MANIC: Multimedia Asynchronous Networked Individualized Courseware *

James F. Kurose   Hu Imm Lee †   Jitendra Padhye   Jesse Steinberg
Mia Stern
Department of Computer Science
University of Massachusetts
Amherst MA, 01003
email:{kurose, hilee, jitu, steinber, stern}@cs.umass.edu

Abstract

The MANIC project in the Department of Computer Science at the University of Massachusetts is aimed towards facilitating individualized asynchronous distance learning. We take advantage of current technology, such as the World Wide Web and high-speed networks, to present a rich combination of text, graphics, and sound. We use student modeling to aid in the presentation of the material, including individualizing the instruction for each student and reducing the delays by predicting what the student will see next. In this paper, we present the initial work on the MANIC system and discuss the future possibilities and research.

keywords: multimedia, asynchronous, networked, individualized courseware, WWW, student model, prefetch

1 Introduction

With rapid advances in telecommunication technology, distance learning is fast becoming a viable adjunct to the standard classroom model of learning and teaching. The MANIC project is an attempt to harness the power of general purpose computers (e.g., web servers), networking technologies (e.g., the Internet and higher-speed local area networks), and information presentation tools

*This work is supported by the National Science Foundation under contracts NCR-9508274 and CDA-9502639. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

†H.I. Lee is a student at Mt. Holyoke College, South Hadley, MA

(such as WWW browsers and multimedia browser plug-ins) to create an asynchronous multimedia learning environment. Specifically, we are exploring how to most effectively use WWW multimedia technology (HTML and audio/video browser plugins such as RealAudio together with customized WWW servers) to deliver stored course or training materials such as class notes/overheads and audio/video of classroom lectures to students.

There are two ways of presenting educational material on the web: synchronous and asynchronous. Synchronous distance learning environments focus on creating an illusion of a classroom. This includes the delivery of live classroom lectures via satellite channels and video recordings of live classroom presentations. Internet-based conferencing tools have also been used to create live, networked distributed "classrooms" [6]. On the other hand, asynchronous systems are designed to allow students to use the software at any time, without simulating a classroom. With these systems, students proceed at their own pace and are not assumed to be accessing the same material at the same time as others. The goal of these systems, such as the system described in [8], is to serve as an easy-to-access repository of reference material.

Our focus in MANIC is also on *asynchronous* learning environments, so students can access the software at any time without regard to others also using the software. Thus, while the synchronous distance learning approach attempts to recreate the classroom by connecting together distributed participants in real-time, our focus is on delivering *stored* multimedia materials and past classroom activities to distant students.

Given our focus on asynchronous student access, the goals of the MANIC project are threefold. The first goal is to develop *effective* WWW-based stored materials and presentation paradigms – to learn what does, and does not, work well by implementation and experimentation. Because we assume that a "live" instructor is not on-line with the student, our second goal is to develop student modeling and intelligent tutoring capabilities that allows the software to interactively guide a student through the stored material, providing individually tailored instruction for each student. The third goal of the project is to use MANIC as a test bed for research in the areas of multimedia networking and operating systems. In this paper, we focus our discussion on the first two goals.

In this paper we describe our initial efforts in constructing MANIC, focusing on both the software and the course materials we have developed. We also discuss the various larger lessons we have learned thus far. Although the MANIC project is still under development, a preliminary version of the system is currently being used in a for-credit course being offered in the Fall of 1996 at the University of Massachusetts, with additional courses being scheduled to use MANIC software in Spring of 1997.

The remainder of this paper is structured as follows. Section 2 overviews the hardware and software components of MANIC, as well as how course materials used for MANIC are created. Section 3 describes the audio synchronization feature of MANIC. Section 4 describes the various student-oriented features that we have built into MANIC and discusses their implementation. Section 5 describes future plans for the project. Section 6 concludes the paper.

2 MANIC: multimedia, asynchronous, networked individualized courseware

Currently, MANIC delivers text (HTML documents and still GIF images) synchronized with audio (encoded and delivered using a RealAudio server) over the Internet to students with multimedia capable machines with WWW browsers. This architecture provides a classroom like learning experience (unlike [8]), and also allows for asynchronous learning (unlike [6]). RealAudio codes audio for potential delivery over 28.8Kbps dialup lines and hence MANIC course materials can be accessed at home, at school, or at work. Our design also has provisions for a video server (MPEG) to be added as well, and we are currently adding this functionality. In this section we first overview the MANIC hardware and software components. We begin by describing the "raw" classroom materials we start with.

2.1 Classroom Materials

MANIC makes classroom materials, including the HTML of the instructor's class notes (overheads), classroom audio and classroom video available to students over the WWW. The HTML notes are provided by the instructor. Initially, the notes we used were exact duplicates of the paper notes (specifically, a copy of the class overheads used by the instructor) used by in-class students. However, we soon realized that the on-line notes could (and should) be augmented with embedded hyperlinks to related material.

The course that we have already made available on-line via MANIC is a six-hour course on UNIX Network Programming (that was taught by J. Kurose in May 1996 in the studios of the Video Instructional Program (VIP) at the University of Massachusetts and distributed via the National Technological University). Since the course was taught in the VIP studios, we were provided with professional-quality recorded audio and video of the live classroom activities, which were then digitized.

The on-line MANIC-based version of this six-hour course is currently being used as half of a 1-credit Fall 1996 on-campus mini-course on Network Programming and Java. The course material (overheads and audio) is on-line at `http://www-aml.cs.umass.edu/~amldemo/`.

2.2 MANIC Overview

The MANIC architecture is shown in Figure 1. On the server side, MANIC is built around a standard WWW server, with customized common gateway interface (CGI) scripts that generate, "on the fly", the HTML document to be sent to the client. As we will see shortly, the ability to generate/customize a document on the fly allows the server to dynamically and individually tailor a document for an individual student. Currently, our primary use of this tailoring is to highlight selected portions of a document in synchrony with the audio.

The client accesses the course material using a WWW browser (such as Netscape Navigator or Internet Explorer) and the RealAudio and other multimedia plugins. The WWW server sends HTML

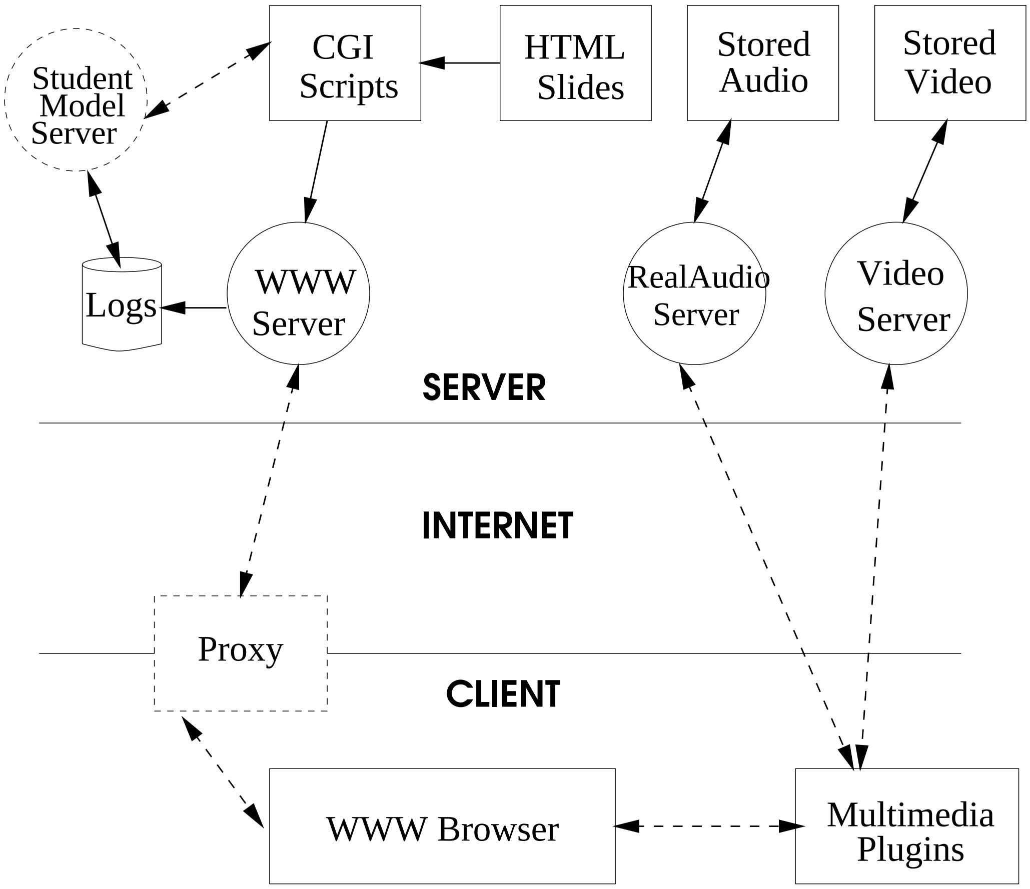

Figure 1: MANIC structure

text to the client browser. An audio stream is also sent to the client plugin by our separate RealAudio server. Audio playout and HTML presentation are synchronized and controlled at the client computer using the synchronized multimedia feature provided by the RealAudio server/player. See section 3 for a discussion of this feature.

The client side components of MANIC are all “off the shelf” pieces of software. Indeed, one of our design criteria in building MANIC was that we *not* develop specialized client-side software beyond some simple Javascript programs. Instead, we leveraged WWW browser and plug-ins (such as the RealAudio client, which is available from RealAudio as a free download) as much as possible. This not only minimized our own software development effort, but aided in portability as well – companies that make browsers and plug-ins naturally want to make their products available on as many platforms as possible. Using off-the-shelf client software allows any student with a multimedia PC, WWW browser, and an Internet connection to use MANIC without having to download any MANIC-specific software. In retrospect, this decision to leverage existing client side software and focus our development efforts at the server side was a good one.

There are two optional pieces to the architecture. The first is the student model server, which keeps track of activities of individual students and constructs a model of the student's behavior. This model can be used to customize the course material to tailor it to individual student's need (done on-the-fly). The second optional component of MANIC software is the proxy. As shown in figure 1, the proxy is neither at the client nor at the server. This piece of the software is primarily used to reduce the response time (download time) of client requests. The proxy is advised by the student model server on the documents a particular student is likely to see next. The proxy prefetches this document from the web server and stores it on a local disk. If the “guess” made by the student model server is accurate, the download time for the document will be negligible, as the proxy will be able to serve the document from its local cache [7].

3 The synchronized multimedia facility

MANIC uses the synchronized multimedia facility provided by the RealAudio server/player to present courseware material to the users. To use the synchronized multimedia facility, it is necessary to set up four kinds of files:

- **The RA files:** This file contains audio data encoded in a format suitable for playback by RealAudio player/server. We used one file per lecture, so each of our RA files represents about 60 minutes worth of audio.
- **The RAE files:** Each RA file is associated with an RAE file. The RAE file records the relationship between the audio stream and the URLs to be displayed by the browser. The entries in the RAE file contain a timestamp followed by the URL to be displayed. The RealAudio player downloads the RAE file at the beginning of each session. It then starts downloading and playing the corresponding RA file (RA and RAE files have names that differ only in the extension). As it continues playing the audio, at each timestamp recorded in the RAE file, the player sends a signal to the associated HTML browser (e.g. Netscape),

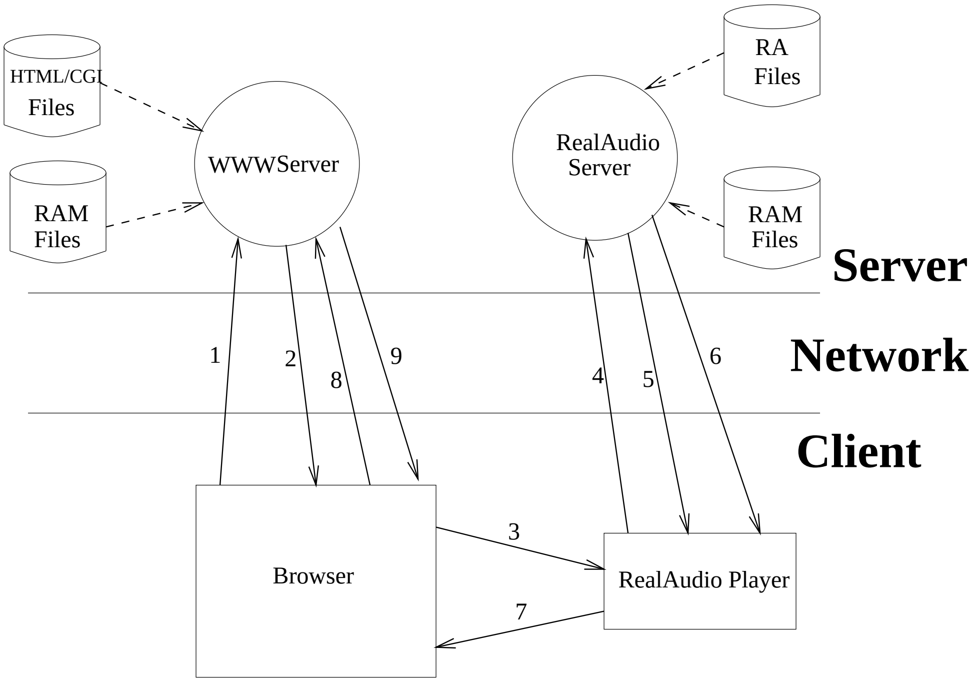

Figure 2: The synchronized multimedia operation.

requesting it to load the corresponding URL. This creates an illusion of a "slide show" to go along with the audio playback.

It was initially perceived that digitizing the audio from the lectures would be the trickiest part of the whole exercise. It turned out that this was fairly easy; however, synchronizing it with the text display (determining the timestamps in the RAE files) was what took most of the time.

- **The HTML files:** The slides for the lecture are converted into HTML documents. The URLs in the RAE files point to these files.

- **The RAM files:** These files provide the link between the RealAudio player and the HTML browser. The RAM files are available on the HTTP (web) server. Each RAM file contains a single line of text specifying the URL for the RA file. When the HTTP server serves this file to a browser client, (due to, say, the user following a link), it also notifies the client that the contents are not to be displayed by the browser itself, but rather, should be given to an external application capable of handling RealAudio. The browser then invokes such a player (if not already present) and hands the contents of the file to the player. The RealAudio player then uses the URL to retrieve the RAE file and the RA file and commences playback.

The interaction between the RealAudio player, the RealAudio server, the browser and the WWW server is illustrated in Figure 2. The sequence of operations is as follows:

1. The user clicks on a link that points to a RAM file. The browser sends a request to the WWW server for the file.
2. In response to the request, the RAM file is sent to the browser, with `content-type` field in the header [1] set to `audio/x-pn-realaudio`.
3. Based on the entry in the `content-type` field, the browser invokes the RealAudio player and hands to it the contents of the of the RAM file.
4. The RealAudio player decodes the URL specified in the RAM file and sends a request for the RA file specified in the URL to the RealAudio server (also specified in the URL).
5. The RealAudio server, after receiving the request for the RA file, checks to see if it also has a RAE file by the same name (i.e. a pair such as `foo.ra` and `foo.rae`). If it finds such a file, the entire file is then sent to RealAudio player. As noted previously, the RAE file contains a list of `<URL, timestamp>` pairs.
6. The RealAudio server then starts streaming the contents of the requested audio file to the RealAudio player.
7. Based on the epoch in the audio stream and the entries in the RAE file, the RealAudio player sends a signal to the browser indicating the URL to download.
8. The browser uses this URL to contact the WWW server.
9. The server sends the requested document to the browser which is then displayed by the browser.

It is interesting to note that the when an HTML browser and a RealAudio player work together to provide such a slide show, the controlling entity is the RealAudio player. The browser simply displays the URLs it receives from the RealAudio player. If the users attempts to change the currently displayed document (by using, say, their bookmarks list), their choice will be overridden by the RealAudio player when it reaches the next timestamp in the RAE file corresponding to the RA file that is being played. This relationship between the player and the browser has had significant impact on the design of the MANIC user interface. Since RealAudio does not provide support for interactivity via the browser, we were forced to implement some "workarounds". For example, for the user to stop the audio, it is necessary to download a "silence" file that the RealAudio player could play, thus stopping the playback of the previous audio file. It should be noted that when the audio is playing, the RealAudio client is in control of what is being displayed by the Web browser. Therefore, every time the user views a different slide which was not signaled by RealAudio, the audio is stopped to ensure that the RealAudio player does not override the user's choice of which slide to view.

4 Features of the MANIC Environment

In this section we describe the user's interaction with the MANIC system as well as the features of the system that enhance this interaction.

4.1 Non-linear perusal of course material

A design goal of MANIC was to allow students to have an active role in the way they see the instruction. Interactivity is important in order to keep students interested in the material [5]. To provide for interactivity, students are given the opportunity to browse the material at their own pace, stopping and starting the audio at will. However, students also have the option to view the material as they would in a traditional classroom (e.g., where synchronized audio and HTML playout continues non-stop for 60 minutes). Students can switch between either mode (browsing or passively watching the course material) at any time; they are not restricted to only one mode of interaction. When a student begins browsing after having listened to the audio, the playback is stopped. The student must then begin the audio again explicitly if desired.

Because we want to allow students to browse the material in a non-linear fashion, we have decided to include a method in which students could jump to other parts of the course easily [3]. For this reason, we have included a table of contents which allows students to see the structure of the material as well as to jump to new material with one mouse click. In the future, we will we adding an index of topics (in addition to a table of contents) as well as a text-based topic search engine. To further allow for easy browsing, we have added "next" and "previous" buttons to each slide. These buttons link to slides immediately following and immediately preceding the current slide, in the linear ordering. In this way, students can linearly browse through the course without relying on the index.

Since students may be browsing the material, they may want to hear the audio for only one slide and no others. Therefore, we have designed two play modes: play continuously from the currently displayed slide onward, and play the audio for the current slide only. The first mode will play the audio from the current slide and continue until the user stops the sound, automatically changing the slides during the playback. The second mode will play the audio only until the end of the slide is reached.

4.2 Highlighting of course material

The earliest versions of MANIC simply displayed a static HTML page while the audio associated with that page was being played out. It soon became apparent, however, that such a static display was rather dull. To make the material more interesting and to capture/focus the student's attention, we chose to dynamically highlight selected portions of text within the single "page" of notes displayed by the browser. Now as the audio progresses through the displayed page, the highlighting within the page changes. This helps the student to follow the material on the slides while listening to the audio.

Dynamic highlighting is achieved by downloading a different HTML document each time the highlight changes. Thus, several documents may contain the same basic text but will have different portions of the text in the highlight color (red). The repeated download of HTML documents proved to be problematic over 28.8K modems. Half of the bandwidth was occupied by the audio stream almost all the time. The remaining bandwidth proved to be too little for effective downloading of the new document each time the highlight changed. This led us to provide an option to the user whereby the material can now be viewed with or without highlighting. In general, we have decided to categorize users by bandwidth, CPU processing power, and memory available to them. The users will see a different version of MANIC materials depending on the category to which they belong. For example, in the future, users with powerful workstations and multi-megabyte bandwidth could be provided with motion MPEG video while users with dialup lines would not be able to receive video. Off campus students may also be denied video service due to the University's limited T1 bandwidth into the Internet.

4.3 Displaying supplementary material

Simply putting the slides and audio on line without any changes is rather shortsighted. While traditional classrooms require linear delivery of material, the WWW allows for non-linear presentations. To take advantage of this, we have added the index as well as hypertext links to supplemental materials. The hypertext links present related information to the student, sometimes in the form of review material and other times in the form of interesting sidenotes.

However, when students access additional material, they should not lose the original context [9]. Therefore, when a student clicks on a hypertext link or chooses to view the index, an additional window is spawned to present that material. With this method, students can view all the information simultaneously.

4.4 Putting user controls in HTML documents

As discussed in section 2.2, the RealAudio player is actually the controlling application on the client side. However, the RealAudio player provides a cassette-deck-like interface to audio. Some RealAudio controls, such as those that start and stop audio/HTML playout are appropriate for a stored delivery system such as MANIC. Other aspects of the RealAudio interface are not appropriate. For example, the only way to index into audio using a RealAudio player is by time (e.g., "move the playout to 35 minutes and 16 seconds into the audio"), whereas a better method for indexing into material is by content (e.g., "move forward to where the instructor begins covering a specific topic"). In addition, user commands for requesting help and sending in commentary on MANIC are also required.

Thus, we decided that user controls needed to be placed within the HTML documents (i.e., displayed and accessible in the "page" of notes being displayed in the WWW browser) rather than via the standard RealAudio control. In essence, we "hid" the RealAudio control interface from the student using our own control interface instead. A screen shot of MANIC, showing our control interface at the left hand side is shown in Figure 3. Each of the control buttons shown in Figure 3 activates a CGI script on the server which forces the RealAudio client and server to perform the requested operations.

Since RealAudio does not provide support for interactivity via the browser, we were forced to implement some "workarounds". For example, for the user to stop the audio, it is necessary to download a "silence" file that the RealAudio player could play, thus stopping the playback of the previous audio file. It should be noted that when the audio is playing, the RealAudio client is

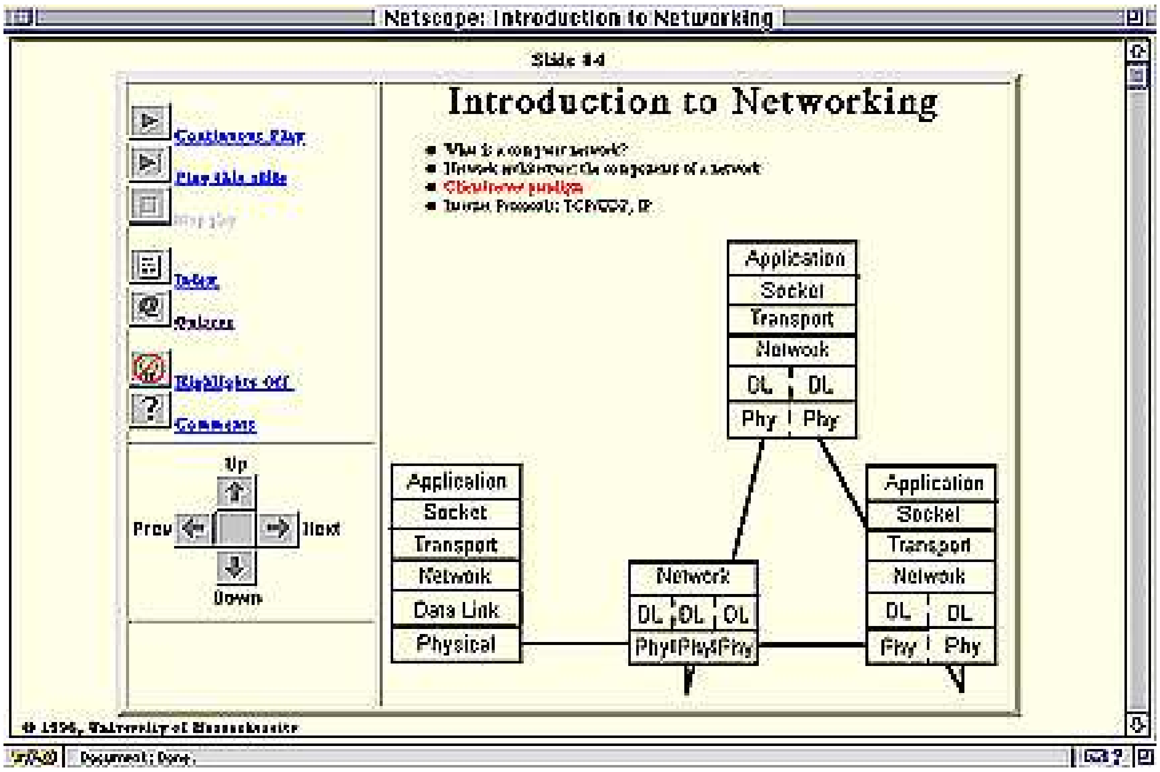

Figure 3: A screen shot of MANIC course material.

in control of what is being displayed by the Web browser. Therefore, every time the user views a different slide which was not signaled by RealAudio, the audio is stopped to ensure that the RealAudio player does not override the user's choice of which slide to view.

4.5 Software and hardware portability

We decided early in the project that the presented material should be portable across several hardware and software platforms. This implied that none of the browser specific features (e.g. style sheets supported by IE 3.0) could be used. Widespread use of Javascript was also not possible due to bugs and minor inconsistencies in the support for different platforms. Support for interfacing Java with RealAudio was available only on the WIN95 platform, so that option was considered infeasible.

4.6 On-the-fly generation of HTML documents

Most of the MANIC material presentation is generated "on-the-fly" by extensive use of server-side CGI processing. On-the-fly generation of material is essential to facilitate customization of MANIC material for individual needs. It also allows for multiple highlight areas to be defined in the same slide using special tags. These tags are interpreted at run time and the correct text portion is highlighted.

However, the current delivery of the course still retains many aspects of the traditional classroom. For example, the slides are mostly black and white. Furthermore, the audio for the course is focused towards an "in-class" audience. The instructor does not tell the student about the hypertext links. Also, the presentation of the material is not yet customized towards each individual's students likes/dislikes and abilities.

4.7 User tracking and intelligent tutoring

One of the long term goals of the MANIC project is to provide the users with training material tailored to their rate of progress and other individual preferences. To achieve this goal it is essential to keep track of actions taken by individual users. The HTTP communication protocol does not support this as it is a stateless protocol. Hence single-session cookies [4] were used to create a "session" out of the stateless HTTP protocol. The cookies carry information such as user login name and visual preferences and allow user-specific actions (such as logging information in proper files) to be taken.

The same mechanism allows us to distinguish between registered and unregistered users. Registered users are required to fill out a small HTML form at the beginning of each session. The form and the associated CGI script captures and sets the user id in the session cookie. If this information is not present, the user is considered to be an unregistered user. Using the information in the user id field of the cookie, it is possible to track users' progress, their access patterns etc.

5 Future Work

In this section we discuss ongoing and future work for the MANIC project. Specifically, we discuss how to make the course material more individualized for each learner.

5.1 Making the material less like a "course"

The materials used for the first course made available via MANIC were directly adapted from pre-existing in-class materials. The current delivery of the course still retains many aspects of the traditional classroom. For example, the slides are mostly black and white; the audio for the course is focused towards an "in-class" audience; the instructor does not tell the student about the hypertext links in the on-line materials. These will change as instructors gain more experience with teaching a live class while keeping an eye towards making the materials available on the WWW via MANIC.

As noted earlier, our ability to generate material being sent to a student "on the fly" opens up the possibility of customizing MANIC material for individual student needs. Thus far, the only customization of the material is the part of the slide that is being highlighted. However, a main focus of our current work is on the use of student modeling and intelligent tutoring. From an educational standpoint, it is imperative that we incorporate more intelligent tutoring properties into the current system [2]. One way to do this is to add quizzes which students can take on the course material. The grades from these quizzes can be used to update a "student model", which is used to track a student's abilities on the topics within the course. Furthermore, given that MANIC tracks and records every action the student takes, it knows which slides have been seen, and thus which material has already been covered. With this information, the system can tutor students by suggesting material on which they should concentrate, or even alter the material the student sees based on the student model. The tutor can also provide quizzes to the student that are at the right level of challenge. With intelligent tutoring, we can tailor the presentation of the material so that it is customized towards each individual's students likes/dislikes and abilities.

5.2 Intelligent prefetching

One way to reduce delay in network transmission is to prefetch the material that will be displayed to the user. While this is not a new idea, we intend to extend the prefetching process by using intelligence derived from the student models. The system can predict (with some uncertainty) the slides the student will be viewing next, and can download those to the client before they are explicitly requested. For example, if a student is proceeding through the course in a linear fashion, the system can detect this and send the next set of slides in the ordering.

To implement the intelligent prefetching, we are using a proxy that will work between the client's browser and the server. This proxy will be able to cache course material. When a request is made from the client, if it is in the cache, then the data does not have to be downloaded from the server. The information on exactly what to cache is determined by the server. Whenever the proxy connects to the server (which it will do even when the page is served from the cache, if only to log the user's action), the server will determine from the student model, and possibly from more general population models, what should be cached. The proxy will read this information, and will then request pages to be sent to the proxy's cache – hopefully just before they are needed by the student. The server can also send information about when to remove data from the cache. If it is unlikely a student will need to see a slide that has been cached, then it should be removed.

5.3 Annotations

The ability to make annotations on the course material is an important learning tool. The annotations can be classified into three categories: personal annotations, annotations visible to a certain group and "global" annotations. Personal annotations would be made by individual students and the text of those would be visible only to the student making the annotations. Group annotations can be made by any person in a specific group and all members of that group will have access to them. Global annotations can be used by the course instructor to clarify difficult topics. Note that the annotations can use HTML syntax and include links to other slides in the course material or to outside sites (e.g. Request for Comments (RFCs) of various networking protocols). Currently we are designing an "Annotation Server" that will support such note-taking capabilities.

5.4 Client side processing

One of the current priorities for us is to perform as much of data processing on the client side as possible. Doing more processing on the client side may result lowering the bandwidth requirement and lower response times. The client side tasks range from relatively simple ones such as processing highlights (done using Javascript) to more complex ones like maintaining local copies of student models and making prefetching decisions. Currently, slide highlighting is done by downloading a different HTML document each time, with font color of the relevant section changed. Using Javascript on the client side, we can achieve the same effect by re-rendering the slide after each highlight change.

5.5 Use of Multimedia databases

Currently the MANIC material (the slides and the audio) is stored in form of simple Unix files. As the architecture becomes more sophisticated, and as more and more users start using the system, this approach to data storage/management is unlikely to scale well. To achieve customized delivery of course material based on student's preferences and his/her past performance, it will be necessary to store data at a much finer granularity. Currently the audio is stored using one large (approx. 3 MB) file per lecture (60 minutes) while the quizzes/slides are stored in individual files. Our preliminary analysis shows that finer granularity (audio accessible in short, approx. 1 minute long segments) is needed to achieve even simple customization of course material. Such large number data items are most efficiently managed by a multimedia database with native support for specifying and achieving quality of service during both storage and retrieval. For example, an audio database may be required to retrieve and deliver specific audio segments at a certain average rate (bytes/second) while guaranteeing a specified jitter bound. Retrieval of raw data and blending it together to form customized course material can be achieved by using SQL or other database languages. (Currently this is done by ad-hoc file manipulation).

We also envisage adding video to the MANIC presentations for high bandwidth users. Since video data is much more voluminous than audio data (approx. 1.5 MB/second for MPEG encoding), the need for a database manager becomes even more pressing. In addition to guaranteeing quality of service, such a database manager will also have to provide an "open" standard for implementing buffer management strategies. As mentioned in the previous section, we have envisaged several intelligent buffer management and prefetching strategies that take advantage of common patterns in student accesses to the MANIC material to speed up access.

6 Conclusion

We have built a simple yet flexible distance learning environment that takes advantage of the latest advances in the computation and networking technologies. This prototype learning environment has several interesting features such as user tracking which can be used to provide individualized instructions to each student. Future work includes implementation of intelligent prefetching algorithms, building of an annotation server and creating a prototype course which can be customized for individual student's needs. We are also working on incorporating video into the available on-line course materials.

References

[1] [Bern96] "Hypertext Transfer Protocol - HTTP/1.0", T. BernersLee, *Network Working Group, RFC 1945*, May, 1996.

[2] [Bloom84] "The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring", B. Bloom, *Educational Researcher*, volume 13, 1984, pp. 3-16.

[3] [Eklund95] "Cognitive Models for Structuring Hypermedia and Implications for Learning from the World-Wide Web", J. Eklund, *AUSWEB95- The First Australian World-WideWeb Conference*, http://www.scu.edu/ausweb95/papers/index.html, 1995.

[4] [Gund96] *CGI Programming on the World Wide Web*, S. Gundavaram, O'Reilly & Associates, Inc., Sebatopol, CA, 1996.

[5] [HS95] "Learner Control in Full and Lean CAI Programs", R. Hannafin and H. Sullivan, *Educational Technology Research and Development*, volume 43, number 1, 1995, pp. 19-30.

[6] [Lieb96] "gwTTS: A grounds-wide Tele-Tutoring System", J. Liebeherr et. al., `http://www.cs.virginia.edu/ gwtts/final2.html`.

[7] [Pad96] "Using Predictive Prefetching to Improve World Wide Web Latency", V. Padmanabhan and J. Mogul, *Proceedings of ACM SIGComm*, 1996, pp. 22-36.

[8] [Rebe96] "Improving WWW-Aided Instruction: A Report from Experience", A. Rebelsky, *Proceedings of ED-TELECOM 96*, June 17-22, 1996, pp. 276-281.

[9] [Stinner] "Contextual Settings, Science Stories and Large Context Problems: Toward a More Humanistic Science Education", A. Stinner, *Science Education*, volume 79, number 5, 1995, pp. 555-581.