction*, University Park: Pennsylvania State University, 1989.

Cordoni, John R. and Robert J. Tucker, "Tools for Higher Education Distance Teaching," Written for for *Proceedings of 26th SIGGUS on User Services*, Indianna, October 1998.

Davis, Kelly and Tom and Charles Price, "Distance Education using Linux and the MBone," *Linux Journal*, Issue 79, 2000.

DeCarmo, Linden, *Core Java Media Framework*, Prentice Hall PTR, Upper Saddle River, New Jersey, 1999.

Director of Naval Training(N7), The Navy-Wide Distributed Learning Planning Strategy, Chief of Naval Operations, Washington, D.C., 1998.

Ehrmann, S. C., "Moving Beyond Campus-Bound Education*," Chronicle of Higher Education*, July 7, 1995.

Guha, Smita, "An Effective Way of Teaching Early Childhood Eduction On-Line," *Childhood Eduction*, Washington, Summer 2001.

175

Hamilton, David P., "*No Substitute: The Internet Does Not Change Everything*," Wall Street Journal, March 12, 2001 available at: http://interactive.wsj.com/articles/SB984075474665122682.htm.

Heilig, Morton, "El Cine del Futuro: The Cinema of the Future," *Presence*, Massachusetts Institute of Technology, Summer 1992.

Heinzl, Mark, "*Telecoms Hunt for Broadband 'Killer Apps' To Use Nearly Empty Data Delivery Pipes*", Wall Street Journal, June 14, 2001 available at: http://interactive.wsj.com/articles/SB9924786557582760.htm.

Jain, R., "A Revolution in Education," *IEEE Press*, Boston, Massachusettes, January-March 1997.

Kirkpatrick, D. L., *Evaluating training programs: The four levels,* San Francisco: Berret-Koehler Publishers, Inc., 1998.

Laplante, Philip A. and Stoyenko, Alexander D*., Real-Time Imaging: Theory, Techniquies and Applications*, IEEE Press, Piscataway, New Jersey, 1996.

Laurillard, Diana, Jenny Preece, Ben Shneiderman, Lisa Neal and Yvonne Waern, "*Distance Learning: Is It the End of Education as Most of Us Know It?"* Proceedings of the Conference on CHI 98 Summary: Human Factors in Computing Systems, Los Angeles, California, April 1998.

Lomic, Marijana and Zoran Putnik, "*On Distance Education Courseware*," Proceedings of the 4th Annual SIGCSE/SIGCUE on Innovation and Technology in Computer Science Education, Krakow, Poland, June 1999.

Mariani, Matthew, "Distance Learning in Postsecondary Eduction: Learning Whenever, Wherever," *Occupational Outlook Quarterly*, Washington, Summer 2001.

MIT News Release, *MIT OpenCourseWare Fact Sheet*, April 4, 2001. Available at: http://web.mit.edu/newsoffice/nr/2001/ocw-facts.html.

National Science Foundation, "Number of Different Distance Education Courses Offered by 2-Year and 4-Year Higher Education Institutions in 1994-95 and 1997-98," *Report of Science and Engineering Indicators, 2000*, Congressional Informational Services, Washington, D.C., 2000.

Neuman, W. Russell, "Beyond HDTV: Exploring Subjective Responses to Very High Definition Television," Massachusetts Institute of Technology Audience Research Reprint Series, Cambridge, Massachusetts, 1990.

Office of Eductional Research and Improvement, *Projections of Education Statistics to 2010*, Congressional Information Services, Washington, DC, August 2000.

Phillips, J. J.,*Return on investment in training and performance improvement programs.* Houston, TX: Gulf Publishing Company, 1997.

Pullen, Mark J., "The Internet-Based Lecture: Converging Teaching and Technology in Computer Science Education," Written for *ACM Conference on Information Technology in Computer Science Education*, July 2000.

Pullen, Mark J. and Michael Benson, "ClassWise: Synchronous Internet Desktop Education," *IEEE TransEd*, November 1999.

Roach, Ronald, "Safeguarding against online cheating," *Black Issues in Higher Eduction*, Reston, June 7, 2001.

Robinson, Matthew and Pavel Vorobiev, *Swing*, Manning Publications Co., Greenwich, CT, 2000.

Roccetti, Marco and Paolo Salomoni, "Web-Based Synchronized Multimedia System for Distance Education," *Proceedings of the 16th ACM SAC2001 Symposium on on Applied Computing,* Las Vegas*,* 2001.

Rzeszewski, Theodore S., *Digital Video: Concepts and Applications Across Industries*, IEEE Press, Piscataway, New Jersey, 1995.

Schank, R.C., "Active Learning Through Multimedia," *IEEE Multimedia*, Boston, Massachusettes, Spring 1994.

Sheldon, Bob and Wilansky, Ethan, *Microsoft Academic Learning Series: Windows 2000 Server*, Microsoft Press, Redmond, Washington, 2000.

Shi-Kuo, Chang, E. Hassanein and Chung-Yuan Hsieh, "A Multimedia Micro-University," *IEEE Multimedia*, Boston, Massachusettes, July-September 1998.

Sun, The Source for Java[TM] Technology, 2001. Information available at: http://java.sun.com/.
Task Force on Distance Education, "Report of the Task Force on Distance Education," The Pennsylvania State University, University Park, Pennsylvania, November 1993.

Tobagi, F.A., "Distance Learning with Digital Video," *IEEE Multimedia*, Boston, Massachusettes, Spring 1995.

Walsh, Aaron and Mikael Bourges-Sevenier, *Core Web3D,* Prentice Hall PTR, Upper Saddle River, New Jersey, 2001.

Weber, Joseph L., *Special Edition: Using Java 2 Platform*, Que Press, United States of America, 1999.

Woodbury, Marsha, "LEEP3-Distance Education Tips," *Proceedings of the 24th ACM SIGUCCS Conference on User Services*, Monterey, California, November 1997.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A - JMSTUDIO USER'S GUIDE

A.   INTRODUCTION

JMStudio is Sun Microsystems's exemplar program using the full capabilities of the Java Media Frameworks 2.0 API. The executable is available as a free download at http://java.sun.com/products/java-media/jmf/ . The web page also contains multiple links to other sample code and documentation. Additionally, the java code that was used to generate the application is available at this site.

B. JMSTUDIO

For this paper, the Windows Performance Pack was downloaded. After downloading and installing JMStudio, a shortcut is displayed upon the desktop. The application can also be started by the link established under Start > Programs > JavaMediaFrameworks 2.1.1 > JMStudio. Figure A.1 displays the application as it is first opened. Most default settings will work but the following information is provided for when there are multiple devices with multiple input ports.

Figure A-1:   JMStudio.

Once the application is displayed, three menu options are available. The Help option displays the version number and copyright license for JMStudio. The Player menu provides options for capture device controls. Under the video source button, the ports used for video capture and display can be set. For instance, if a video-capture device supports composite and s-video inputs for its video source, a user can select the source by using a few steps.

179

Open JMStudio:

    1.   Desktop Icon

    2.   Start > Programs > JavaMediaFrameworks 2.1.1 > JMStudio

Monitor Video and Audio:

    1. File > Capture > Select from input formats

Saving a file once JMStudio is monitoring:

    1. File > Export > Choose format > Choose filename and location

Table A-1:   JMStudio Multimedia Capture Summary.

First, the user needs to pause the player using the pause symbol in the bottom right corner. Once the player is paused, using the Player menu item and then capture device will open the capture device control dialog box as displayed in Figure A.2. Next, the Video source button needs to be depressed to expose the capture device's driver's properties box. From here, the user can pick their input source. Most low-end capture devices have only one video input port. However, the higher end cards may have three or more.
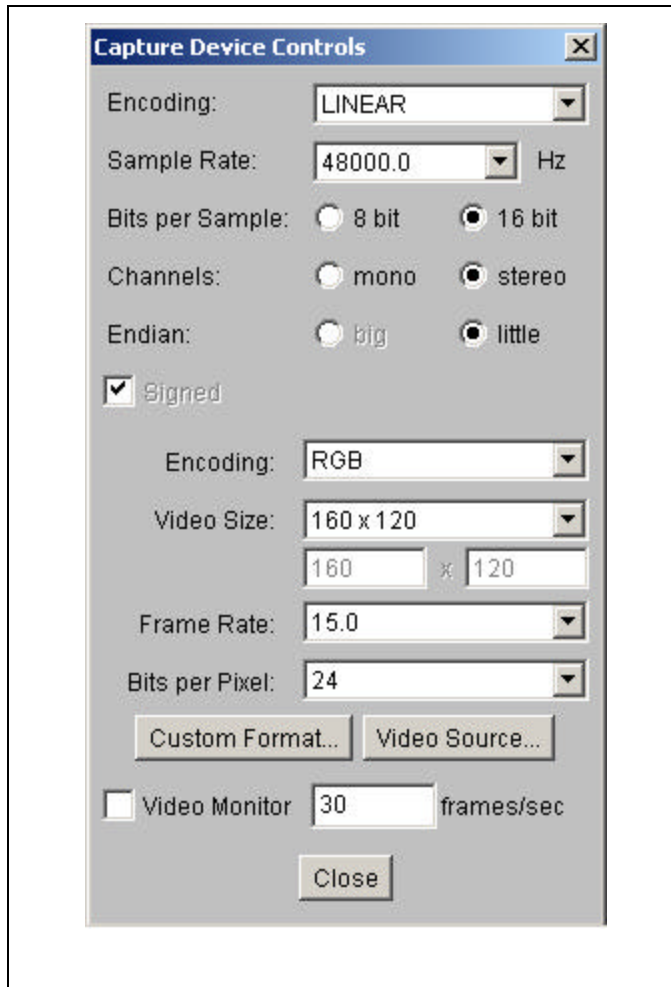
Figure A-2:   JMStudio Capture Device Controls.

Now, the player is configured to use the proper inputs for video. The user needs to insure that the proper audio inputs are configured as well. The JMStudio application reveals JavaSound and Window's DirectSound. If the card has its own sound capture, it will be revealed as well. For instance, the Winnov 1000 card has its own sound capture capability and can be selected from the capture devices dialog box. Some source of confusion can be created when the devices are designed to capture and playback using one set of devices and the computer is configured to use another set of devices as the default. For Windows Operating systems, the sound properties application can be opened and modified to use the correct audio ports and adjust their decibel levels.
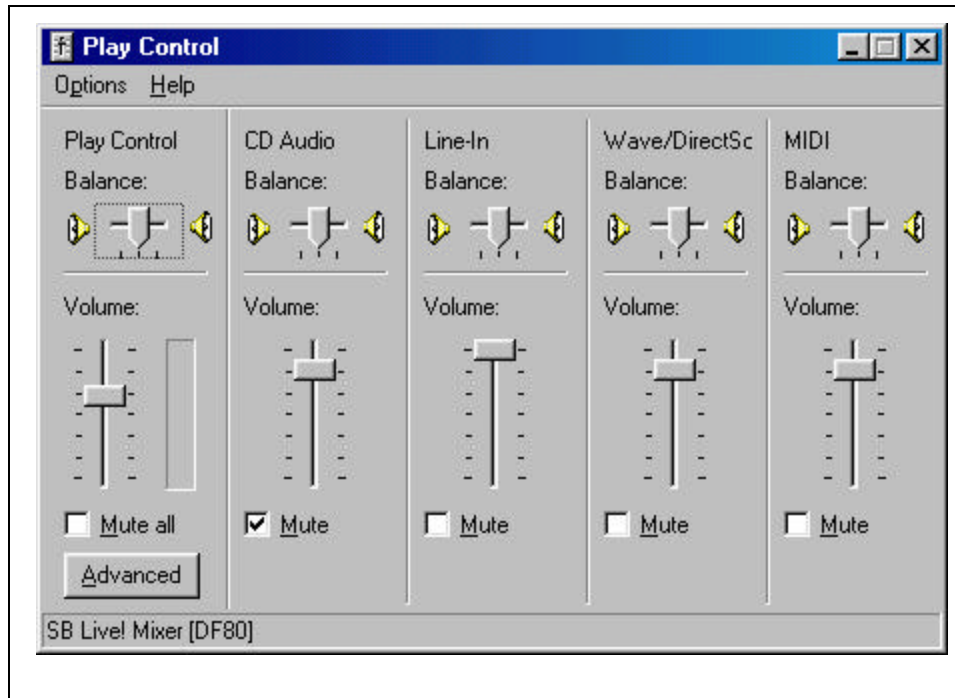
Figure A-3:   Windows Sound Control Application

Once JMStudio is configured correctly for capture, the File Menu item can be utilized to begin the capture process by pressing the capture option that is displayed in Figure A.4. The capture device dialog box will appear and allow the user to select

the capture device that they desire for their session. This dialog box displays both video and audio devices that are registered with JMStudio. Hence, if a capture device is not present, executing the preferences option under File will open a dialog box that allows the application to search for new devices. This option was used when a USB capture device was not seen by JMStudio until a video signal was actually transmitted through the capture device. The settings created in this area, are only for display in the player that will appear next.
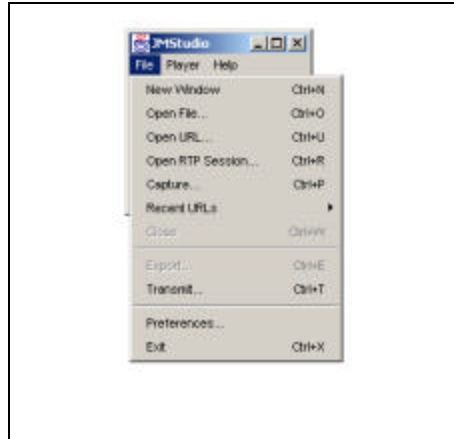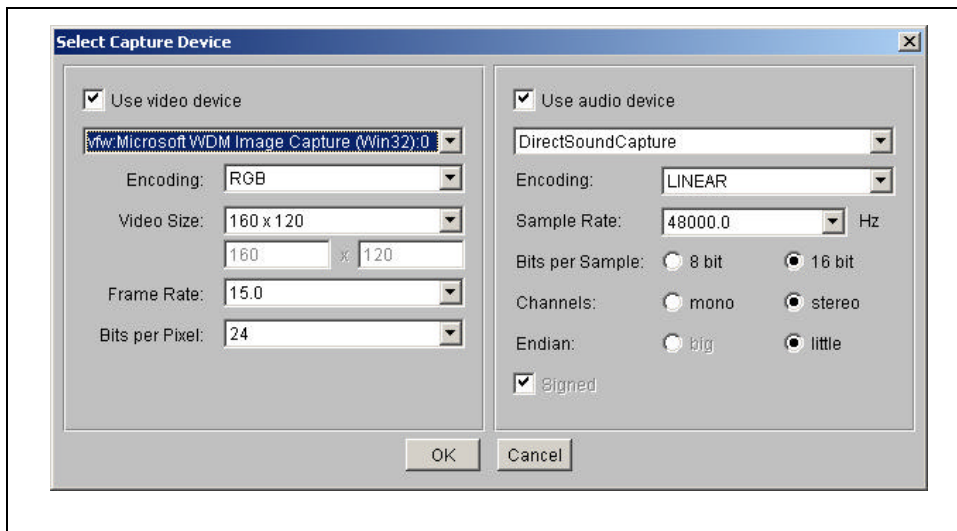
Figure A-4:   JMStudio File Menu

Figure A-5:   JMStudio Capture Device Dialog.

If all settings are correctly set and JMStudio is properly installed, the JMF player appears with picture and controls. From the player, another session can be started or the current session can be stopped. All of the capabilities of the previous instance of the application are available except for the creation of a second capture session. Only one audio and one video capture source can be initiated at one time. Even though the player is not actually saving the file, the capture devices' resources are now dedicated to the program.
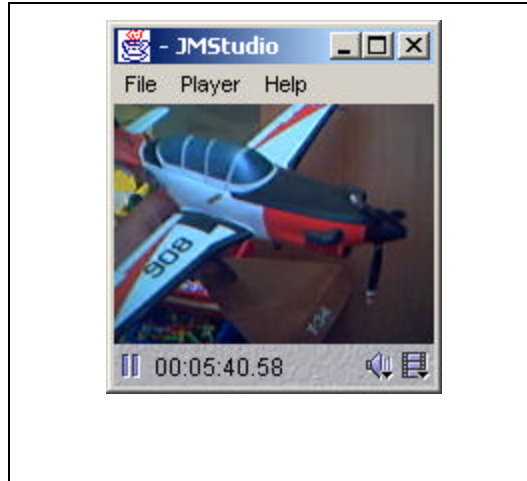
Figure A-6:   JMStudio Capture Display.

The next step in the capture process is to save the output of the capture device by clicking on the export option located under the File menu item.
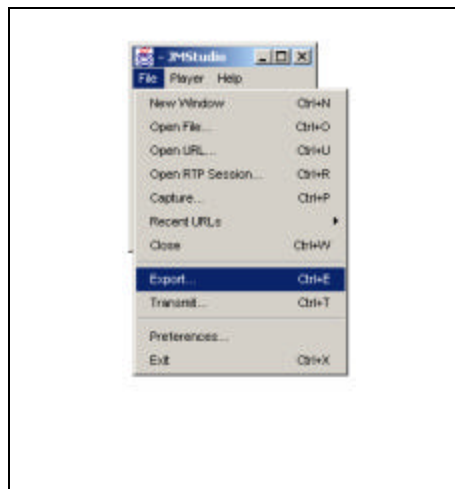

Figure A-7:   JMStudio Export Menu Item.

The export dialog box will open to allow the user to adjust video and audio settings for the saving of the capture data. File format such as Quicktime or GSM can be specified. Audio sample rates, encoding type, bits per sample and endian definition can be set as well.
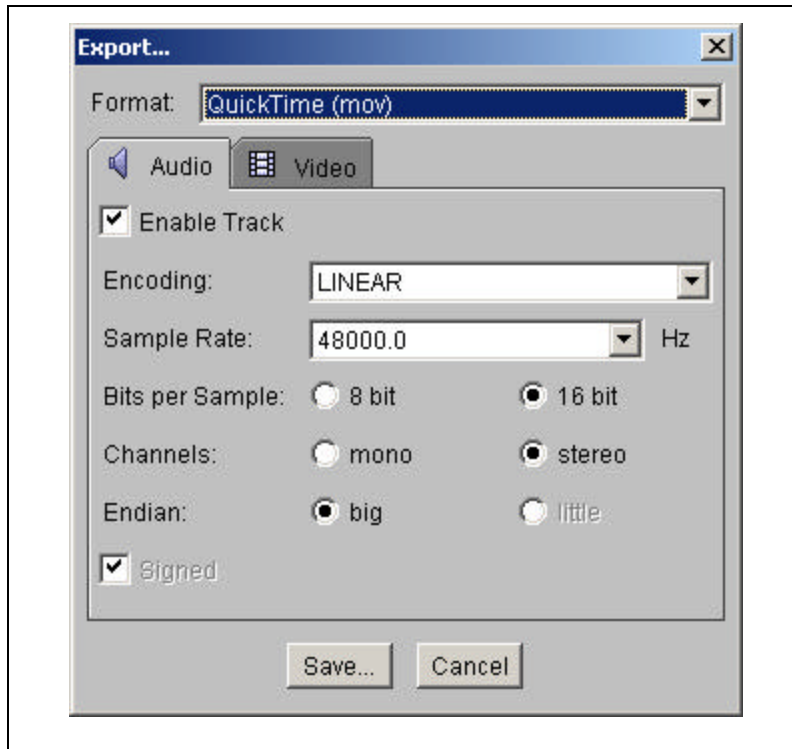
Figure A-8:   JMStudio Export Dialog Box Audio Tab.

The Video Tab has almost as many options. Video frame rate, dimension and encoding can be set under the video tab. These settings will all effect how the data presented from the capture device is recorded.
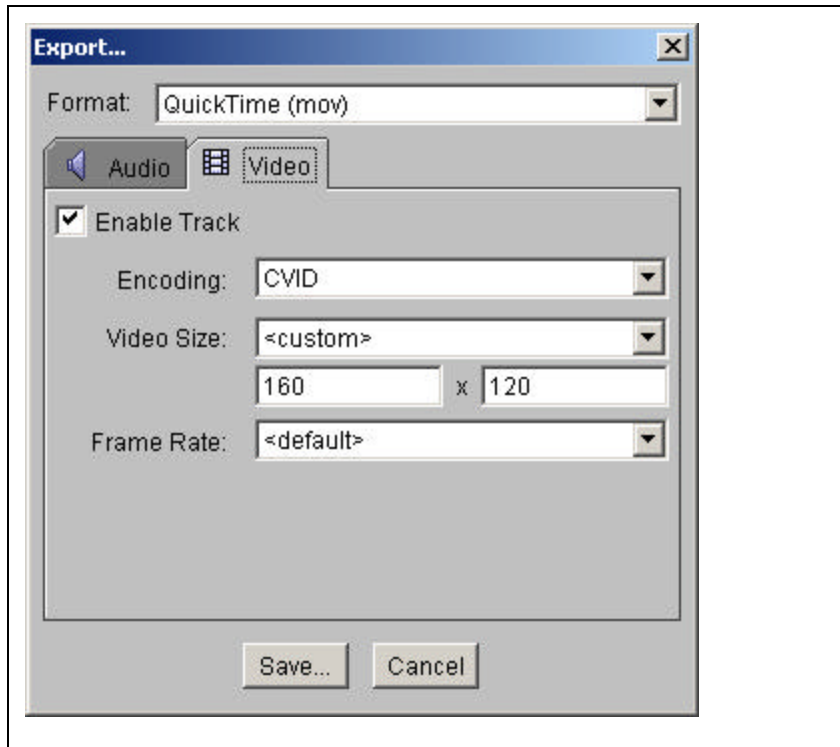
Figure A-9:   JMStudio Export Dialog Box Video Tab.

Once the application is tuned to the desired settings, the save button can be
pressed. A dialog box asks for the name of the file and location where it should be
located. After this information is provided, the saving file dialog box is displayed. By
default, there is no video monitor displayed. The pause button must be pressed to allow
the video monitor checkbox to be checked.

Figure A-10:   JMStudio Saving File Dialog Box.

Once the box is checked, the capture monitor is displayed. Even though the
image will move as data is presented, the file is not recording any of the information.

The resume button must be pressed so that the file can begin to record the data again. The stop button will halt the capturing of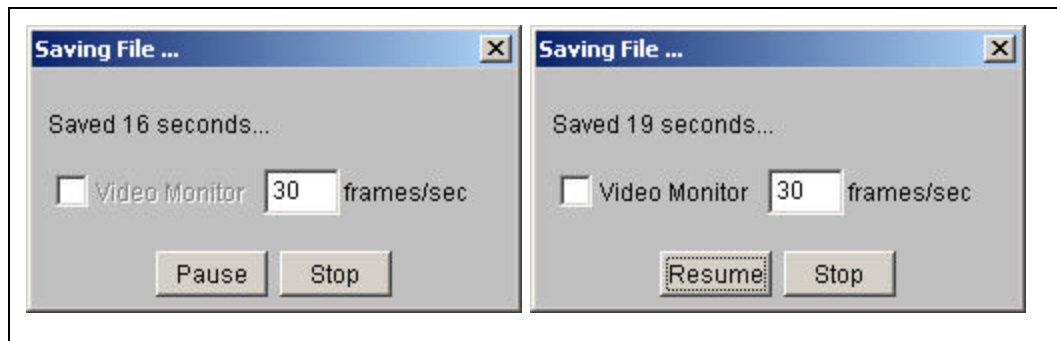 data. JMStudio will usually consume another ten to thirty seconds before it releases the resources that is has acquired for the capture process. If the program is ended before the resources are released, the file will most likely become corrupted and unusable.
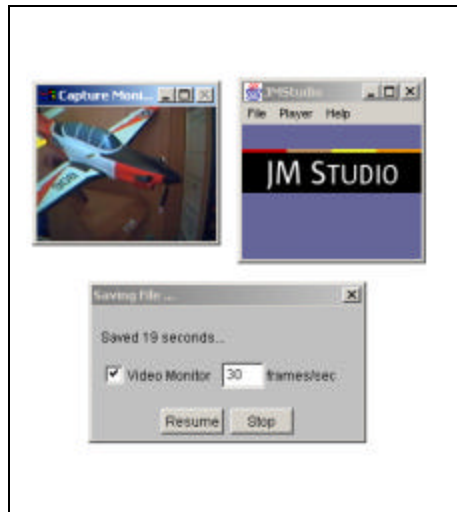
Figure A-11:   JMStudio Saving File Dialog Box with Capture Monitor.

Once the file is complete, it may be displayed using a compatible media player or JMStudio.   The player can display any format of which it can capture. The properties of the playing video can be displayed by pressing the movie reel symbol on the bottom right of the bottom task bar. Some media properties are displayed in Figure A.12.   The general information provides content type, duration, position with in the video, bit rate and frame rate. All of this information is useful for troubleshooting and monitoring video performance. For instance, if a file was supposed to be captured at fifteen frames per second but the media properties displays only five frames per second, the capture device may not have been able to capture at the desired rate.
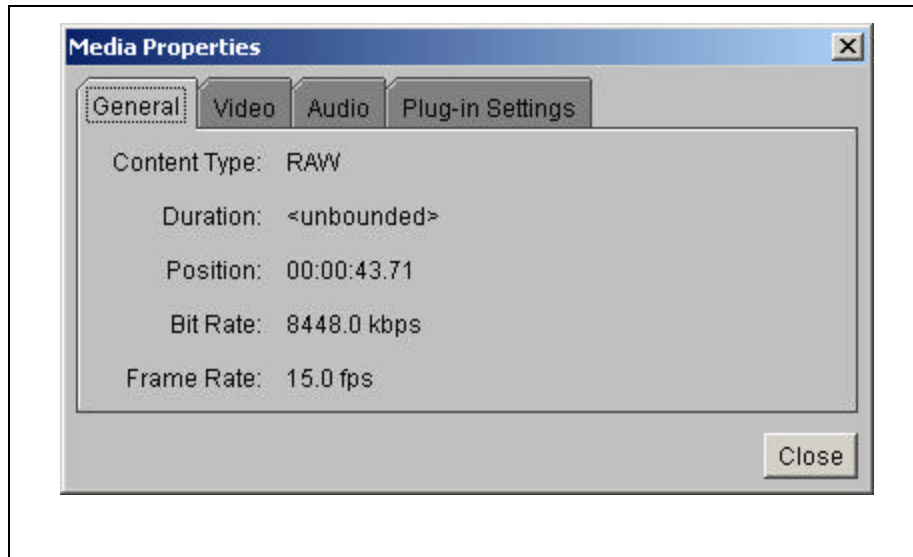
Figure A-12:   JMStudio Media Properties.

The plug in viewer can also be displayed from this location within the application. The viewer graphically displays the different software components that are being utilized in the display process. In the case of Figure A.13, the Microsoft's DirectSound Renderer and Graphics Driver Interface Renderer are receiving information from the video and audio buffer of JMStudio.
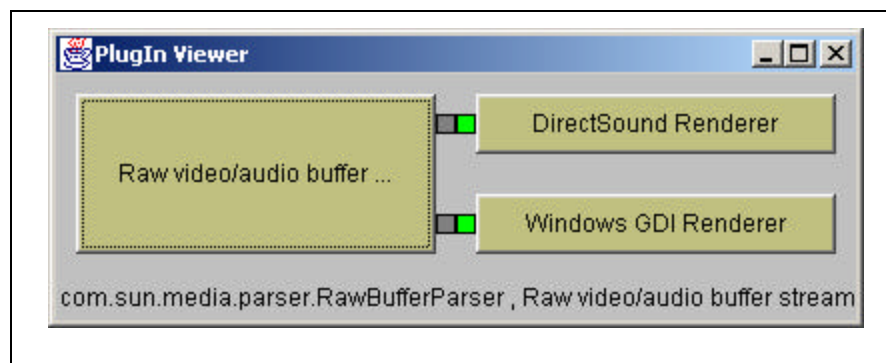
Figure A-13:   JMStudio PlugIn Viewer.

C.   SUMMARY

The JMStudio application demonstrates many of the functions available through the Java Media Framework (JMF) API. A person not familiar with Java can use the

program as they would use any application and gain the ability to monitor, capture, stream, save and transcode multimedia. Programmers familiar with Java can use the source code provided with JMStudio as a baseline for creating their own applications. Additionally, if a programmer's code did not work, the programmer can check to see if JMStudio had the same functionality and if it worked in the current environment. JMStudio is an excellent starting point for people desiring to learn about multimedia and Java's capabilities.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B – MBONE TOOLS

A.   INTRODUCTION

The MBone Tool Set consists of several applications capable of creating and connecting to multimedia sessions on the MBone portion of the Internet. The applications were created by University College London with several grants from DARPA, EPSRC and EC.   All tools are open-source and available in pre-compiled formats at http://www-mice.cs.ucl.ac.uk/multimedia/software/.

This section will detail the tools used for the relay of Advanced Physically Based Modeling during the Spring 2001 term at the Naval Postgraduate School in Monterey, CA. They were Session Directory (SDR), Video Conferencing (VIC) and Robust Audio Tool (RAT). Although available, White Board (WB) and Shared Text Editor (NTE) were not used since the instructor did not interact with the MBone tools. The following describes the tools that were utilized:

B.   SESSION DIRECTORY (SDR)

The first tool used in a standard session is the SDR. An initial advertisement needs to be made using the "New" button on the SDR's tool bar. Initial information is entered such as name of event, name of session contact, e-mail addresses and phone numbers. The event coordinator also decides on the range of the multicast by specifying the number of hops available using Time to Live (TTL). The event is also assigned an unique Multicast address and port number. The information is now retained in the SDR catalog allowing interested parties to connect to the same session. By clicking on the Public Session event, a second dialog box opens and displays more detailed session information. The information that was originally entered when the session was created is available. Also, there are buttons that provide access to the initial session establishment. The "Audio" button allows for the listen and talk settings.
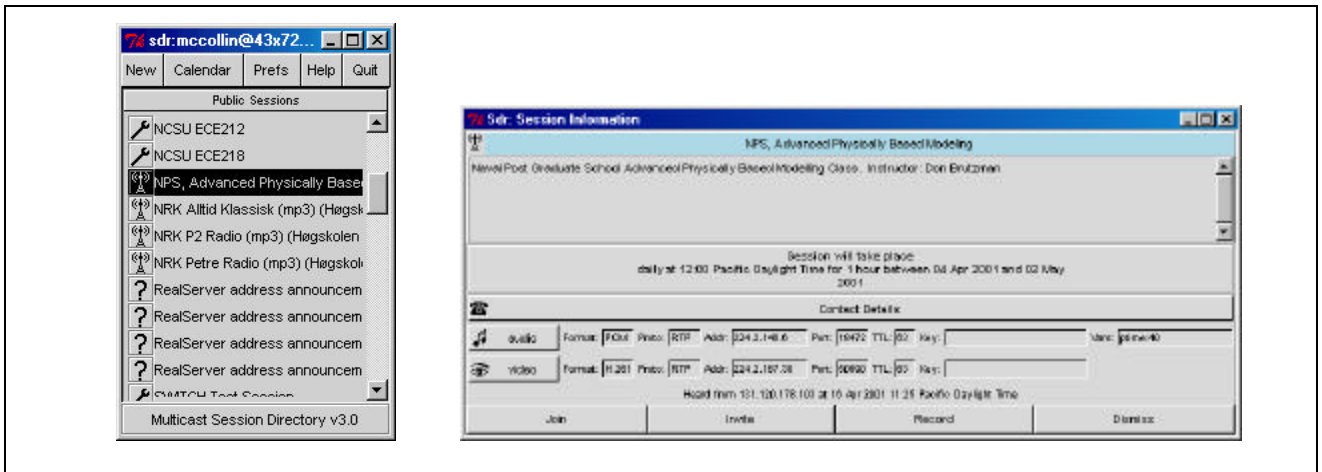
Figure B-1:   Session Directory (SDR).

Of special note is the sideways triangle followed by "Line In". By clicking on the triangle, the method of input can be modified. If the sound card, which is enabled on the system, has multiple inputs such as Aux, Microphone and Line In, the applet will give the user options to choose a particular channel for input.

C.   ROBUST AUDIO TOOL (RAT)

RAT is the tool utilized for audio communication and sound transference. Once a connection to a session is established, the audio tool can change the settings of talk or listen when the user checks or uncheck the appropriate box. The volume and gain can be adjusted for both channels. Also, as sound is received or sent, a sliding color scale is used to represent the levels of the signal's volume. Green is ideal while yellow is slightly degraded and red becomes almost incomprehensible. The reason for the degradation of the sound quality is that the sinusoidal shape of the frequency becomes squared off as it is pushed higher into the usable frequency range.
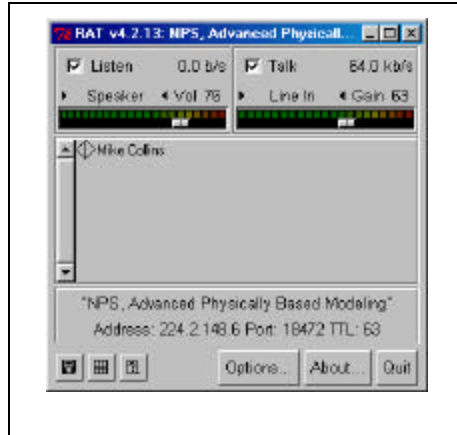
Figure B-2:   Robust Audio Tool (Rat).

D.   VIDEO CONFERENCING (VIC)

The video conferencing tool is utilized to adjust the properties of the conference session. This tool establishes the desired transmission rates and frames per second. The transmission rate is the limit in bandwidth allowed for transmission. If the data stream is less than the transmission rate, the data will be sent as it is originally created. If the data stream exceeds the transmission rate, frames are dropped to limit the amount of data sent from the application. Device choice and encoding format can also be selected by this application. Hence, if a machine is connected to multiple devices, the user can change the video or audio source to match the area of interest. If the devices have multiple ports for input, the different ports can be selected as well. By establishing these parameters, the user can customize the information sent from their station in the conference.
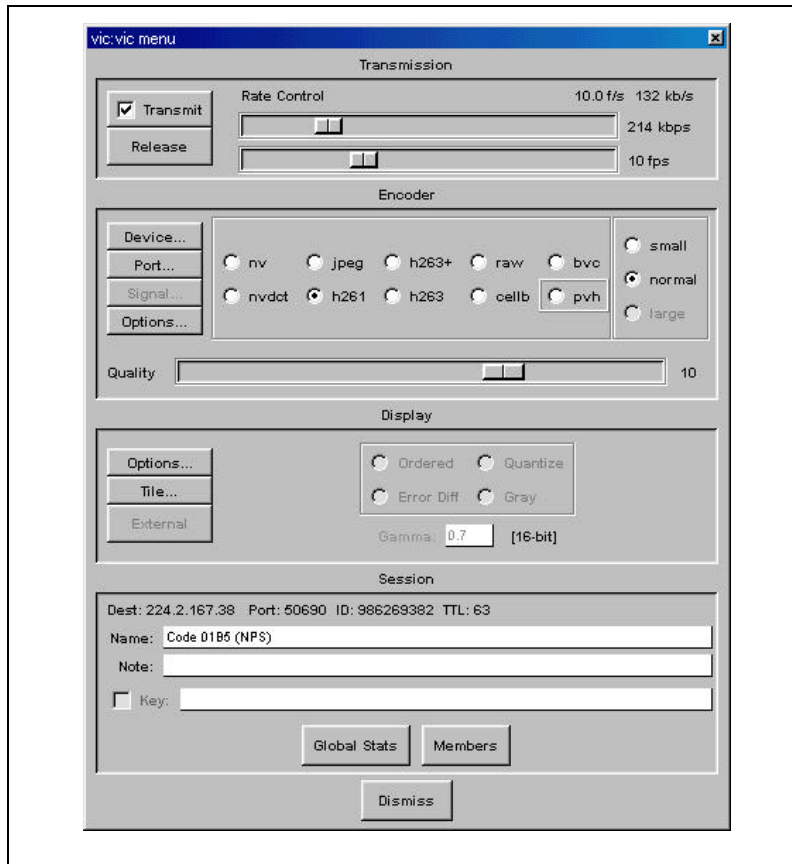
Figure B-3:   Video Conferencing (VIC).

E.   SUMMARY

The MBone toolset includes a rich suit of tools capable of transmitting and receiving multimedia sessions. During the experiments in the Video Teleconference Center, only SDR, RAT and VIC were explored. This appendix highlights techniques allowing these tools to provide distance education across the country and across the world. The use of these tools also provided instant feedback about connectivity to the Internet's multicast backbone.

APPENDIX C – VIDEO CAPTURE WITH ADOBE PREMIERE

A.   INTRODUCTION

Premiere is Adobe Systems Incorporated's video capture and editing tool. The current release is 6.0, which offers support for many video formats, web formats and cross platform capabilities. For instance, the current standard of DV is now supported by Premiere. Hence, a DV compatible camera, an IEEE 1394 cable and computer interface allow Adobe Premiere to capture non-degraded digital video directly to the hard drive. The Movie Capture window allows settings to be modified such as the capture source, comments about movies, movie names and batch capture. Web export plug-ins allows files to be cleaned and saved in either Microsoft's Advanced Windows Media or Advanced RealMedia formats. The export can convert and place these files directly on a Microsoft or RealMedia server. The clips can also have markers placed within their tracks to allow added functionality such as opening web pages.

Besides capture, Adobe Premiere contains an extensive array of editing tools for the clips. First, an audio mixer can control up to 99 audio tracks. Volume, gain, pan, fade and other effects are among the possible controls. The storyboard window allows extensive control over timing and synchronization. Enhanced monitor and timeline controls give precise control over joining multiple clips and how they transition from one scene to the next. Key frame and effect controls have also improved to allow the operator precise control over the final product. Finally, Premiere is designed to seamlessly integrate with Adobe's other products such as Illustrator, After Effects and Go Live.

B.   CAPTURE PROCESS

Adobe Premiere operates under the project paradigm. Hence, a project should be created before any other action is performed. A project can be created by choosing File > Open. Then, locate and select the file. Finally, click Open. It should be noted that the project directories only contain references to the original video clips. If the file is moved, Premier will display a dialog box presenting the user with options to search for the file, place a blank placeholder in its location, or skip (remove) the file from the project. Using

multiple disk drives as scratch disks may optimize the project. While most processing occurs in RAM, when the data becomes too large, it must be stored to a hard drive. Low space warnings on the hard drives may also be set.

While it is possible to undo mistakes, saving the project often with different versions is a recommended technique. Premier can undo up to 99 actions. However, once a project has been saved, a previous version is required to regress to a previous state.   Additionally, the File > Revert command will undo all changes since the project was first opened.

Once the project is set in the proper directories, multimedia is captured using video and audio capture cards. The cards are connected to camcorders or video decks and installed with the proper drivers. Adobe Premier is able to connect to many of these capture devices (www.adobe.com/premiere) and create digital content directly into a project. The user can adjust the Capture Format, Capture Video, Size, Rate (Video for Windows), Video, Audio, Advanced, VFW Settings, Capture Audio, Report Dropped Frames, Abort on Dropped Frames, Capture Limit, Preroll Time, Timecode Offset and Log Using Reel Name [Adobe Premier 6.0]. Additionally, Premier allows for batch capturing from a single source. The user sets the time limits for each clip and Premier generates clips from the continuous source at the predefined times.

- Specify the scratch disk for captured movies.
- Set up the video source with Project > Project Settings > Capture, click Video, choose Source, and choose a video source from the Digitizer menu.
- Carefully check other settings in the Capture panel.

Table C-1:   Adobe Premier video capture steps.

C.   BATCH PROCESSING

You can export multiple video programs or clips to multiple files automatically using Adobe Premier. The Batch Processing command uses the export settings and compression options you specify for each video file being created. The user can create

and save multiple batch lists for easy, repeatable exporting of groups of projects. Batch processing can save time and greatly simplify exporting several video files overnight, testing several different export settings files to find out which work best, creating versions for different media (such as videotape, CD, and Web delivery), and creating versions for different editing tasks (such as offline editing or rotoscoping). Table C-2 demonstrates the steps required to export a batch of files.

1. Choose Project > Utilities > Batch Processing.

2. Click Add. Select a project or video file that you want to add to the list, and click Open. Repeat for as many projects or video files as you want to export.

3. For each file in the list, select the project or clip, and click Target. Specify the location and filename of the file that will be produced from this project or clip, and click Save. Click Settings if you want to verify the project settings for a selected project or clip.

4. Make sure project source files are ready for processing by selecting any number of projects in the list and click Check. This is particularly important if you plan to leave the computer unattended, because missing files will stop all processing. Premiere opens each selected project in turn and verifies that the clip files exist on disk that correspond to the clip instances used in each selected project. If a clip file is missing, Premiere notifies the user with the Locate File Dialog box.

5. To process the entire batch list, click Make, select Make All Sources in List, and click OK. To process a continuous range of projects in the list, click to select the first project you want to process, hold down Shift as you click the last project you want to process, click Make, select Make Selected Sources Only, and click OK.

6. To process a discontinuous range of projects in the list, click to select the first project you want to process, hold down Control (Windows) or Command (Mac OS) as you click each.

7. To process additional projects, click Make, select Make Selected Projects Only, and click OK

Table C-2:   Adobe Premier Batch Processing Steps.

.

It is worth noting that, if the user clicks Cancel, they will lose all of the changes since he Batch Processing dialog box was opened. If the user decides not to make a movie but wants to retain batch list settings that were specified, the settings need to be saved as described in the Table C-3. At a later time, the user can open the Batch Processing dialog box and load the saved settings.

1. Open the Batch Processing window, if not already open.

2. Delete a project or clip from the Batch Processing list, select a project or clip in the list, and click Delete.

3. Save the Batch Processing list to disk by clicking on Save, specify a location and filename, and click Save.

4. Load a Batch Processing list from disk by clicking on Load, locate and select a batch-processing list file, and click Open.

Table C-3:   Adobe Premier steps to save batch-processing settings.

D.   SUMMARY

Adobe Premier is a professional grade application designed for the commercial market. While this appendix touches on some of the common functions utilized for Siggraph 2001, it is by no means comprehensive. Entire books have been written on the use of Premiere. Additionally, graphic artists may spend years learning the intricacies of Adobe's effects and filters. However, the application can still be used as a simple capture and batch-encoding tool.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D – TESTBUTTON.JAVA CODE

A.   INTRODUCTION

The TestButton class was developed as a framework for multiple, small programs developed for the final multimedia capture program. While TestButton was used to test multiple Swing configurations as well as unique system calls, it was never intended as a stand alone program. The functionality developed within this application was cut and pasted into the more complex programs. Section B contains the code used to launch an executable file located elsewhere in a Window's environment.

B.   TESTBUTTON.JAVA CODE

```
/*
* TestButton.java
* TestButton is an intermediate program that was designed to test the
*launching an executable from within a form.  By pressing the button that is
*generated by this code, jmstudio.exe is launched from a pre-staged location.
*If the program is not in the exact location as specified in the code, the
*program generates an error code, does not launch the program and continues to
*run.
*
* Created on May 25, 2001, 2:43 PM
*/


/**
*
* @author  Michael C. Collins
* @version 0.1
*/

public class TestButton extends javax.swing.JFrame {

    /** Creates new form TestButton */
    public TestButton() {
        initComponents ();
        pack ();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     */
    private void initComponents() {
        //GEN-BEGIN:initComponents
        jButton1 = new javax.swing.JButton();
        addWindowListener(new java.awt.event.WindowAdapter() {
            public void windowClosing(java.awt.event.WindowEvent evt) {
                exitForm(evt);
            }
        }
        );

        jButton1.setText("jButton1");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
                    jButton1ActionPerformed(evt);
            }
        }
        );

        getContentPane().add(jButton1, java.awt.BorderLayout.CENTER);

    }//GEN-END:initComponents

  private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
      //GEN-FIRST:event_jButton1ActionPerformed

      try{

          Runtime.getRuntime().exec(PROGRAM);
      }
      catch (Throwable t) {
          System.out.println("Unable to Load and Execute Program " + PROGRAM);
      }
  }//GEN-LAST:event_jButton1ActionPerformed

    /** Exit the Application */
    private void exitForm(java.awt.event.WindowEvent evt) {//GEN-
FIRST:event_exitForm
        System.exit (0);
    }//GEN-LAST:event_exitForm

    /**
     * @param args the command line arguments
     */
    public static void main (String args[]) {
        new TestButton ().show ();
    }


    // Variables declaration
    private javax.swing.JButton jButton1;
    final String PROGRAM = "C:/Program Files/JMF2.1.1_beta3/bin/jmstudio.exe";
    // End of variables declaration
}
```

C.   SUMMARY

This simple little program was used to launch JMStudio from its location within the Window's file structure. By giving a Java application the ability to launch executable files, previously unavailable programs behaved as if they were part of the new application. The user no longer needed to find the program and execute it. They only needed to click on a button and have the executable run.

APPENDIX E – RECORDBUTTON.JAVA CODE

A. INTRODUCTION

The RecordButton.java program was the final capture program developed using the Java Media Framework (JMF) API. The application could capture and render video and audio. The video and audio tracks were synchronized and stored in a Quicktime file. H.263 was chosen as the compression codec since it was well documented and provided excellent results.

B. RECORDBUTTON.JAVA CODE

```
/*
 * RecordButton.java
 *
 * Created on June 11, 2001, 3:43 PM
 */


import java.awt.*;            //needed for Components
//import java.awt.event.*;  //not needed since Forte used full library name
//import java.awt.font.*;   //but kept for completeness
//import javax.swing.*;
//import javax.swing.border.*;
import javax.media.*;
import javax.media.format.*;
import javax.media.protocol.*;
import java.io.*;
import javax.media.datasink.*;
import javax.media.control.MonitorControl;
/** RecordButton.java was created using Forte(tm) for Java(tm), Community
Edition.
 * The program was designed to capture and record the Siggraph 2001 papers,
 * panels and class presentations.
 *
 * @author Michael Collins
 * @version 0.2
 */
public class RecordButton extends javax.swing.JFrame implements
ControllerListener{

    /** Creates new form of RecordButton.
     * There is only one class for RecordButton.
     */
    public RecordButton() {
        initComponents ();
        pack ();

    }//end constructor

    /** This method is called from within the constructor to
     * initialize the form.
     */
    private void initComponents() {
        btnRecord = new javax.swing.JButton();
        btnStop = new javax.swing.JButton();
```

```java
jTextField1 = new javax.swing.JTextField();
lblStatus = new javax.swing.JLabel();
btnPlay = new javax.swing.JButton();
jPanel1 = new javax.swing.JPanel();
getContentPane().setLayout(new java.awt.GridBagLayout());
java.awt.GridBagConstraints gridBagConstraints2;
setTitle("Record Button");
addWindowListener(new java.awt.event.WindowAdapter() {
    public void windowClosing(java.awt.event.WindowEvent evt) {
        exitForm(evt);
    }
}
);

btnRecord.setText("Record");
btnRecord.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnRecordActionPerformed(evt);
    }
}
);

gridBagConstraints2 = new java.awt.GridBagConstraints();
gridBagConstraints2.gridx = 2;
gridBagConstraints2.gridy = 1;
gridBagConstraints2.fill = java.awt.GridBagConstraints.BOTH;
gridBagConstraints2.insets = new java.awt.Insets(20, 53, 0, 0);
getContentPane().add(btnRecord, gridBagConstraints2);


btnStop.setText("Stop");
btnStop.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        btnStopActionPerformed(evt);
    }
}
);

gridBagConstraints2 = new java.awt.GridBagConstraints();
gridBagConstraints2.gridx = 1;
gridBagConstraints2.gridy = 1;
gridBagConstraints2.fill = java.awt.GridBagConstraints.BOTH;
gridBagConstraints2.insets = new java.awt.Insets(20, 40, 0, 0);
getContentPane().add(btnStop, gridBagConstraints2);


jTextField1.setText("c:/SimpleCapture.mov");



gridBagConstraints2 = new java.awt.GridBagConstraints();
gridBagConstraints2.gridx = 1;
gridBagConstraints2.gridy = 0;
gridBagConstraints2.gridwidth = 4;
gridBagConstraints2.fill = java.awt.GridBagConstraints.BOTH;
gridBagConstraints2.ipadx = 152;
gridBagConstraints2.ipady = 10;
gridBagConstraints2.insets = new java.awt.Insets(50, 50, 0, 0);
getContentPane().add(jTextField1, gridBagConstraints2);


lblStatus.setText("Status: Stopped");
```

```java
        gridBagConstraints2 = new java.awt.GridBagConstraints();
        gridBagConstraints2.gridx = 2;
        gridBagConstraints2.gridy = 4;
        gridBagConstraints2.gridwidth = 2;
        gridBagConstraints2.fill = java.awt.GridBagConstraints.BOTH;
        gridBagConstraints2.ipadx = 33;
        gridBagConstraints2.ipady = 14;
        gridBagConstraints2.insets = new java.awt.Insets(3, 33, 0, 0);
        getContentPane().add(lblStatus, gridBagConstraints2);


        btnPlay.setText("Play");
        btnPlay.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnPlayActionPerformed(evt);
            }
        }
        );

        gridBagConstraints2 = new java.awt.GridBagConstraints();
        gridBagConstraints2.gridx = 4;
        gridBagConstraints2.gridy = 1;
        gridBagConstraints2.gridwidth = 2;
        gridBagConstraints2.fill = java.awt.GridBagConstraints.BOTH;
        gridBagConstraints2.insets = new java.awt.Insets(20, 30, 0, 0);
        getContentPane().add(btnPlay, gridBagConstraints2);


        jPanel1.setPreferredSize(new java.awt.Dimension(400, 400));
        jPanel1.setMinimumSize(new java.awt.Dimension(400, 400));
        jPanel1.setMaximumSize(new java.awt.Dimension(400, 400));

        gridBagConstraints2 = new java.awt.GridBagConstraints();
        gridBagConstraints2.gridx = 0;
        gridBagConstraints2.gridy = 6;
        gridBagConstraints2.gridwidth = 7;
        gridBagConstraints2.fill = java.awt.GridBagConstraints.BOTH;
        gridBagConstraints2.ipadx = -40;
        gridBagConstraints2.ipady = -130;
        gridBagConstraints2.insets = new java.awt.Insets(30, 180, 0, 0);
        getContentPane().add(jPanel1, gridBagConstraints2);

    }//end initComponents() function


    //action for when the play button is pushed
  private void btnPlayActionPerformed(java.awt.event.ActionEvent evt) {

      if(!recording && !playing){
          try {
              String filename = jTextField1.getText();
              player = Manager.createPlayer(new MediaLocator("file:///" +
filename));

              player.addControllerListener(this);
              player.start();
          }//end try
          catch (Exception e) {
              System.out.println(e.toString());
          }//end catch
           lblStatus.setText("Status: Playing");
          playing = true;//change the value of playing flag to true
      }//end if
```

```java
   }// end function btnPlayActionPerformed(java.awt.event.ActionEvent evt) play
button




   //action for when the stop button is pushed
   private void btnStopActionPerformed(java.awt.event.ActionEvent evt) {

       if (recording){
           lblStatus.setText("Status:Stopping");
           p.stop();
           p.close();
           filewriter.close();
           recording = false;//set value of recording flag to false
           lblStatus.setText("Status:Stopped");

           jPanel1.removeAll();

       }//end if

       if (playing) {

           player.stop();
           player.close();
           lblStatus.setText("Status:Stopped");
            jPanel1.removeAll();//why play games
           playing = false;//change the vlaue of playing flag to false
       }//end if

   }//end function btnStopActionPerformed(evt)     stop button




   //action for when record button is pushed
   private void btnRecordActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_btnRecordActionPerformed

       if(!recording && !playing){

           formats[0] = new AudioFormat(AudioFormat.IMA4);
           Dimension prefSize = new Dimension(160, 120);
           float prefFPS = 30.0f;
           String prefEncoding = "H263";

           VideoFormat prefFormat = new VideoFormat(prefEncoding,
           prefSize,
           Format.NOT_SPECIFIED, // length
           null, // data type
           prefFPS);


           formats[1] = prefFormat;
           FileTypeDescriptor outputType =
           new FileTypeDescriptor(FileTypeDescriptor.QUICKTIME);


           try {

               p = Manager.createRealizedProcessor(new ProcessorModel(formats,
               outputType));
           } catch (IOException e) {
               System.exit(-1);
```
206

```java
            } catch (NoProcessorException e) {
                System.exit(-1);
            } catch (CannotRealizeException e) {
                System.exit(-1);
            }//end try-catches
            // get the output of the processor
            DataSource source = p.getDataOutput();
            // create a File protocol MediaLocator with the location
            // of the file to
            // which bits are to be written
            MediaLocator dest = new MediaLocator("file://"
+(String)jTextField1.getText());

            try {


                filewriter = Manager.createDataSink(source, dest);
                filewriter.open();
            } catch (NoDataSinkException e) {
                System.exit(-1);
            } catch (IOException e) {
                System.exit(-1);
            } catch (SecurityException e) {
                System.exit(-1);
            }//end try-catches

            // start the filewriter
            try {
                filewriter.start();
            } catch (IOException e) {
                System.exit(-1);
            }//end try-catch

            //search for and create controls for monitoring of video and audio

            Control controls[] = p.getControls();
            Component monitorComp = null;

            for (int i = 0; i < controls.length; i++) {
                if (controls[i] instanceof MonitorControl) {
                    MonitorControl mc = (MonitorControl)controls[i];
                    monitorComp = mc.getControlComponent();

                    if (monitorComp != null){

                        jPanel1.add(monitorComp);//adding monitor of capture to
    bottom panel
                    }//end if


                    //commented out but will create a separate frame with video
    monitor in it.
                    //can be used if separate independent box is needed on top

                    //   mc.setEnabled(true);
                }//end if
            }//end for loop

            //start processor
            p.start();
            lblStatus.setText("Status: Recording");
            recording = true; //set the value of the recording flag to true
```

```
        }//end if

    }//end function btnRecordActionPerformed(evt)    record button

        /** Exit the Application */
        private void exitForm(java.awt.event.WindowEvent evt) {
            System.exit (0);
        }//end function exitForm(evt)



/** This class takes no arguments. It is designed to
 * run from the command line as "java RecordButton"
 * @param args No arguments necessary to run class
 */
        public static void main (String args[]) {
            new RecordButton ().show();

        }//end function main (String args[])

/** The controllerUpdate method is used for the Player player.  The player is
 * utilized to render videos from the file specified in the text area.
 * @param event Java Media Frameworks generates a multitude of events
 * that can be monitored by using an appropriate
 * addControllerListener method to a processor or player.
 * When an event is generated, JMF checks the controllerUpdate
 * method for status changes.
 * By using a series of if-then statements, a multitude
 * of conditions may be checked and conditions changed
 * accordingly.
 */
        public void controllerUpdate(javax.media.ControllerEvent event) {

            if (event instanceof RealizeCompleteEvent) {

                if ((comp = player.getVisualComponent()) != null)
                    jPanel1.add (comp);//end if


                if ((comp = player.getControlPanelComponent()) != null)
                    jPanel1.add(comp);//end if

                validate();//refresh swing components

            }//end if

        }//end function controllerUpdate(Controller event)



        // Variable declarations
        private javax.swing.JButton btnRecord;
        private javax.swing.JButton btnStop;
        private javax.swing.JTextField jTextField1;
        private javax.swing.JLabel lblStatus;
        private javax.swing.JButton btnPlay;
        private javax.swing.JPanel jPanel1;

        private Format formats[] = new Format[2];
        private Processor p = null;
        private Player player = null;
        private Component comp;
        private DataSink filewriter = null;
```

```
    private boolean recording = false;
    private boolean playing = false;
}
```

C.   SUMMARY

This application performed all of the required functions of digitizing and rendering multimedia. It was robust enough to withstand a production environment. However, several assumptions needed to be made. For instance, the program assumes the video capture card is compatible with JMF. Also, the capture card is assumed to have the capability of capturing in RGB format with dimensions of 320x240 and thirty frames a second. Any programmer desiring to improve this application would need to address these issues by querying the capture devices and displaying the available formats. JMF allows for this functionality but since the program was with these limitations in mind, there was no need to build in the capability.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX F – TRANSCODE.JAVA

A.   INTRODUCTION

The following code was the original program modified for the SigTrans.java program.   The modified code commented out the main method as well as any references to parsing command line arguments. The remaining code was used as the Transcode class where the doIt(…) method was given all required parameters. The required parameters were generated within the GUI component of SigTrans.java.

B.   TRANSCODE.JAVA CODE

```
/*
 * @(#)Transcode.java      1.6 01/03/13
 *
 * Copyright (c) 1999-2001 Sun Microsystems, Inc. All Rights Reserved.
 *
 * Sun grants you ("Licensee") a non-exclusive, royalty free, license to use,
 * modify and redistribute this software in source and binary code form,
 * provided that i) this copyright notice and license appear on all copies of
 * the software; and ii) Licensee does not utilize the software in a manner
 * which is disparaging to Sun.
 *
 * This software is provided "AS IS," without a warranty of any kind. ALL
 * EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY
 * IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR
 * NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SUN AND ITS LICENSORS SHALL NOT BE
 * LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING
 * OR DISTRIBUTING THE SOFTWARE OR ITS DERIVATIVES. IN NO EVENT WILL SUN OR ITS
 * LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT,
 * INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER
 * CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF
 * OR INABILITY TO USE SOFTWARE, EVEN IF SUN HAS BEEN ADVISED OF THE
 * POSSIBILITY OF SUCH DAMAGES.
 *
 * This software is not designed or intended for use in on-line control of
 * aircraft, air traffic, aircraft navigation or aircraft communications; or in
 * the design, construction, operation or maintenance of any nuclear
 * facility. Licensee represents and warrants that it will not use or
 * redistribute the Software for such purposes.
 */

import java.awt.*;
import java.util.Vector;
import java.io.File;
import javax.media.*;
import javax.media.control.TrackControl;
import javax.media.control.QualityControl;
import javax.media.Format;
import javax.media.format.*;
import javax.media.datasink.*;
import javax.media.protocol.*;
import javax.media.protocol.DataSource;
import java.io.IOException;
```

```java
import com.sun.media.format.WavAudioFormat;


/**
 * A sample program to transcode an input source to an output location
 * with different data formats.
 */
public class Transcode implements ControllerListener, DataSinkListener {

    /**
     * Given a source media locator, destination media locator and
     * an array of formats, this method will transcode the source to
     * the dest into the specified formats.
     */
    public boolean doIt(MediaLocator inML, MediaLocator outML, Format fmts[],
                int start, int end) {

        Processor p;

        try {
            System.err.println("- create processor for: " + inML);
            p = Manager.createProcessor(inML);
        } catch (Exception e) {
            System.err.println("Yikes!  Cannot create a processor from the given
url: " + e);
            return false;
        }

        p.addControllerListener(this);

        // Put the Processor into configured state.
        p.configure();
        if (!waitForState(p, p.Configured)) {
            System.err.println("Failed to configure the processor.");
            return false;
        }

        // Set the output content descriptor based on the media locator.
        setContentDescriptor(p, outML);

        // Program the tracks to the given output formats.
        if (!setTrackFormats(p, fmts))
            return false;

        // We are done with programming the processor.  Let's just
        // realize the it.
        p.realize();
        if (!waitForState(p, p.Realized)) {
            System.err.println("Failed to realize the processor.");
            return false;
        }

        // Set the JPEG quality to .5.
        setJPEGQuality(p, 0.5f);

        // Now, we'll need to create a DataSink.
        DataSink dsink;
        if ((dsink = createDataSink(p, outML)) == null) {
            System.err.println("Failed to create a DataSink for the given output
MediaLocator: " + outML);
            return false;
        }
```

212

```java
    dsink.addDataSinkListener(this);
    fileDone = false;

    // Set the start time if there's one set.
    if (start > 0)
        p.setMediaTime(new Time((double)start));

    // Set the stop time if there's one set.
    if (end > 0)
        p.setStopTime(new Time((double)end));

    System.err.println("start transcoding...");

    // OK, we can now start the actual transcoding.
    try {
        p.start();
        dsink.start();
    } catch (IOException e) {
        System.err.println("IO error during transcoding");
        return false;
    }

    // Wait for EndOfStream event.
    waitForFileDone();

    // Cleanup.
    try {
        dsink.close();
    } catch (Exception e) {}
    p.removeControllerListener(this);

    System.err.println("...done transcoding.");

    return true;
}


/**
 * Set the content descriptor based on the given output MediaLocator.
 */
void setContentDescriptor(Processor p, MediaLocator outML) {

    ContentDescriptor cd;

    // If the output file maps to a content type,
    // we'll try to set it on the processor.

    if ((cd = fileExtToCD(outML.getRemainder())) != null) {

        System.err.println("- set content descriptor to: " + cd);

        if ((p.setContentDescriptor(cd)) == null) {

            // The processor does not support the output content
            // type.  But we can set the content type to RAW and
            // see if any DataSink supports it.

            p.setContentDescriptor(new
            ContentDescriptor(ContentDescriptor.RAW));
        }
    }
}
```

```
    /**
     * Set the target transcode format on the processor.
     */
    boolean setTrackFormats(Processor p, Format fmts[]) {

        if (fmts.length == 0)
            return true;

        TrackControl tcs[];

        if ((tcs = p.getTrackControls()) == null) {
            // The processor does not support any track control.
            System.err.println("The Processor cannot transcode the tracks to the
given formats");
            return false;
        }

        for (int i = 0; i < fmts.length; i++) {

            System.err.println("- set track format to: " + fmts[i]);

            if (!setEachTrackFormat(p, tcs, fmts[i])) {
               System.err.println("Cannot transcode any track to: " + fmts[i]);
               return false;
            }
        }

        return true;
    }


    /**
     * We'll loop through the tracks and try to find a track
     * that can be converted to the given format.
     */
    boolean setEachTrackFormat(Processor p, TrackControl tcs[], Format fmt) {

        Format supported[];
        Format f;

        for (int i = 0; i < tcs.length; i++) {

            supported = tcs[i].getSupportedFormats();

            if (supported == null)
               continue;

            for (int j = 0; j < supported.length; j++) {

               if (fmt.matches(supported[j]) &&
                   (f = fmt.intersects(supported[j])) != null &&
                   tcs[i].setFormat(f) != null) {

                   // Success.
                   return true;
               }
            }
        }

        return false;
    }
```

```java
    /**
     * Setting the encoding quality to the specified value on the JPEG encoder.
     * 0.5 is a good default.
     */
    void setJPEGQuality(Player p, float val) {

        Control cs[] = p.getControls();
        QualityControl qc = null;
        VideoFormat jpegFmt = new VideoFormat(VideoFormat.JPEG);

        // Loop through the controls to find the Quality control for
        // the JPEG encoder.
        for (int i = 0; i < cs.length; i++) {

            if (cs[i] instanceof QualityControl &&
                cs[i] instanceof Owned) {
                Object owner = ((Owned)cs[i]).getOwner();

                // Check to see if the owner is a Codec.
                // Then check for the output format.
                if (owner instanceof Codec) {
                    Format fmts[] =
((Codec)owner).getSupportedOutputFormats(null);
                    for (int j = 0; j < fmts.length; j++) {
                        if (fmts[j].matches(jpegFmt)) {
                            qc = (QualityControl)cs[i];
                            qc.setQuality(val);
                            System.err.println("- Set quality to " +
                                    val + " on " + qc);
                            break;
                        }
                    }
                }
                if (qc != null)
                    break;
            }
        }
    }


    /**
     * Create the DataSink.
     */
    DataSink createDataSink(Processor p, MediaLocator outML) {

        DataSource ds;

        if ((ds = p.getDataOutput()) == null) {
            System.err.println("Something is really wrong: the processor does not
have an output DataSource");
            return null;
        }

        DataSink dsink;

        try {
            System.err.println("- create DataSink for: " + outML);
            dsink = Manager.createDataSink(ds, outML);
            dsink.open();
        } catch (Exception e) {
            System.err.println("Cannot create the DataSink: " + e);
            return null;
```

```
        }

    return dsink;
}


Object waitSync = new Object();
boolean stateTransitionOK = true;

/**
 * Block until the processor has transitioned to the given state.
 * Return false if the transition failed.
 */
boolean waitForState(Processor p, int state) {
    synchronized (waitSync) {
        try {
            while (p.getState() < state && stateTransitionOK)
                waitSync.wait();
        } catch (Exception e) {}
    }
    return stateTransitionOK;
}


/**
 * Controller Listener.
 */
public void controllerUpdate(ControllerEvent evt) {

    if (evt instanceof ConfigureCompleteEvent ||
        evt instanceof RealizeCompleteEvent ||
        evt instanceof PrefetchCompleteEvent) {
        synchronized (waitSync) {
            stateTransitionOK = true;
            waitSync.notifyAll();
        }
    } else if (evt instanceof ResourceUnavailableEvent) {
        synchronized (waitSync) {
            stateTransitionOK = false;
            waitSync.notifyAll();
        }
    } else if (evt instanceof EndOfMediaEvent) {
        evt.getSourceController().close();
    } else if (evt instanceof MediaTimeSetEvent) {
        System.err.println("- mediaTime set: " +
            ((MediaTimeSetEvent)evt).getMediaTime().getSeconds());
    } else if (evt instanceof StopAtTimeEvent) {
        System.err.println("- stop at time: " +
            ((StopAtTimeEvent)evt).getMediaTime().getSeconds());
        evt.getSourceController().close();
    }
}


Object waitFileSync = new Object();
boolean fileDone = false;
boolean fileSuccess = true;

/**
 * Block until file writing is done.
 */
boolean waitForFileDone() {
    System.err.print("  ");
```

```
    synchronized (waitFileSync) {
        try {
            while (!fileDone) {
                waitFileSync.wait(1000);
                System.err.print(".");
            }
        } catch (Exception e) {}
    }
    System.err.println("");
    return fileSuccess;
}


/**
 * Event handler for the file writer.
 */
public void dataSinkUpdate(DataSinkEvent evt) {

    if (evt instanceof EndOfStreamEvent) {
        synchronized (waitFileSync) {
            fileDone = true;
            waitFileSync.notifyAll();
        }
    } else if (evt instanceof DataSinkErrorEvent) {
        synchronized (waitFileSync) {
            fileDone = true;
            fileSuccess = false;
            waitFileSync.notifyAll();
        }
    }
}


/**
 * Convert a file name to a content type.  The extension is parsed
 * to determine the content type.
 */
ContentDescriptor fileExtToCD(String name) {

    String ext;
    int p;

    // Extract the file extension.
    if ((p = name.lastIndexOf('.')) < 0)
        return null;

    ext = (name.substring(p + 1)).toLowerCase();

    String type;

    // Use the MimeManager to get the mime type from the file extension.
    if ( ext.equals("mp3")) {
        type = FileTypeDescriptor.MPEG_AUDIO;
    } else {
        if ((type = com.sun.media.MimeManager.getMimeType(ext)) == null)
            return null;
        type = ContentDescriptor.mimeTypeToPackageName(type);
    }

    return new FileTypeDescriptor(type);
}
```

```java
/**
 * Main program

public static void main(String [] args) {

    String inputURL = null, outputURL = null;
    int mediaStart = -1, mediaEnd = -1;
    Vector audFmt = new Vector(), vidFmt = new Vector();

    if (args.length == 0)
        prUsage();

    // Parse the arguments.
    int i = 0;
    while (i < args.length) {

        if (args[i].equals("-v")) {
            i++;
            if (i >= args.length)
                prUsage();
            vidFmt.addElement(args[i]);
        } else if (args[i].equals("-a")) {
            i++;
            if (i >= args.length)
                prUsage();
            audFmt.addElement(args[i]);
        } else if (args[i].equals("-o")) {
            i++;
            if (i >= args.length)
                prUsage();
            outputURL = args[i];
        } else if (args[i].equals("-s")) {
            i++;
            if (i >= args.length)
                prUsage();
            Integer integer = Integer.valueOf(args[i]);
            if (integer != null)
                mediaStart = integer.intValue();
        } else if (args[i].equals("-e")) {
            i++;
            if (i >= args.length)
                prUsage();
            Integer integer = Integer.valueOf(args[i]);
            if (integer != null)
                mediaEnd = integer.intValue();
        } else {
            inputURL = args[i];
        }
        i++;
    }

    if (inputURL == null) {
        System.err.println("No input url is specified");
        prUsage();
    }

    if (outputURL == null) {
        System.err.println("No output url is specified");
        prUsage();
    }

    int j = 0;
    Format fmts[] = new Format[audFmt.size() + vidFmt.size()];
```

218

```
    Format fmt;

    // Parse the audio format spec. into real AudioFormat's.
    for (i = 0; i < audFmt.size(); i++) {

        if ((fmt = parseAudioFormat((String)audFmt.elementAt(i))) == null) {
            System.err.println("Invalid audio format specification: " +
                                      (String)audFmt.elementAt(i));
            prUsage();
        }
        fmts[j++] = fmt;
    }

    // Parse the video format spec. into real VideoFormat's.
    for (i = 0; i < vidFmt.size(); i++) {

        if ((fmt = parseVideoFormat((String)vidFmt.elementAt(i))) == null) {
            System.err.println("Invalid video format specification: " +
                                      (String)vidFmt.elementAt(i));
            prUsage();
        }
        fmts[j++] = fmt;
    }

    // Generate the input and output media locators.
    MediaLocator iml, oml;

    if ((iml = createMediaLocator(inputURL)) == null) {
        System.err.println("Cannot build media locator from: " + inputURL);
        System.exit(0);
    }

    if ((oml = createMediaLocator(outputURL)) == null) {
        System.err.println("Cannot build media locator from: " + outputURL);
        System.exit(0);
    }

    // Trancode with the specified parameters.
    Transcode transcode  = new Transcode();

    if (!transcode.doIt(iml, oml, fmts, mediaStart, mediaEnd)) {
        System.err.println("Transcoding failed");
    }

    System.exit(0);
}


/**
 * Create a media locator from the given string.
 */
static MediaLocator createMediaLocator(String url) {

    MediaLocator ml;

    if (url.indexOf(":") > 0 && (ml = new MediaLocator(url)) != null)
        return ml;

    if (url.startsWith(File.separator)) {
        if ((ml = new MediaLocator("file:" + url)) != null)
            return ml;
    } else {
```

```
            String file = "file:" + System.getProperty("user.dir") +
File.separator + url;
            if ((ml = new MediaLocator(file)) != null)
                return ml;
        }

        return null;
    }


    /**
     * Parse the audio format specifier and generate an AudioFormat.
     * A valid audio format specifier is of the form:
     * [encoding]:[rate]:[sizeInBits]:[channels]:[big|little]:[signed|unsigned]
     */
    static Format parseAudioFormat(String fmtStr) {

        int rate, bits, channels, endian, signed;

        String encodeStr = null, rateStr = null,
               bitsStr = null, channelsStr = null,
               endianStr = null, signedStr = null;

        // Parser the media locator to extract the requested format.

        if (fmtStr != null && fmtStr.length() > 0) {
            while (fmtStr.length() > 1 && fmtStr.charAt(0) == ':')
                fmtStr = fmtStr.substring(1);

            // Now see if there's a encode rate specified.
            int off = fmtStr.indexOf(':');
            if (off == -1) {
              if (!fmtStr.equals(""))
                    encodeStr = fmtStr;
            } else {
              encodeStr = fmtStr.substring(0, off);
              fmtStr = fmtStr.substring(off + 1);
              // Now see if there's a sample rate specified
              off = fmtStr.indexOf(':');
              if (off == -1) {
                  if (!fmtStr.equals(""))
                    rateStr = fmtStr;
              } else {
                  rateStr = fmtStr.substring(0, off);
                  fmtStr = fmtStr.substring(off + 1);
                  // Now see if there's a size specified
                  off = fmtStr.indexOf(':');
                  if (off == -1) {
                     if (!fmtStr.equals(""))
                         bitsStr = fmtStr;
                  } else {
                     bitsStr = fmtStr.substring(0, off);
                     fmtStr = fmtStr.substring(off + 1);
                     // Now see if there's channels specified.
                     off = fmtStr.indexOf(':');
                     if (off == -1) {
                         if (!fmtStr.equals(""))
                             channelsStr = fmtStr;
                     } else {
                         channelsStr = fmtStr.substring(0, off);
                         fmtStr = fmtStr.substring(off + 1);
                         // Now see if there's endian specified.
                         off = fmtStr.indexOf(':');
```

220

```java
                    if (off == -1) {
                        if (!fmtStr.equals(""))
                            endianStr = fmtStr.substring(off + 1);
                    } else {
                        endianStr = fmtStr.substring(0, off);
                        if (!fmtStr.equals(""))
                            signedStr = fmtStr.substring(off + 1);
                    }
                }
            }
        }
    }
}

// Sample Rate
rate = AudioFormat.NOT_SPECIFIED;
if (rateStr != null) {
    try {
        Integer integer = Integer.valueOf(rateStr);
        if (integer != null)
            rate = integer.intValue();
    } catch (Throwable t) { }
}

// Sample Size
bits = AudioFormat.NOT_SPECIFIED;
if (bitsStr != null) {
    try {
        Integer integer = Integer.valueOf(bitsStr);
        if (integer != null)
            bits = integer.intValue();
    } catch (Throwable t) { }
}

// # of channels
channels = AudioFormat.NOT_SPECIFIED;
if (channelsStr != null) {
    try {
        Integer integer = Integer.valueOf(channelsStr);
        if (integer != null)
            channels = integer.intValue();
    } catch (Throwable t) { }
}

// Endian
endian = AudioFormat.NOT_SPECIFIED;
if (endianStr != null) {
    if (endianStr.equalsIgnoreCase("big"))
        endian = AudioFormat.BIG_ENDIAN;
    else if (endianStr.equalsIgnoreCase("little"))
        endian = AudioFormat.LITTLE_ENDIAN;
}

// Signed
signed = AudioFormat.NOT_SPECIFIED;
if (signedStr != null) {
    if (signedStr.equalsIgnoreCase("signed"))
        signed = AudioFormat.SIGNED;
    else if (signedStr.equalsIgnoreCase("unsigned"))
        signed = AudioFormat.UNSIGNED;
}

return new AudioFormat(encodeStr, rate, bits, channels, endian, signed);
```

```
    }


    /**
     * Parse the video format specifier and generate an VideoFormat.
     * A valid video format specifier is of the form:
     * [encoding]:[widthXheight]
     */
    static Format parseVideoFormat(String fmtStr) {

        String encodeStr = null, sizeStr = null;

        // Parser the media locator to extract the requested format.

        if (fmtStr != null && fmtStr.length() > 0) {
            while (fmtStr.length() > 1 && fmtStr.charAt(0) == ':')
                fmtStr = fmtStr.substring(1);

            // Now see if there's a encode rate specified.
            int off = fmtStr.indexOf(':');
            if (off == -1) {
                if (!fmtStr.equals(""))
                    encodeStr = fmtStr;
            } else {
                encodeStr = fmtStr.substring(0, off);
                sizeStr = fmtStr.substring(off + 1);
            }
        }

        if (encodeStr == null || encodeStr.equals(""))
            prUsage();

        if (sizeStr == null)
            return new VideoFormat(encodeStr);

        int width = 320, height = 240;

        int off = sizeStr.indexOf('X');
        if (off == -1)
            off = sizeStr.indexOf('x');

        if (off == -1) {
            System.err.println("Video dimension is not correctly specified: " +
sizeStr);
            prUsage();
        } else {
            String widthStr = sizeStr.substring(0, off);
            String heightStr = sizeStr.substring(off + 1);

            try {
                Integer integer = Integer.valueOf(widthStr);
                if (integer != null)
                    width = integer.intValue();
                integer = Integer.valueOf(heightStr);
                if (integer != null)
                    height = integer.intValue();
            } catch (Throwable t) {
                prUsage();
            }

            return new VideoFormat(encodeStr,
                        new Dimension(width, height),
                        VideoFormat.NOT_SPECIFIED, // maxDataLen
```

```
                              null,                    // data class
                              VideoFormat.NOT_SPECIFIED  // FrameRate
                              );
        }

        return null;
    }


    static void prUsage() {
        System.err.println("Usage: java Transcode -o <output> -a <audio format> -
v <video format> -s <start time> -e <end time> <input>");
        System.err.println("     <output>: input URL or file name");
        System.err.println("     <input>: output URL or file name");
        System.err.println("     <audio format>:
[encoding]:[rate]:[sizeInBits]:[channels]:[big|little]:[signed|unsigned]");
        System.err.println("     <video format>: [encoding]:[widthXheight]");
        System.err.println("     <start time>: start time in seconds");
        System.err.println("     <end time>: end time in seconds");
        System.exit(0);
    }
}
```

C. SUMMARY

The Transcode.java program was instrumental in the rapid development of a batch encoder. With code reuse from this application, SigTrans.java, the GUI batch encoder was developed within a few hours.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX G – STREAMING SERVER HARDWARE COMPONENTS

A.   INTRODUCTION

Due to the high cost of purchasing a high-end server from manufacturers such as Compaq and Dell, a decision was made to build the Siggraph servers from individually purchased components. The final cost was marginally more than $3000 while a server with the exact specifications cost $9000. Since three servers can be built for the price of one commercially assembled, parts and suppliers were identified for the construction of the Siggraph machines.

B.   SPECIFICATION

The original specification developed within this appendix was for three multimedia CAVE Artificial Virtual Environment (CAVE) computers. As such, high-end video and sound cards were included in the specification. While this requirement was not necessary for the purpose of streaming media, at a minimum, a low-end sound and video card should be included in the completed server. There are many instances where video and sound on the streaming server are a quick remedy to troubleshooting poorly created content.

An IBM clone computer has several key components that may be purchased individually from a multitude of suppliers. Although many manufacturers make multiple claims to the superiority of their product, most products are becoming commodity items that are only separated by packaging, extra software and technical support.

The components required to create a functioning computer begin with a motherboard. The motherboard limits the form factor, number and speed of the Central Processing Unit (CPU). Intel and AMD manufacture the CPUs commercially available for this type of computer under names such as Pentium III, Pentium 4, Athalon, Duron and Thunderbird. Due to the limited number of multiprocessor motherboards for AMD chips and the inability of Pentium 4 chips to operate in a multiprocessor configuration, One Ghz Pentium III chips were selected.

The motherboard also limits the type of memory that can be utilized. Memory is currently commercially available as PC2400 DDR, PC2100 DDR, PC1600DDR, PC133,

PC100, and RDRAM. The Intel chips are designed to work with PC133, PC100 and RDRAM. RDRAM, operating at 800MHZ, has a burst transfer maximum of 1.6 Gigabytes per second while PC133 has a maximum burst transfer of 1.2 Gigabytes per second. The motherboard selected for the server only supported PC133 RAM. It also supported a maximum of 2GigaBytes.

In order to allow the motherboard, CPU and Ram to operate, a power supply is necessary. The current form factor supports ATX power supplies. It is possible to purchase power supplies individually, but it is cost effective to purchase it with a case to store the computer components. However, some rendering farms have opted to not include cased for their individual machines. They have opted to screw the minimum necessary components to plywood and connect each system through a local area network. In order to allow portability and familiarity, a server case was purchased.

Hard drives are necessary for persistent storage of information. In order to maximize speed, the motherboard selected contained hardware Raid Level 1 controller to allow stripping across four Ultra 100 disks. After calculating the required disk space for Siggraph 2001, four eighty Gigabyte disks were purchased for a total of 320 Gigabytes of storage.

In order to monitor and control the system, video card, sound card, keyboard, mouse, speakers and monitor were ordered. While the individual brands and specifications were named for this system, a streaming server does not need any of these parts once it is configured and in a running state. Other components that were used to install and configure the computer were a CDROM drive, 3.5 inch magnetic floppy drive and an Iomega Zip drive. With a network connection and a pre-set installation method, these devices are not necessary. However, they also provided a secondary means of transferring programs and data to and from the system.

Finally, the network connection is extremely important for a streaming media server. In order to allow for future network upgrades, a D-Link Copper 10/100/1000 Mb/s Ethernet card was selected. The network interface card selection must match the interface and media of the network it will connect to. For instance, if the network was fiber optic, a Stick and Turn (ST) or Stick and Click (SC) optical interface is necessary for the server. Also, if the network is 100 Mb/s Ethernet a 10Mb/s Ethernet interface

may not work. Table G-1 contains the components specified for the streaming media server used for the Fred Brook's and Siggraph 2001 case studies.

Base Computer:
1.   One: Abit VP6 Motherboard with Ultra 100 Raid Controller
2.   Two: Pentium III One Ghz Processors
3.   Four: Generic 512 MB PC133 SDRAM
4.   Four: Maxtor Ultra 100 hard disks
5.   One: Supermicron full server case with 300 watt power supply

I/O:
6.   One: Microsoft Internet keyboard and mouse combination
7.   One: Samsung 3.5 Inch 1.44 Floppy Drive
8.   One: Iomega 250 MB internal zip disk
9.   One: Creative Labs Annihilator video card
10. One: Creative Labs Sound Blaster Live!Platinum 5.1 sound card
11. One: Creative Labs PCWorks 5 piece surround speaker system.
12. One: Yamaha 16x10x40 internal CD-RW drive
13. One: Acer 56x internal CD-ROM drive

Network:
14. One: D-link DGE-500T Gigabit LAN adapter

Miscellaneous:
15. Four: Generic Y-power splitter cables
16. Two: Vaster 30 Inch Ultra 100 cables
17. Four: Generic removable hard drive kits

Table G-1:   Siggraph 2001 Server Hardware Components.

C.   SUMMARY

Unfortunately, there was not a single vendor selling all of the components for the best price. Vendors stocking all of the components were normally the most expensive. Many vendors carried multiple components but appeared to specialize in a small arena to maintain the ability to sell hardware at substantially lower prices. This atmosphere lead to the use of several vendors with multiple policies and differing quality of merchandize. Since the system required the integration of several components to function properly, non-working components delayed the assembly of the machines while the merchandise

was replaced. A conclusion from this project was that, for a large and bureaucratic organization, purchasing pre-assembled machines and then modifying components specific to the project may result in quicker utilization of the machines.

APPENDIX H –SIGGRAPH 2001 CAPTURED PRESENTATIONS

A.   INTRODUCTION

The courses, papers, panels, special sessions and additional content captured for Siggraph 2001 amounted to over 250 hours of captured multimedia. As the presentations were captured, they were placed into a rigidly named file structure so they can easily be incorporated into the final web site for delivery. While additional content was captured after the event using videotapes created during the presentations, this appendix contains the directory structure originally developed for Siggraph 2001.

B.   DIRECTORY STRUCTURE

Courses:

Courses/03_PerformanceOpenGl

Courses/xx_FundamentalsOf3dGraphics

Courses/18_TensorDiagrams

Courses/28_IntroductionTo3dGraphics

Courses/26_Internetworked3D

Courses/53_GeometricAlgebra

Courses/38_GlobalIlluminationPhotonMaps

Courses/41_HowToPresentAtSIGGRAPH

Emerging Technologies Papers:

Papers/01_NaturalAnimation

Papers/02_VolumetricAndGraphingTechniques

Papers/03_Reality-BasedModeling

Papers/04_HardwareAndHardwareRendering

Papers/05_Meshes

Papers/06_MeasurementAndPresentation

Panels/13_BeyondCopyright-BraveNewWorldOfDigitalRightsManagement

Panels/14_NewtonsNightmare-RealityMeetsFauxPhysics

Panels/15_ImmersedInAnxietyOrProcessToHealing-VrMeetsMentalHealth

Panels/16_GameStories-SimulationNarrativeAddiction

Panels/17_InternetAppliances-UbiquitousHorizonsForTheWeb

Panels/18_ArtInSpace-WhatFor

Special Sessions:

SpecialSessions/MastersOfTheGame

SpecialSessions/Web3dRoundup

SpecialSessions/SiliconSenses

SpecialSessions/2001in2001

Additional Content:

ArtGalleryN-Space/exhibitNames/*.*

CreativeApplicationsLab/exhibitNames/*.*

EducatorsProgram/exhibitNames/*.*

SiggraphTV/exhibitNames/*.*

SketchesAndApplications/exhibitNames/*.*

TheStudio/exhibitNames/*.*

Conference:

/Conference/*.*

/GraphicsNet/*.*

/InternationalResources*.*

/Pathfinders*.*

/SpeakerPrep*.*

/StudentVolunteers*.*

C. SUMMARY

The strong naming of files and directories was required for Siggraph organizational challenges. With locations to store all multimedia files, volunteers did not misplace any of their work.. However, even with training some volunteers still managed to misplace several files due to the rush and heavy workload of Siggraph.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center..2
    8725 John J. Kingman Road, Suite 0944
    Ft. Belvoir, VA 22060-6218

2. Dudley Knox Library..2
    Naval Postgraduate School
    411 Dyer Road
    Monterey, CA 93943-5101

3. Mr. Arison..1
    JCS/J6
    Pentagon
    Washington, D.C. 20350-2000

4. Dr. Michael P. Bailey..1
    Technical Director, Marine Corps Training and Education Command
    Commanding General
    Marine Corps Combat Development Command, Code 46T
    3300 Russell Road
    Quantico, VA 22134

5. Dr. Philip S. Barry and Dr. Nil Zimmaran..1
    Chief, S&T Initiatives Division
    Defense Modeling and Simulation Office
    1901 N. Beauregard Street, Suite 500
    Alexandria VA 22311

6. Miguel Encarnagao, Jr. 1
    Fraunhofer Center for Research in Computer Graphics (CRCG)
    321 South Main St
    Providence, RI 02903

7. Curtis Blais..1
    Institute for Joint Warfare Analysis
    Naval Postgraduate School
    Monterey, CA 93940-5000

8. Gordon Bradley...1
    Naval Postgraduate School
    Monterey, CA 93940-5000

34.   Capt Mark Murray USAF 1
        Joint Battlespace Infosphere (JBI)
        AFRL/IFSE
        Building 3, Room E-1078
        525 Brooks Road
        Rome, NY 13441-4505

35.   Michael Myjak 1
        Vice President and CTO
        The Virtual Workshop
        PO Box 98
        Titusville, FL 32781

36.   Neal Park, President 1
        Nexternet, Inc.
        2900 Gordon Ave.
        Suite 202
        Santa Clara, CA 95051

37.   George Philips...1
        CNO, N6M1
        2000 Navy Pentagon
        Room 4C445
        Washington, DC 20350-2000

38.   Dr. Mark Pullen & Dr. Robert Simon...2
        Department of Computer Science/C3I Center MS4A5
        George Mason University
        FairFax, VA 22030

39.   Dick Puk..1
        President
        Intelligraphicas Incorporated
        7644 Cortina Court
        Carlsbad, CA 92009-8206

48.   Walter H. Zimmers 1
        Defense Threat Reduction Agency
        CPOC
        6801 Telegraph Road
        Alexandria VA 22310-3398

49.   Dr. Michael Zyda, CodeCS/Zk 1
        Director, Modeling Virtual Environments and Simulation (MOVES) Institute
        Computer Science Department
        Naval Postgraduate School
        Monterey, CA 93940-5000