2006

Making presentations web ready

Binil Kurian

Master's Project Proposal

Making presentations web-ready

Binil K Kurian
bkk8001@cs.rit.edu

Chair Person
Dr. Axel T. Schreiner

Rochester Institute of Technology
Department of Computer Science
102 Lomb Memorial Drive
Rochester, NY 14623-5608

Table of Contents

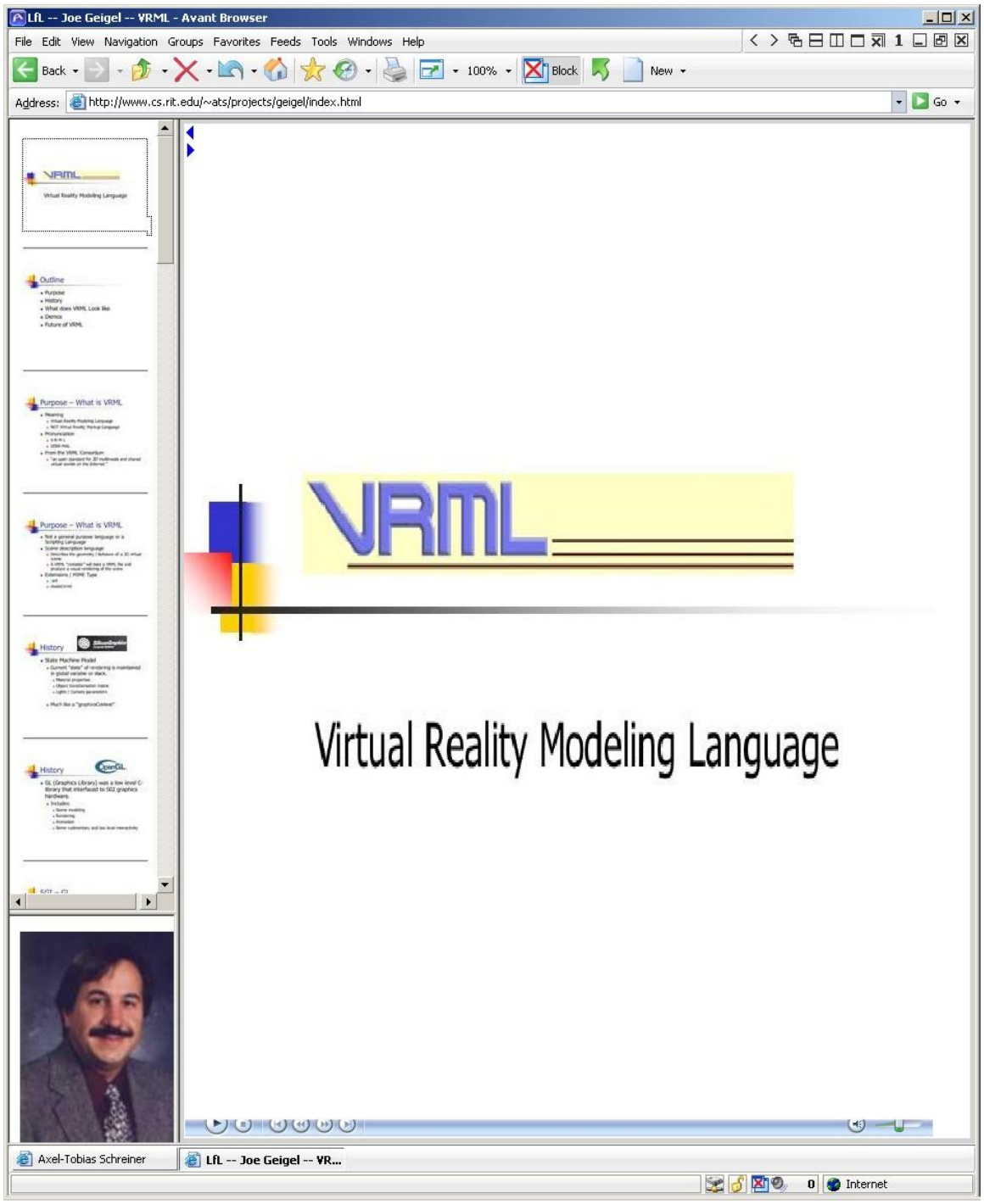
Abstract:

Presenting sound along with slides to convey more information about a presentation is not easy. Capturing the sound is easy, but giving the user random access to the sound bytes to match the slide he is viewing is difficult. If the sound bytes are sliced at the appropriate interval, and a menu is provided so that the user can navigate to any slide he wants, the presentation becomes more effective. Doing this would allow distributing the presentation for a wider audience and also for distribution using high capacity optical media. This project tries to solve this problem in a generic fashion.

Overview:

Educational institutions and corporations need employees, teachers and students to make presentations almost on a day to day basis. These presentations are made mostly in some presentation software like PowerPoint and the presenter then talks using the points given in those slides. Sometimes the presentation needs to be published on the world wide web or the local intranet for reaching a wider audience than those who were present in the room when it was delivered. This is not an easy task. Converting the slides to be put on the web is easy, but adding sound and/or visuals to give it a more realistic feel is a difficult problem. Capturing the video of the presentation and making it available is a solution. The problem with that solution is the huge size of a video file and also the fact that the user cannot jump into a particular slide or section at will. A good trade-off would be adding just sound. The sound adds more perspective to the slides than just the text on the slides. Any questions that were asked during the session are captured as well. By adding a menu based navigational system, the user can then jump to the slides s/he would like to watch.

The Geigel[1] project on Prof. Schreiner's website demonstrates the concept. The figure below gives the basic idea.



This presentation was captured during a Languages for Lunch [2] talk given by Prof. Geigel. The slides are on the left frame. The media player is embedded at the bottom. Clicking on a slide (slide 1 for example) actually plays the sound that was captured during the presentation. This enables navigating to any slide and being able to listen to the sound bytes matching the slide. The splitting of the sound bytes was done manually in this case.

The capture/splitter tool written as part of the project would make it easier to create the sound bytes to match the slides. The converter tool will convert the presentation to a web-friendly

format. The scripts would put everything together to make a sound-enabled web-presentation.

The entire presentation above, captured in QuickTime movie format comes to around 600 MB. In the web-ready format with sound, the size is reduced to 35MB. This is a huge saving. Downloading 600MB even over a broadband connection would take an enormous amount of time. That is significantly reduced by having just the presentation and the sound clips, which would convey the essence of the presentation. The presentation in this format is also suitable for CD distribution, which would be nothing but a snapshot of the website.

The real-time capture of the sound bytes and annotating them during the presentation makes it easy to slice the big sound file into little chunks to match the slides. Without the annotation, though it might be possible to slice the file based on volume levels, and sound gaps, it might not match up with the slide set. Annotating the sound bytes with the help of the capture tool helps the creator of the presentation to listen to the sound file after the presentation, and decide where the marker for the split has to be. This provides an even, smoother feel for the presentation. Assembling the individual sound files and matching slides can be automated with the help of scripts. Adding them onto a template presentation with menus completes the process of creating a sound-enabled web presentation.

Other solutions:

There are commercial software available like AnyStream Apreso [3] which captures the presentation and makes it available on the web, but these are not generic. Apreso is more or less a PowerPoint plugin and works only on the Microsoft Windows platform.

Real Networks has a product called RealPresenter [4], which offers its users a way to make presentations available as streaming media. It has the ability to take a PowerPoint presentation and add narration (both audio and video) to create a streaming media presentation that can be viewed in RealPlayer. A markup language called SMIL (Synchronized Multimedia Integration Language [8]), which has become a recommended standard for multimedia presentations is used to combine everything together. RealPresenter also has the functionality to capture and make live broadcast of presentations.

Other interesting products which do similar work are Impatica for PowerPoint [9] and Imptica OnCue [10] from Impatica Inc. Impatica for PowerPoint converts a PowerPoint presentation into a compressed format so that it is easy to stream over the internet. Impatica OnCue lets you to synchronize video and audio narration with a PowerPoint presentation. Impatica for PowerPoint does the conversion and then narration is added on top using OnCue. The final format is proprietary and can be played using a Java applet.

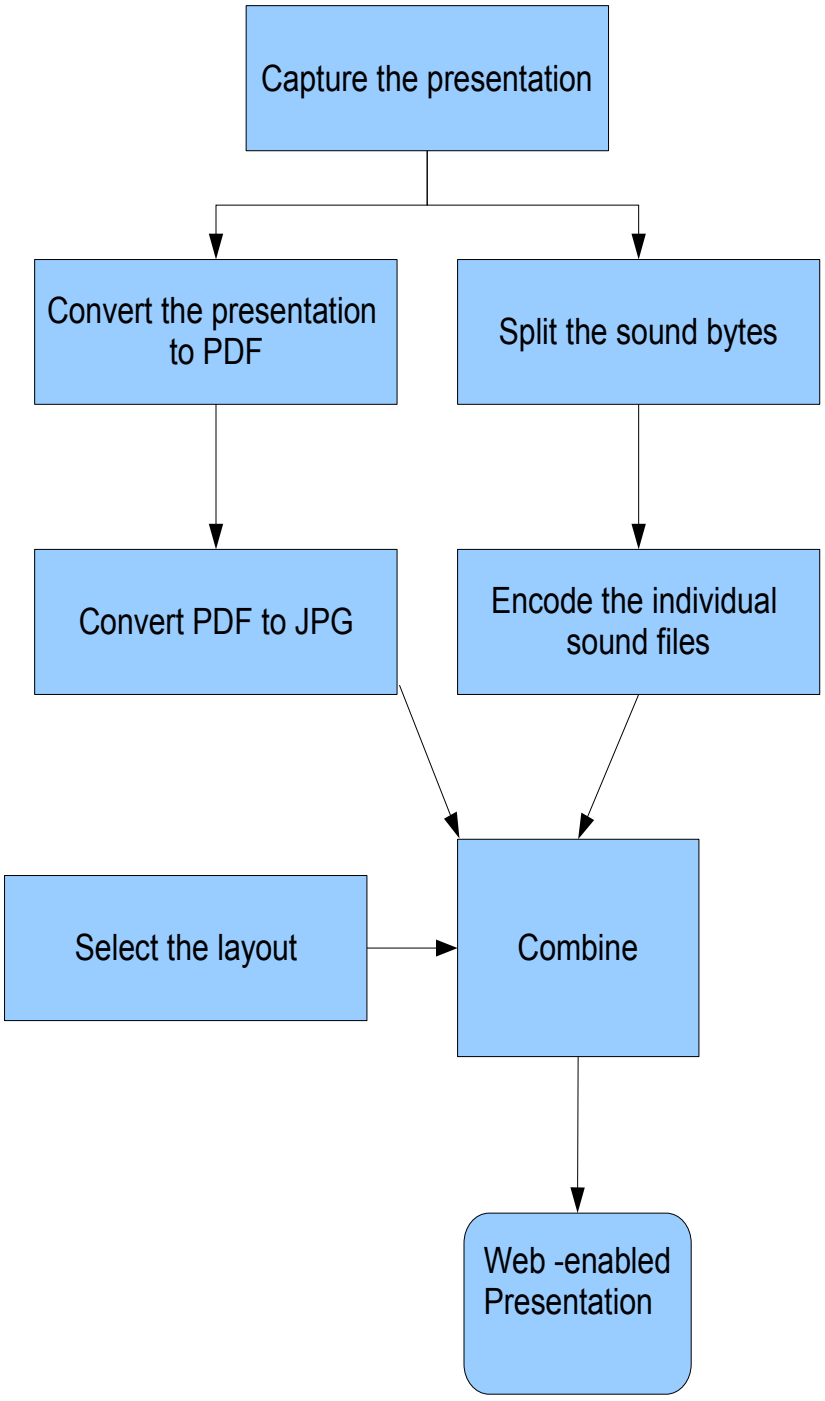
The major drawback of all of the above products is that they mostly cater to Microsoft PowerPoint, as it is one of the most popular presentation software. But support for other presentation formats like PDF, HTML should be easy to add, and there should be newer versions of these software in the future which would cater to a variety of formats.

This project will attempt to solve the problem of web-enabling a presentation in a more generic fashion.

Functional Specification:

The outline

The following picture explains the flow.


Capture the presentation
Convert the presentation to PDF
Split the sound bytes
Convert PDF to JPG
Encode the individual sound files
Select the layout
Combine
Web -enabled Presentation

This section gives an outline of the workflow given above.

1. Capture the presentation with the SoundCapture tool. The tool is a Java application.
2. Convert the original presentation to PDF (which all presentation softwares have the cabability).
3. Convert PDF to JPG using netpbm (the route is pdf->ps->ppm->jpg). This can be scripted using shell scripts. Idea is one JPG per slide.
4. Slice the presentation using the SoundSplitter tool. This is used to slice up the audio to exactly match the slide set. This uses the information captured in step 1.
5. Encode them. One mp3/ogg file per slide. SoundSplitter is a Java application.
6. Let user select one of the canned styles for layout. CSS will be used as stylesheets.
7. Use shell scripts to generate HTML which will combine outputs of steps 2-4 to create the final web-ready, sound-enabled presentation. The sound player will be a Java applet.

Sound capture

Capturing the narration can be done either before delivering the actual presentation to an audience or during the presentation itself. There are advantages for both these options. Sometimes presentations are not available (because the presenter is not accessible) before hand, hence it has to be captured while it is being presented. Whereas, if the presenter has access to this software while he is preparing the presentation, he might choose to add narration before the actual presentation is made.

The SoundCapture tool captures the narration. It uses a PC and an external mic to capture the sound. It will frequently write out the sound file, so as not to lose the whole presentation in the event of a crash. A new project is created with the name of the presentation. There are buttons to control the state transitions during the presentation. It will have the following buttons.

Start
Starts the capture

Stop
Stops the capture

Pause
Pauses the capture. Clicking Start or Pause buttons again will restart the capture.

Next
Marks a slide transition to the next slide.

Prev
Marks a slide transition to the previous slide

Start Demo
Marks the beginning of a demonstration during the presentation.

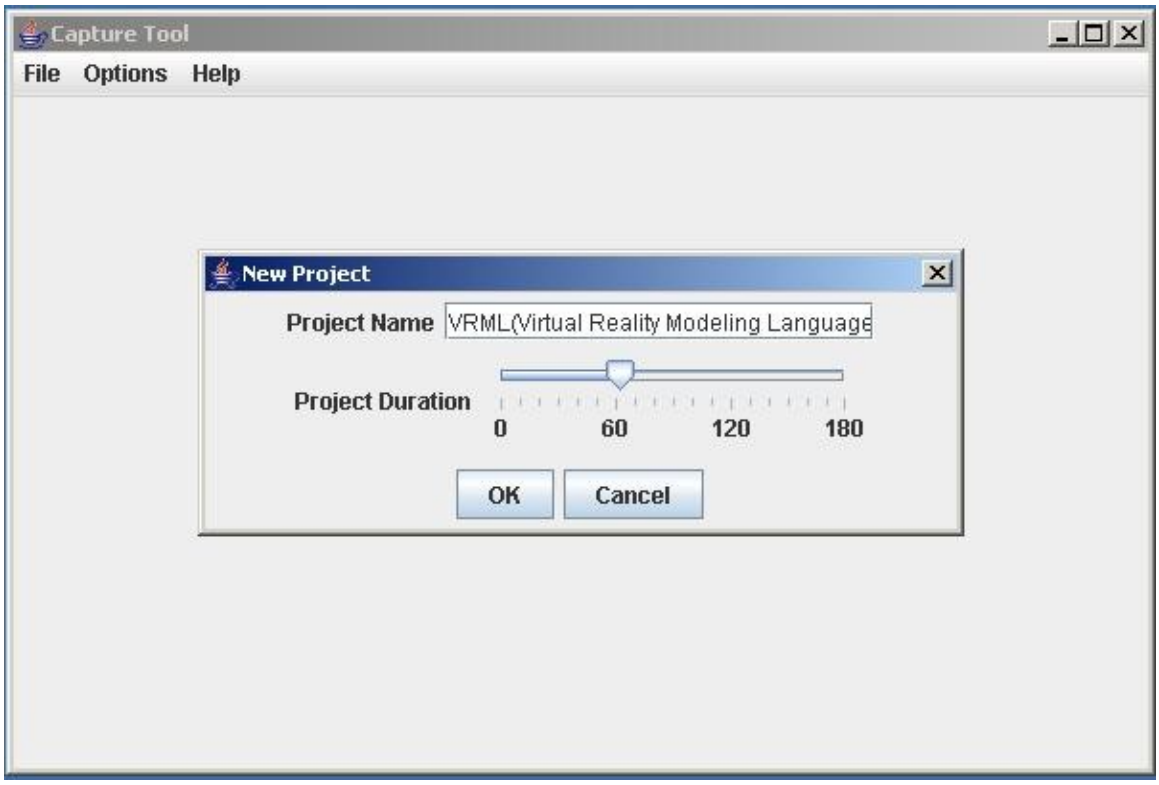
End Demo
Marks the end of a demonstration during the presentation.

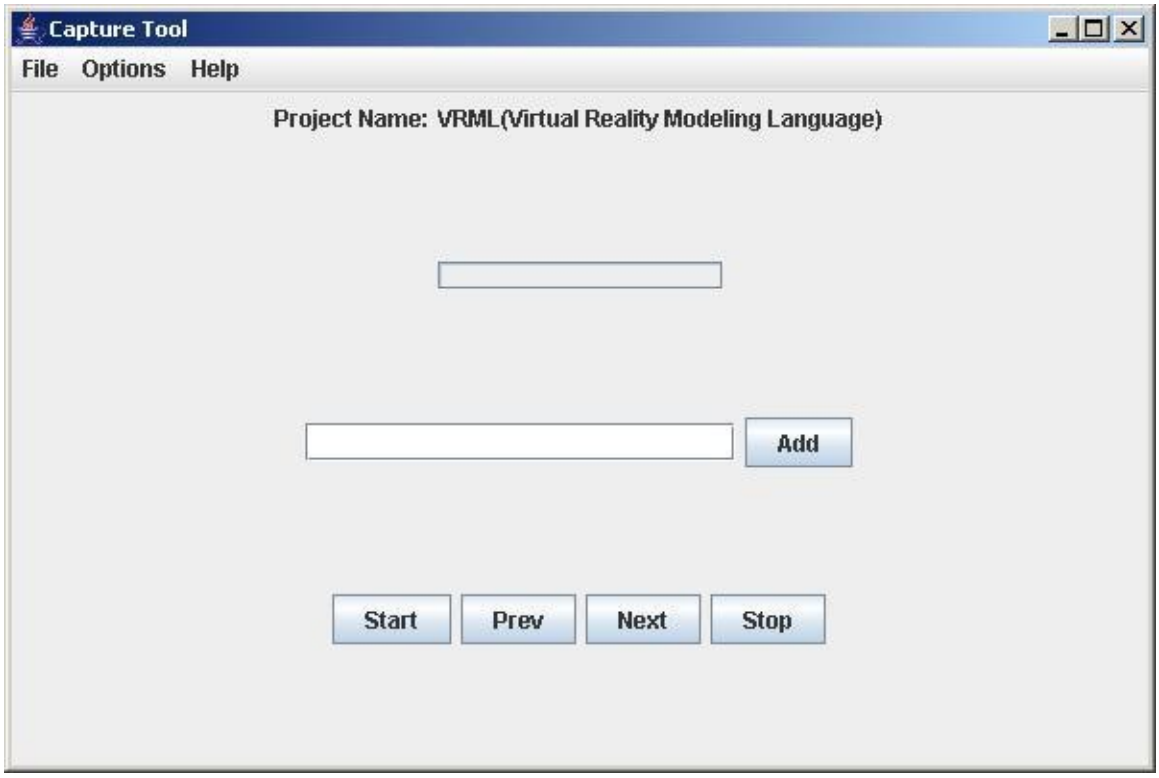
In addition there will be a text box to add additional information about the presentation during capture.

Once “Stop” is pressed, the tool writes two files. One is the audio data in WAV format,

called <*ProjectName*>*.wav*. The other is a data file which contains the state transitions. The data file name is <*ProjectName*>*.dta*. Here is a sample dta file. The numbers on the right indicate the time in nanoseconds when the transition occurred.

start=0
next=91862000000
next=155514000000
next=234016000000
data=578612000000#start demo
data=629155000000#end demo
next=638057000000
next=821752000000
data=832757000000#start demo
data=843503000000#end demo
next=2973065000000
next=3000114000000
stop=3483960000000


Capture Tool
File Options Help
New Project
Project Name VRML(Virtual Reality Modeling Language
Project Duration
0 60 120 180
OK Cancel


Capture Tool
File Options Help
Project Name: VRML(Virtual Reality Modeling Language)
Add
Start Prev Next Stop

The above images give a glimpse of the Capture tool. It doesn't exactly match with the specification above, as it is a work in progress.

Sound processing

The SoundSplitter tool takes both the WAV file and the data file created in the previous step. It then presents “part” of the WAV file as a waveform of the amplitude of the audio. It will also have the state transitions captured in the data file marked up on the waveform. This enables easy splitting of the audio file into slices. The user will be able to select any mark and listen to the audio in that region. The region size will be configurable. Then the user can mark the ends of a slice, and mark it to be ready for slicing.

After all the slices are marked ready, the user will be able to do the actual slicing procedure. The file(s) will be marked accordingly, so that the final processing script can take it and form the layout.

Sound encoding

One of the goals here will be to have the minimum sound quality that gives a good user experience. This will allow to keep the size of the files down.

The Encoder tool will be an extension to the SoundSplitter tool, and will provide the option of encoding after each slice is made. The encoding will be in mp3 or ogg. Options will be explored if a plugin-style design is possible, making this tool extensible to future formats.

Web converting

The WebConverter tool. The Open sourced netpbm package [11] allows the conversion of most popular graphical formats to the standard JPEG format which can be displayed on any web browser. The sources are available, and can be used to build this package on multiple platforms, including Microsoft Windows. These tools can then be used in the scripts to convert the given format into JPEG. During the prototype phase, a PDF presentation was successfully converted to JPEG. Given the most of the common presentations can be converted to PDF, the netpbm package can be used to convert the slides to be in JPEG.

Presenting

The presentation of the audio media will be done using an applet, which will play an audio stream which corresponds to the current slide set being viewed. The presentation will be in a slideshow format. Each slide will stay visible till the audio part corresponding to it finishes. On completion of the audio slide, the presentation will transition into the next slide and audio slice. The user will have an option of viewing thumbnails of the slideset and selecting any slide he/she wants to. If a user selects a slide, the slideshow will resume from that slide on, and continue till all the slides are exhausted.

Assembling

The assembling script needs the following inputs.

slides/ -> the converted slideset.

The slides should be named <project>-slide<N><N>.jpg.

media/ -> the converted media files.

The media files should be named <project>-audio<N><N>.mp3 or <project>-audio<N><N>.ogg.

The assembled presentation will have the following directory structure.

html/ -> contains the html which glues everything together
slides/ -> slides in HTML or JPEG format
style/ -> stylesheets
media/ -> encoded media files
classes/ -> java class files
etc/ -> miscellaneous files

The *createPresentation* script will use the inputs from slides and media directories to assemble the final presentation in the structure given above.

Design Specification:

Since platform-independence is a goal, the project will be implemented using Java. The Java Media Framework[5] provides a framework for capturing and formating various forms of media.

Since WAV is a common format and captures the audio in high resolution, the capture will be done using WAV.

If license restrictions do no prevent using an mp3 encoder, the media after slicing will be encoded into mp3. If mp3 encoding is not possible, the media will be encoded in ogg, in which case a third party library like Jorbis[6] or J-Ogg[7] will be used.

The netpbm package will be used to convert the PDF slidesets to JPEG. The assumption is that it is very easy to convert common presentation formats to PDF.

The scripts will be written in shell scripts (bash for UNIX and windows shell scripting language for windows).

Deliverables:

The following are the deliverables for the project.

- SoundCapture Tool
- SoundSplitter Tool
- EncoderTool
- WebConverter
- Applet to Play Sound in a Web browser
- Scripts to generate HTML and assemble the pieces to make a consistent presentation

Schedule:

Updated schedule can be found at : http://www.cs.rit.edu/~bkk8001/msproj/

Sr#	*Item*	*Date*	*Status*
1	*Pre-proposal*	*10/22/04*	*Completed*
2	*Proposal*	*11/19/2005*	*In-progress*
3	*Implementation*	*03/04/2006*	-
4	*Report*	*03/31/2006*	-
5	*Defense*	*04/14/2006*	-

Status:

See above. Currently the proposal is being worked on.

References:

[1] http://www.cs.rit.edu/~ats/projects/geigel/index.html
[2] http://www.cs.rit.edu/~ats/lfl/
[3] http://www.apreso.com/
[4] http://service.real.com/help/library/guides/presenter8/htmfiles/intro.htm
[5] http://java.sun.com/products/java-media/jmf/index.jsp
[6] http://www.jcraft.com/jorbis/
[7] http://www.j-ogg.de/
[8] http://www.w3.org/TR/REC-smil/
[9] http://www.impatica.com/imp4ppt/
[10] http://www.impatica.com/imponcue/
[11] http://sourceforge.net/projects/netpbm/