Seminal: Additive Semantic Content for Multimedia Streams

Sami Rollins
Department of Computer Science
University of California
Santa Barbara, CA 93106-5110
srollins@cs.ucsb.edu

Kevin C. Almeroth
Department of Computer Science
University of California
Santa Barbara, CA 93106-5110
almeroth@cs.ucsb.edu

Abstract

Technological advances such as higher network bandwidth and greater end-user computing power provide the basis for new types of media rich applications. As applications produce larger numbers of more diverse media streams, the content becomes too overwhelming to be useful in its raw form. The contribution of this work is the initial design of *Seminal*, a model that solves the problem of multimedia overload by enhancing multimedia streams with semantic information about their content and relationship. The goal of *Seminal* is to manually or automatically derive semantic meaning from a given set of media streams. When the media streams are presented, archived, or distributed between users, the semantics are used to filter the most relevant information from the entire information base. We have designed a digital classroom-based prototype to validate our assumption that semantic information can be used to allow users to interact in a media rich environment.

1 Introduction

Technological advances such as higher network bandwidth and greater end user computing power provide the basis for new types of media-rich applications. These multimedia applications enable users to communicate and interact in more advanced ways. For example, multimedia enhanced classrooms can allow instructors to teach local and remote audiences alike using a full compliment of physical gestures, voice, and visual aids. Currently, college courses are captured and distributed using technology such as Real Networks streaming tools or multicast-based tools such as *vic* and *vat* or IPTV[1]. Business meetings are conducted using the Access Grid as well as other tools including Microsoft NetMeeting and Cu-Seeme. While these tools largely support straightforward audio and video exchange, a number of challenges remain.

As the number of potential media types becomes larger and more diverse, the content produced becomes too overwhelming to be useful in its raw form. For example, a user may not be able to receive and display more than a few video streams simultaneously; an end user's network connection may only be able to handle a subset of available media streams; multiple media streams may need to be synchronized and presented together; or a single stream may need to be compressed or condensed. A more specific example is what might happen if a remote student is trying to watch a lecture consisting of instructor video, powerpoint slides, whiteboard content, and video of students in the classroom. In addition, existing tools may not be able to handle new requirements such as scalable distributed collaboration. Raw information dissemination is simply not sufficient for next-generation multimedia applications.

The primary contribution of this work is the design of *Seminal*, a model that solves the problem of multimedia overload by enhancing multimedia streams with semantic information about their content and relationship. In *Seminal*, media streams are decomposed and semantically enhanced with application-specific semantic metadata. The metadata and media streams are then disseminated and received by *Seminal services*. *Seminal* services use the metadata to interpret the raw media and perform a service-specific action. We illustrate the benefits of *Seminal* by describing its use in a digital classroom environment. Many universities have implemented technologically enhanced, interactive classrooms. A typical classroom might include digital video cameras, data projectors, instructor laptop(s), student laptops (or Personal Digital Assistants), a VCR, a DVD player, and remote student video feeds. Using *Seminal*, we can automatically generate metadata about the plethora of media streams and use the metadata to display streams in the classroom itself as well as to efficiently deliver streams to and display streams for remote students viewing the lecture.

Section 2 presents an overview of the *Seminal* model and its use in the digital classroom environment. In Section 3, we look at a set of techniques for semantic content extraction and Section 4 presents a set of services that make use of the semantics and media streams. Finally, we conclude in Section 5.

2 The *Seminal* Model

Media streams provide limited benefit in their raw form. Applications such as digital classrooms simply generate too much information to be distributed and displayed for end users. Many efforts have focused on developing solutions for digital learning environments [1, 2, 3, 4, 5, 6]. However, those solutions are generally ineffective because they do not communicate the learning process. The best current solutions support only two-way audio and video streaming. But, a number of problems still exist. Me-

dia may need to be synchronized and organized for display, transcoded or filtered to meet end user network restrictions, filtered to support end user display capabilities, and indexed for later retrieval. Our goal is to address these problems using the *Seminal* model. In *Seminal*, control information accompanies the raw media streams to provide both *intra-stream* as well as *inter-stream* information. It uses semantic metadata to identify how media should be composed or decomposed for the end application. The Semantic Multicast project[7] proposes a similar framework. However, the focus of the Semantic Multicast project has largely been database storage and retrieval. Our goal is to enable a broader range of end-user services.

Seminal serves two functions. The first function is to create metadata about the media streams in an application. Intra-stream metadata describes properties of a particular media stream. For example, video of an instructor in a classroom may be accompanied by instructor and lecture information including the time, date, topic, and an outline of the material. Inter-stream metadata contains information about how multiple streams in an application relate. In a classroom application, inter-stream metadata might include timing information to indicate how to synchronize across multiple streams.

The second function of *Seminal* is to analyze the metadata to provide a set of services. *Seminal services* use the semantics provided by the metadata to interpret and use the raw media generated by an application. *Any* tool that uses the *Seminal* metadata to interpret and make use of the media streams is providing a *Seminal* service. Unlike current tools, *Seminal* services can manipulate, filter, or display media streams in application-specific ways. Services may be provided at the source, in the network, or at the user side. In the case of a classroom application, they may include routing media streams from the main classroom to remote student sites, displaying the top priority stream in the user display, and synchronizing the display of static information such as a PowerPoint presentation with the display of realtime streams such as video.

We identify three components that are integral to our framework, the **content**, the **metadata**, and the **metadata filters**. Figure 1 illustrates each of these components. These components together comprise a general framework that is used to support many types of collaborative applications. The novelty of the model is the use of semantic control information. The goal is to augment existing applications so their basic functionality does not have to be re-implemented, but so they can be enhanced to offer additional metadata-based functionality. Therefore, we can integrate existing tools such as Real Player as well as create new tools that may be used for a number of different *Seminal* applications.

In the remainder of this section, we discuss the three *Seminal* components and illustrate their use in a digital classroom environment.

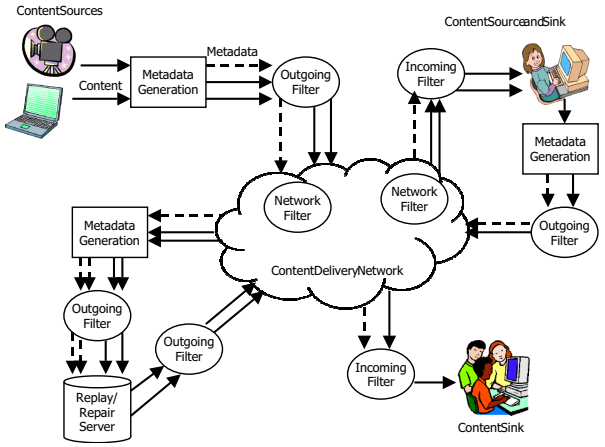

Figure 1. An overview of the *Seminal* model.

2.1 Content - Sources and Sinks

In any multimedia application, there may be a number of content sources and sinks. A single user may source multiple different streams such as audio, video, and text. Many entities will function as both a content source and a sink. For example, a student may watch video generated by other students, but may also generate a video stream to be sent to the rest of the group. In many applications, a single stream will function as the primary stream while the remaining streams are supplementary and not as important.

The UCSB classroom infrastructure [8] supports a main classroom site, secondary remote classroom sites, and tertiary remote student sites. The main site is the primary content source. Video feeds generated at the main classroom site include video of the instructor, video of the audience, up to two computer video feeds, a video feed of a remote site, and/or video from a VCR source. Each source may also generate one or more audio feeds. Secondary remote student sites may send media streams (e.g. video and audio) back to the classroom site where they are displayed using a data projector. Tertiary student sites do not function as content sources. They only receive and display media streams. At the main site, a maximum of three of the available streams (including a remote stream) are selected using a commercial video matrix switcher and sent to the remote audience. The audio streams are mixed into a single signal and distributed as well. The content sinks in the classroom include the secondary remote student sites, the tertiary remote student sites, and the data archive. The sinks archive or display the three video streams as well as the single, mixed audio stream.

2.2 Metadata and Metadata Generation

Metadata is semantic information about one or more media streams. It may describe information about a single stream or may contain information relating multiple streams. The

metadata *schema* is a description of what information the metadata may or must contain, and the format in which it must be written. While it is possible to define a general schema to encompass many different applications, the specifics of the schema are left to the application-specific implementation.

Seminal supports metadata generation both at content sources and sinks. In the classroom environment, metadata can be information about the media streams generated by the main classroom site as well as by the remote sites. The metadata may describe the context of one or more media streams, events that occur in the streams, or how multiple streams relate to one another. Examples of straightforward content descriptions would be the name of the instructor shown in a video stream, the date the lecture was given, and the subject of the lecture. Event metadata describes actions that are captured in the stream such as the instructor switching from one slide to another. Metadata describing the relationship between streams can indicate synchronization points or presentation cues.

Metadata creation can be done in a number of ways. Ideally, the process should be as automated as possible. However, in some cases more manual solutions may be required. The most straightforward way to provide metadata for a media stream is to require a user to manually create it. An example of this might be allowing an instructor to use a digitizing tablet to alter PowerPoint slides and then distributing the slides and the annotations to users. Manual solutions are straightforward to implement and ensure that the instructor has control over the semantics of the metadata. However, more automatic methods alleviate the burden on the instructor. An improvement over manual creation is to use classroom events to create semantics. For example, an instructor may define a web path, or set of URLs prior to the lecture. During lecture, she can speak a given phrase to traverse the list. However, even this semi-automatic approach may prove to be too cumbersome. Ideally, we could develop tools that would automatically create semantic content for any given media. An example might be an automatic indexing system that indexes the video of an instructor teaching with keywords extracted from the audio track of the video. Unfortunately, automation may require intelligent systems or excessive computational power. Therefore, it may be technically infeasible or may not be possible to implement as a realtime service.

2.3 Metadata Filters - User Services

The final component of the *Seminal* model uses the semantic information provided by the metadata generation components to provide *Seminal* services. An implemented service uses a *filter* to process the metadata to determine the semantics it provides. The semantics are then used to interpret and process the media streams. Therefore, we refer to the service components as *filters*. Any tool that uses the semantics provided by the metadata to interpret and use the raw media data in an application is considered a service or filter component.

Filters may exist at the source, in the delivery network, or at the sink. At the content source, a filter can both display media as well as determine how it should be disseminated. Filters within the delivery network may provide customized delivery and routing options. Incoming filters at the client side use metadata to present information to the end user.

The need for *Seminal*-based classroom services is motivated by a number of factors. Network bandwidth restrictions, organization and presentation of media, client-side display capabilities, as well as user preferences all drive the need for user services. Examples of categories of services include media presentation, media delivery, and media indexing and retrieval. A presentation service can select the primary streams and display them using the resources available at a given site (i.e., primary, secondary, or tertiary). A delivery service may transcode or select a subset of the available media to be sent to the end user. Finally, to enable retrieval, services must provide access to stored media streams. This might include query interfaces to allow students to search for specific portions of the lecture such as when the instructor mentions exam information. This might also include replay servers where students can request replay of previous lectures.

3 Extracting Semantics from Audio Streams

The first phase of development of our *Seminal* implementation focuses on generating metadata based on the instructor's speech. Using a speech-to-text engine, we create *annotations* for the media streams generated in the classroom and distribute the annotations along with the media streams. An annotation is a single piece of semantic information. The collection of annotations comprises the set of semantic metadata.

The first goal of our prototype is to provide realtime services to remote students. In addition, we want to minimize the amount of time the instructor must spend in additional preparation. This section discusses three techniques that we employ for metadata generation in our *Seminal* prototype: command definition, key phrase recognition, and predefined configuration. Section 4 provides more detail about how we use metadata to provide user services.

3.1 Command Definition

Command definition is a relatively explicit, manual technique to provide the instructor with the ability to specify exactly what a given annotation should look like. A command can be issued in two ways. The first possibility is to define a single word or phrase that is used as the annotation. For example, the instructor speaks the command "next slide" and the annotation is built using that information. The second type of command is defined by a begin keyword or phrase and an end keyword or phrase. To begin

the annotation, the begin key phrase is spoken. Everything spoken between the begin and end commands is used as the annotation. After the end command is heard, the annotation is completed and distributed to the users.

This type of annotation provides two pieces of information. Because this information was explicitly specified by the instructor, we can deduce that the information contained in this annotation is noteworthy. Any service that relies on filtering the most important information from the produced streams can focus on this kind of information. Also, this gives the instructor the ability to provide semantics about the various media being produced at the lecture site. The exact semantics depends on several factors: what the command is, what the instructor chooses to say, and how it is used by the user services.

The clear benefit to this approach is that there is no guesswork involved. It gives the instructor full control over defining the semantics of the media streams. The problem with this approach is that it does not meet our goal of automated extraction of semantic content. While the speech-based interface is designed to be more intuitive and less cumbersome for the instructor, the method itself is still manual. The next two techniques attempt to provide a more automatic solution.

3.2 Key Phrase Recognition

In some cases it may be possible to determine what an annotation should look like based upon a single key phrase and the surrounding context. In our design, we define a set of key phrases that are recognized by the system. When one of those key phrases is spoken, the following words are analyzed to determine what an annotation should look like. In most cases, the phrase following the key phrase should fit into an explicit template. However, it is possible to perform some simplified natural language processing to extract information.

An example key phrase is “assignment due”. The instructor might mention to the class that there is an “assignment due February 28th.” The metadata generation component recognizes the key phrase “assignment due” and looks for a set of words that match a template of a date. The date “February 28th” is extracted and included in the annotation.

Like the command definition annotations, key phrase annotations provide insight into the most important information discussed in a lecture. Also like command definition, the usefulness of a key phrase annotation depends on how it is used by a a corresponding user service. For example, a service might see the due date of an assignment and recognize that it needs to be added to the student's calendar.

The benefit of this technique over command definition is that it attempts to deduce important information based upon cues rather than requiring explicit specification. Therefore, as long as an instructor uses the set of key phrases, annotations can be created. This provides a more natural interface. However, the major disadvantage is that for each new key phrase, the component must be altered to accommodate the information that may follow that particular key phrase. A solution might be to integrate a complete natural language processing system that could be trained to learn what information is important for a given course.

3.3 Predefined Configuration

While it is cumbersome to require additional lecture preparation on the part of the instructor, some instructors may be willing to prepare information about the lecture topic. The prepared information is used in conjunction with keywords spoken during the lecture to create annotations. This technique is essentially a combination of the previous two techniques. The difference is that the instructor may prepare the explicit annotations in advance rather than speaking the entire text of the annotation during the lecture.

For example, an instructor who visits many web sites during her lecture may prefer to specify the URLs of the sites in a configuration file prior to the lecture. During the lecture, she uses commands to indicate which URL should be visited next. The URL is used as the annotation provided to the user service components. In this case, services can be implemented at both the user site and the classroom site. In the classroom, the instructor's voice can be used to change to a new site and automatically download the new page for classroom display.

This technique gives the instructor the ability to direct events that will occur in the classroom as well as at the remote sites. The annotations contain event triggers that are used by the service components. This is a more explicit technique than the previous two in that it does not rely on service components to interpret the annotation and act accordingly given the relative importance of the annotation.

While defining a configuration file does require some extra preparation on the part of the instructor, this can be a very powerful way to generate metadata and improve instruction. With minimal preparation before the lecture, the instructor can produce meaningful annotations during the lecture without a great deal of effort. She only needs to remember a few different command words that will be recognized by the system.

Our ultimate goal is a completely automated system that will be able to extract semantic information from any media stream and provide a series of meaningful annotations to end user service components. This section has described a first step. The techniques we describe focus on using audio tracks produced in a classroom environment to semi-automatically create metadata. The creation of more meaningful metadata is left for future work. The following section describes a set of end user service components that make use of the annotations produced using the techniques developed so far.

4 User Services

Our design focuses on providing three services to a student watching a lecture from a remote location. Each service uses the annotations provided by the components discussed in Section 3 to provide a view, or portion of a view of one or more of the media streams. In this section, we discuss each service in more detail.

4.1 Automated Whiteboard

A problem with remote lecture viewing is that it is difficult if not impossible to capture all of the activity that happens at the lecture site. For example, an instructor might turn around to a whiteboard and jot down "2/28" indicating that there is an exam on February 28th. Both local and remote students see the instructor write this information on the board. For the local student, the information remains persistent until the instructor erases that portion of the board. On the other hand, for the remote student, this information was observed because a camera was focused on the whiteboard and the output was transmitted as a video stream. The problem with this solution is that if the stream is transmitted continuously, bandwidth is wasted transmitting information that is not changing, the quality of other video streams may degrade, and the user interface must accommodate space for a rarely changing stream. However, if video is transmitted intermittently, i.e. when there are changes, (1) the information does not remain persistent at the remote location and (2) a camera operator is needed to constantly identify and target what is currently "active".

The solution we propose is to use a combination of command definition and key phrase recognition annotations to determine the kinds of information an instructor may note on a whiteboard, or the information a student might want to be persistent. Information might include exam dates or relevant exam information. When the end user component receives the annotation, the information is extracted and displayed on a local whiteboard. Figure 2 shows an example of a typical view a user might see. The information on the whiteboard remains persistent throughout the session and a user may refer to information displayed from any point in the lecture.

4.2 Display of Primary Streams

While the end user may receive and view multiple streams, viewing too many streams simultaneously can be distracting. Even in the classroom setting, there is generally a single focus. The focus may be on the instructor who is speaking, the instructor may have diverted the attention to a PowerPoint slide that is currently being displayed, or the students' attention may be focused on another source, for example a slide of a sculpture, or video of a remote site.

One solution is to provide a multi-paneled interface to allow remote students to focus on the stream they find to be most relevant at a given time. Another solution might allow students to choose one of three streams by displaying thumbnail images of all streams and a full sized image of a single selected stream. However, not only do these solutions require effort on the part of the user to try and determine stream priority, in some cases network bandwidth to the end user may be limited. Users may be forced to limit what they receive to a subset of the available material.

Our solution is to use metadata derived from command definition techniques to determine stream priority. We define a set of keywords that will allow an instructor to define stream priority while lecturing. In general, if the instructor is speaking, the instructor stream is labeled the primary stream. If the instructor mentions the *next slide*, the stream containing the slides will be assigned the top priority. If the instructor mentions that it is time to *show a video*, then the video stream will be primary. After the primary stream is identified, or when the primary stream changes, the display is updated to show the primary stream (see the left panel in Figure 2). This technique directs the focus of the student without requiring much effort on the part of the instructor. In addition, allowing the instructor to control stream priority ensures that the student will view the information the instructor finds most important rather than allowing the student to choose to watch a stream that may be unrelated to the current content.

4.3 Classroom Content Pre-Fetching

Straightforward content streaming may not always be the most logical solution for lecture material. The other two services we discuss assume that the primary media streams are simply video captured from the lecture site. However, it would also be useful to derive semantics from the lecture to coordinate use of other media. Suppose an instructor wants to distribute a handout to students. If all the students were local, the instructor would simply photocopy the worksheet and give one to each student in the class. This solution does not work if the students are in remote locations. The alternative solution is to distribute an electronic copy of the worksheet to students by making it available via the Internet. But, requiring that the instructor to give instructions about how to retrieve handouts during the lecture may be too distracting. Ideally, the instructor would only have to mention that it is time to look at the worksheet and the rest of the process would be automated. Similarly, an instructor may have a RealMedia file that she wants to play during a lecture. Rather than streaming it to multiple remote students simultaneously, it may be more efficient to have students download the entire video beforehand and automatically play the local copy when the lecturer indicates.

Our solution uses predefined configuration information to automatically initiate download of the necessary files before or at the time that the remote student session begins. In this scenario, the instructor indicates in advance the location of the information that is to be downloaded. The end user component processes the configuration information and begins download of the object, e.g. the video

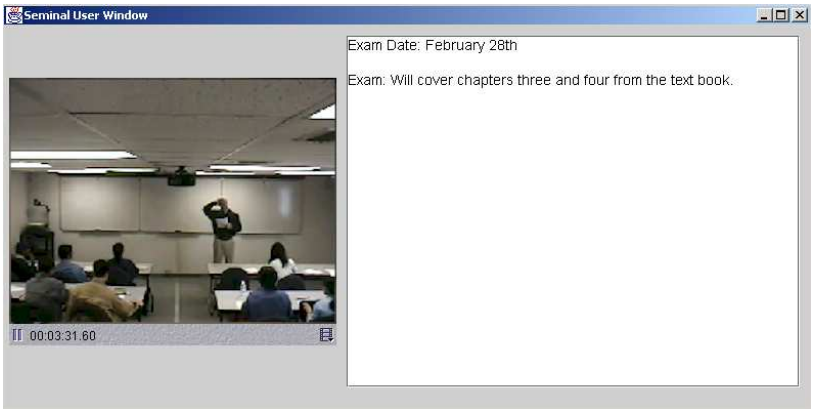

Figure 2. The user interface.

file, handouts, etc. During the lecture, when the instructor issues an activation command, i.e. indicates that it is the appropriate time to view the media, an annotation is generated. The end user component processes the annotation and displays the media using an appropriate media viewer.

There are a number of challenges in developing a distributed lecture environment. While the goal is to provide the same experience for both local and remote students, that goal is difficult to meet. In this section we have looked at three tools that use semantic information generated at the lecture site to overcome these challenges. We have found that with minimal preparation and effort on the part of the instructor, we can efficiently distribute and display media for the remote student.

5 Concluding Remarks

Many current and future applications have the goal of producing media rich environments for Internet-based, multi-user collaboration. While a lot of the basic technical support for these applications exists today, little effort has gone into integrating tools and producing more advanced technical solutions. In this paper, we have developed a model for producing collaborative applications. *Seminal* proposes the use of semantic information to provide personalized services to the end user.

Our design specifically looks at metadata extraction and user services for a digital classroom environment. While a *Seminal* implementation is largely application-specific, our experience designing a prototype has shown that we can effectively extract semantics of a lecture by using speech recognition techniques. In addition, our user service components provide more functionality than current tools offer. This is achieved by enhancing a remote user's experience with more than a simple video stream. These results are encouraging. In the near future, we hope to evaluate the use of *Seminal* for applications beyond digital learning.

References

[1] T. Yu, W. D., K. Mayer-Patel, and L. Rowe, "dc: A live webcast control system," in *Proceedings of Multimedia Computing and Networking 2001*, 2001.

[2] F. Shipman, C. Marshall, R. Furuta, D. Brenner, H. Wei, and V. Kumar, "Creating educational guided paths over the world-wide web," in *Proceedings of the ED-TELECOM*, (Boston, MA, USA), June 1996.

[3] M. Miller and W. L., "Computed web links: The cool link model," in *Proceedings of Hypertext '98*, (Pittsburgh, PA, USA), June 1998.

[4] J. Junhe, A. Jenson, and K. Gronbaek, "Ariadne: A java-based guided tour system for the world wide web," in *Proceedings of the Seventh International World Wide Web Conference*, (Brisbane, AUSTRALIA), Apr. 1998.

[5] R. Malpani and L. Rowe, "Floor control for large-scale mbone seminars," in *Proceedings of ACM Multimedia 97*, (Seattle, WA, USA), Nov. 1997.

[6] G. Abowd, "Classroom 2000: An experiment with the instrumentation of a living educational environment," *IBM Systems Journal*, vol. 38, no. 4, 1999.

[7] S. Dao, E. Shek, A. Vellaikal, R. Muntz, L. Zhang, M. Potkonjak, and O. Wolfson, "Semantic multicast: Intelligently sharing collaborative sessions," *ACM Computing Surveys*, 1999.

[8] S. Rollins and K. Almeroth, "Deploying an infrastructure for technologically enhanced learning," in *ED MEDIA 2002*, (Denver, Colorado, USA), June 2002.