SMPL a Specification Language Based Framework for the Semantic Structure, Annotation and Control of SMIL Documents

Ronen Vaisenberg, Ramesh Jain and Sharad Mehrotra
School of Information and Computer Sciences, University of California - Irvine
{ronen,jain,sharad}@ics.uci.edu

Abstract

In this paper we describe the design and implementation of a declarative XML language used to support the semantic structure, annotation and control of SMIL documents. We evaluate the system in the context of distant learning environment.

We named the new framework SMPL as it intended to support the design of simple, declarative language based, multimedia document management standards, based on SMIL - a W3C standard.

SMPL is supported by all browsers de-facto. Unlike perhaps other web standards, where adding functionality has to be included in a release of a new version for a plug-in or a web browser, SMPL is extendable and yet requires no specific changes to the implementation of a plugin or the browser, beyond it's support for SMIL.

We envision SMPL as the framework for the next generation semantic Multimedia document management languages. It supports the creation of different languages that facilitate different requirements for the semantic structuring, annotation or control of SMIL synchronized multimedia documents, much beyond the distant learning context implemented as a case study.

1 Motivation and Introduction

Different needs motivate different ways for the semantic structure, annotation and control of multimedia documents. In this paper we motivate the effectiveness of SMPL in the context of distant learning environment. We identify the following need: professors would like to generate a digital version of their lectures allowing students to access the lectures from home.

The challenge is to allow students to associate a semantic meaning with SMIL[1] documents.

We create a new XML based specification language following the proposed SMPL framework which:

- **Structures** lectures in a tree based hierarchy: class-week number-slide.
- **Annotates** lectures, at particular time-intervals with links to relevant reading material, keywords.
- **Supports user control** of the lecture flow by browsing the contents tree, selecting the slide to start payback from or search for a particular keyword and play a relevant lecture from a particular time offset associated with that keyword.

For the task of presenting a recorded lecture with slide we utilize the SMIL standard. SMIL documents synchronize slides and video recordings, for each lecture.

The distant learning application is an example of a specific application need for the semantic structure annotation and control. Using our specification language framework, SMPL, we answer the application need. Similarly, different specifications can be implemented to answer different application needs for the structure, annotation and control of SMIL multimedia documents.

In the context of distant learning applications, there are existing systems that allow for similar functionality.

What sets the SMPL framework apart from traditional alternatives are the following major advantages:

- **A flexible functionality re-use framework:** SMPL framework builds upon functionality that is already part of SMIL to generate a new, specialized standard that answers semantic structure annotation and control needs.

  The main functionalities of SMIL (*synchronization* and *Content Placement*) are exploited by SMPL. As SMIL solves most of the "difficult" problems that a web-based implementation of a multimedia document

[1] SMIL[4] is a W3C standard which allows users to create multimedia presentations on the world wide web. SMIL provides presentation authors with the ability to control the timing of the presentation and placement of presentation objects in the presentation document.

player will encounter. The SMPL framework stands on the shoulders of SMIL and benefits from the large spectrum of functionalities.

The support for the structure, annotation and control is what brings the semantic meaning to the multimedia documents and SMIL presentations in particular. For example, by grouping relevant content together, annotating it with keywords and allow content based user browsing and control.

- **A single source framwork:** Any language build based on the SMPL framework is a declarative based, and source documents are editable, and describe complete multimedia documents together with their semantic features. This has a great potential for collaborative editing, as the layout, content and annotation of multimedia documents can be done by unexperienced end users, much like an HTML or a Wikipedia page is edited.

- **A semantic based specification language framework:** A specification language defined based on the SMPL framework is conceptually the semantic layer on top of a database of SMIL presentations. This generates a seperation between the presentations and the semantic layer. Different structure, annotation and control can be specified for a given database of SMIL documents and updates to the semantic layer have no effect on the presentation (SMIL) layer.

With that in mind, we design the semantic level specification language that we envision as the fundamental for a distant learning environment.

Much like an "old-fashioned" book that begins with a table of contents, continues with the actual content and ends with a comprehensive index, we define:

- **A table of contents:** correlating slide thumbnails with a specific point in the presentation, which offers content based preview of the presentation and random access.

- **Browse-able and search-able terms:** Each term is associated with a specific time in a presentation and is used as a link into a specific part of the multimedia content.

- **Bookmarks:** A bookmark is used to mark a particular position in a presentation. Bookmarks are annotated with text notes and appears on the progress bar, relative to the time-based position of the content. When clicked, it takes the presentation to the point in time specified by the bookmark.

The contribution of this work is threefold:

First we present an XML based descriptive language framework for the structure annotation and control of multimedia presentations.

Second, we design our language carefully on top of an existing descriptive language for general synchronization of media content - SMIL. By carefully we mean that no added functionality defined by SMPL framework needs anything but browser support for SMIL. This means that browsers that support SMIL can access any document from any language that was implemented based on the SMPL framework.

Third, we implement a set of tags that serve as a proof of concept for semantic annotation structure and control of SMIL presentations. Such a language was implemented and used to allow students of an Operating Systems class offered at the Winter of 2009, at UCI to semantic based browsing, access and control of a complete set of recorded lectures, and can be access at - [2].

2 System Architecture

Our key observation is that SMIL is a very powerful standard for the task of synchronization, however it lacks the semantic layer. Put in other words, SMIL facilitates the definition of what should be displayed where and when, but this simply is not enough for a real lecture management system. As long as the viewers are interested in viewing the presentation from beginning to end, or perhaps click on occasional URL's that appear synchronously, such definitions are sufficient. But, this simply is not the case.

Specifically, users are interested in being able to browse the content, see a high level semantic representation of the lecture, search and bookmark parts of it.

All of those functionalities are available with a textbook but are missing in SMIL and should be supported. One alternative system architecture is to change the SMIL specification. Unlike the synchronization and content placement problem, semantic structure annotation and control highly depend on the application at hand. And, as such, should be flexible and highly adaptable to different changing needs, and cannot be covered by a general poruses standard or wait for its partial support.

Our objective in designing SMPL framework is to stay within the boundaries of what can be achieved within the limits of what can be achieved based on what currently is supported by SMIL. The way we achieve this goal is by adding client side code that controls the execution and add the missing functionality. The execution is divided to three different states: **wait for input**, **prepare to play** and **play**.

The first state is when the presentation either didn't start, stopped, paused or ended. The second state is a transitory state, leading back to the first state if stop or pause was

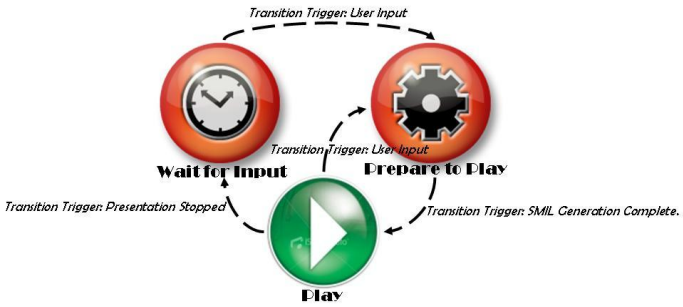

Figure 1. The SMPL Framework Execution Flow

clicked and to the third state to play the presentation for the appropriate location.

SMIL support is responsible for the play state, where a single SMIL presentation is loaded and presented. The prepare to play state loads a SMIL presentation and plays it from a random time offset, based on the user input. The user is presented with a structured and annotated user interface based on which the system receives the user input.

The following example illustrates the three states in the distant learning example: Initially, the general structure of the available lectures is presented to the user as a table of contents (wait for input).

Following user input, such as clicking on a slide in the table of contents will load (prepare to play) the SMIL presentation of the relevant lecture and seek to the time associated with that slide.

The third phase (play state) takes care of the display and synchronization of content according to the definitions in the SMIL file and the current state of the presentation.

2.1 Wait for Input State and Transitions

A static state, where the browser displays the controls and waits for the user to make a selection and trigger the execution of the presentation. The browser reaches this state either after a previous presentation had completed, the user clicked the pause or stop button or no presentation was yet loaded.

2.2 Prepare to Play State and Transitions

This transitory state is where client side code implements the missing functionality in SMIL to support the missing functionality implemented following the SMPL framework. Two main tasks are performed by the client side code at this state: first, the user input is translated to a spatio-temporal link. The specific SMIL presentation that needs to be presented is then loaded and a seek operation is performed to start the presentation according to the temporal link. Both operations are currently supported by browsers: Javascript can write HTML content to a page displayed by a browser, in such a way that the relevant SMIL content can be displayed. A seek operation is supported by all current SMIL implementations, again by Javascript.

Once the SMIL presentation is loaded and is ready to start from the correct time, the browser enters the play state.

	IE7	Firefox2	Opera9
HTML / XHTML	73%	90%	85%
CSS 2.1	56%	92%	94%
CSS 3 changes	13%	24%	19%
DOM	51%	79%	84%

Table 1. Percentage of features supported by the three main browsers of four main specifications.

2.3 Play State and Transitions

This state is no different from the regular state the browser is at while playing any SMIL document. The only difference is the ability to interrupt this state by user input, which will cause the browser enter the prepare state again.

After the SMIL presentation is complete the browser returns to the wait for input state.

2.4 The SMPL Framework - Incremental Standard Design

The main reason that very few standards are actually implemented in browsers is due to the fact that browsers struggle to stay reliable, secure, fast, and light-weight. Keeping up with the exploding number of new standards counteracts that goal. This lack of support by browsers for new types of content forces web site designers to limit their support to a specific browser or a specific technology, which is detrimental to both users and developers.

Table 1 summarizes the percentage of features implemented by three main browsers, for 4 specifications.

We designed SMPL framework such that it builds on top of an existing standard and extends it. By doing so we were able to guarantee that lack of browser support will not stand in the way of the standard being widely used or dnamically updated.

3 The SMPL Framework Implementation

The main modules involved in the implementation of a language L following the SMPL framework are: Parser, User Input Controller and SMIL manager:

- **The Parser** parses a source file based on the document definition language associated with L.
- **User Input Controller** handles all the user click events and forwards commands to:
- **The SMIL Manager** which loads the relevant SMIL object to the browser and seeks it to the correct point, according to the user's input.

We implemented the system and evaluated it on a complete recording of lectures given part of a class at UCI. The implementation exploits Quicktime's support for SMIL.

The implementation is written in Javascript, to allow cross browser and cross platform execution. The first step is to design a language for the structure, annotation and control that meets the particular needs at hand. The missing functionality is implemented in Javascript. As a first step the Javascript parses a source XML file, and implements the structure, annotation and control based on the contents of the source XML file.

The actual presentation of the SMIL documents is implemented by the browser's support for SMIL.

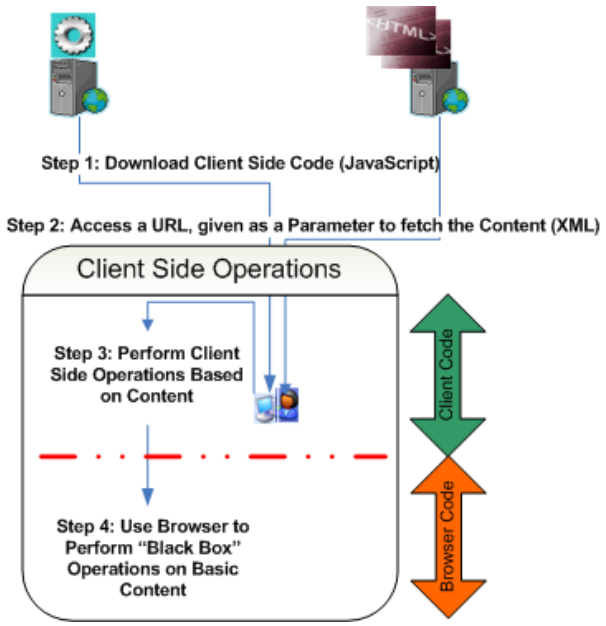

Figure 2. Implementing a DLE Semantic Specification Based on SMIL Sopport.

Fig. 2 illustrates the four steps involved in the execution, after the Javascript support for the missing functionality was implemented:

- Step 1: Download Javascript code. The code for the missing functionality is downloaded to the client machine. The code depends on the language at hand and the needed functionality.
- Step 2: Download the XML source file. This file specifies the semantic information which specifies where the relevant SMIL documents can be found and how to structure, annotate and control them.
- Step 3: Based on the client input, the relevant SMIL document is loaded and seeked to the relevant time offset.
- Step 4: Presentation of the SMIL document is managed by a SMIL "black box" support.

Steps 1 and 2 are language specific. Different languages implemented based on the proposed framework will have different Javascript implementation and source files. Step 3 is equivalent to **wait for input**, **prepare to play** states and step 4 is equivalent to the **play** state described in Sec. 2. Steps 3 and 4 are the same for all languages implemented based on the framework.

4 A Case Example - Distant Learning Environment (DLE) Implemented Following the SMPL Framework.

Fig. 3 the definition language for the DLE example and an XML source file are presented. The different parts of the language are discussed next. **Table of Contents:**

Lectures are part of a class which is part of a particular topic or department. The TOC defines the semantic hierarchy: a lecture is part of a class and a class is part of a set of classes offered at a particular department. A talk should be a complete and an independent unit, expressed by a SMIL object.

An element of the table of contents includes the link to the thumbnail to be displayed, an attribute indicating which lecture it links to, and a temporal attribute to indicate what part of the lecture. This information is used by the client side code to generate the browsable tree.

SMIL: The synchronized content of a lecture, content written in SMIL which defines the flow of execution. In order to break the linearity of SMIL, a hierarchy of timed annotations (usually in the form of thumbnails, of the presenters' slides) should give an overview of each talk. The **Preview slides** allow the viewer to browser the document and jump to interesting parts of a talk or between different talks in the document.

Bookmarks are used as a pointer to important locations in the presentation, and are used to mark different locations on the progress bar which gives the viewer the a ability to "jump" to different points in time in the SMIL presentation.

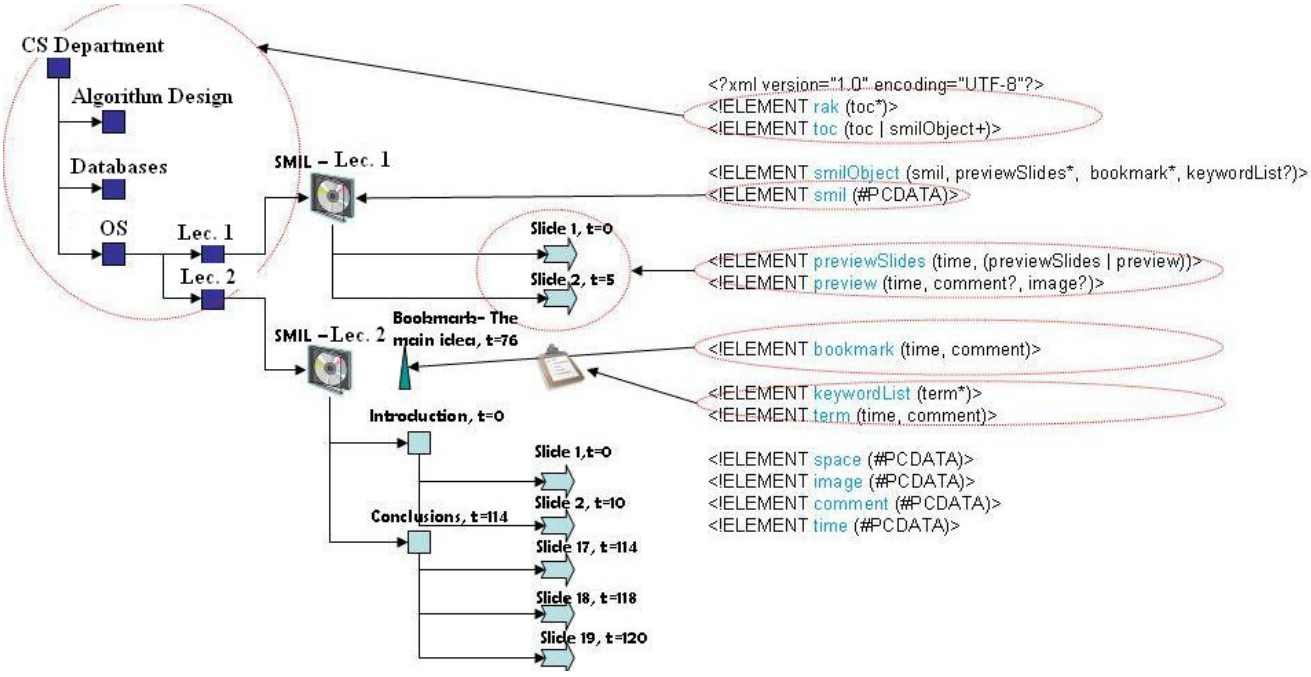

Figure 3. The DLE - Language Implemented based on the SMPL Framework. The definition language (right) and an example XML source file (left).

Keyword Lists are used when a search operation is performed. With each keyword a specific time is associated, which defines the time in the SMIL presentation for this keyword.

In order to illustrate the actual flow of execution we describe what happens in each step of our implementation according to the states of execution described in Fig 1.

- **Step 1:** Download Client Side Code

  In this step the browser will access a domain which hosts the client-side code (Javascript) that can perform the extended functionality.

- **Step 2:** Fetch the XML Content The file to be retrieved is the source XML based on the SMPL framework which has the semantic definitions that structure annotate and control of the different class lectures.

  This information will include the table of contents, and the references to the different lectures, represented as SMIL objects. During this state the client-side code performs the logic needed in order to generate a tree-like table of contents, hash table based on the terms and load the bookmarks to an in memory array. After this step the browser will enter the "wait for input" state.

- **Step 3:** Perform Client-Side Operations Based on user input. This step us equivalent to "prepare to play" state.

  This step is where the new approach demonstrates its usefulness. Since the client-side code executes within the browser, it can simply pass the complicated content, in this case the SMIL presentation, to the browser and utilize its support for a key functionality, namely synchronizing different media streams to play a presentation.

- **Step 4:** Play a SMIL Presentation

  This step is when the browser's code takes over and performs the display of the content. The browser is responsible of parsing and executing the SMIL content in a standard manner, unaware of the table of contents and dynamic nature of the enclosing application.

All of the control elements (table of contents, index and bookmarks) are implemented as spatio-temporal links that, when traversed, load the relevant SMIL object and seek it to the appropriate location. Bookmarks are implemented by regular URL linking mechanism, where two parameters: time and space are passed as URL parameters. The URL to a specific location in a presentation will have the following structure:

```
http:\\www.mydom.com\DLE_App?url=myFile.smpl&time=23&location=1
```

Where location is the index of the SMIL object and time is the relative seek in that presentation. This URL can be

saved as a regular browser bookmark. When accessed, the parameters allow the Javascipt to find the correct SMIL file and load the exact location bookmarked.

4.1 User Interface

The client is presented a system with a user interface as illustrated in Fig. 4.

A) Input box to provide search terms. B) The table of contents as specified by the DLE document. The slide thumbnails are links to particular times within a specific presentation. C) The synchronized conference presentation slide stream in the multimedia document as specified by the SMIL object in the SMPL document. D) The synchronized video stream in the multimedia document as specified by a SMIL object references by the DLE document. E) A bookmark, specifying a location in the presentation. F) The global controls, giving the audience access to important operations such as play, pause and stop. G) Search results window, containing links to different presentations and different times.

As a proof of concept, the language was also used to manage and review recordings of drills, as part of SAFIRE[1] (Situational Awareness for Fire-fighters). The task there is to visualize streams collected from multiple sensors on a limited display area. The presentation is annotated with semantic information such as tags and previews.

The lecture management language had to be extended to support stream selection as another mean for controlling a SMIL presentation. Fig. 5 illustrates the UI of the system system in a different application. The language had to be extended to support the selection of which streams to visualize out of the set of stored streams. In the example illustrated, streams 1-4 are selected out of the possible 5 streams.

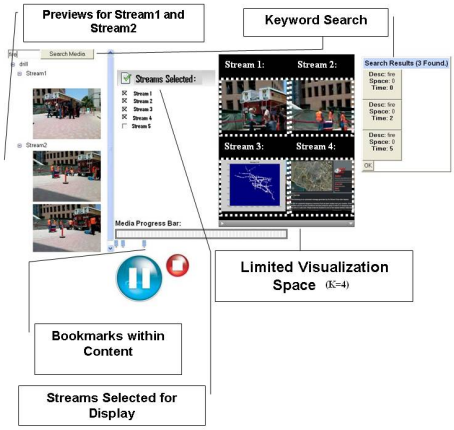

Figure 5. The DLE Descriptive Language Applied to the Application of Drill Management.

5 Related Work

As far as we know, we are the first to propose such a framework. A recent survey[9] proposed nine alternatives for extending the SMIL standard. Specifically, the authors mention scripting as an alternative to browser extensions. The authors state that scripting has the disadvantage of being hard to author by users and thus discourage the use of scripts and advocate the usage of declarative extensions instead.

Our work is different in the way that we propose to address the challenge. We use client-side code on top of existing SMIL browser support. The main difference is that the same version of the browser can access different types of content. We argue that the user does not author scripts, the user authors content that is being displayed with the support of scripting functions. This differentiation is crucial, since our new approach will access two files: one is created by the authoring user; the other is created by a developer, and its purpose is solely to support the display of the content, not the actual authoring process.

Both[8] and[5] approached the problem of implementing advanced content by building specific players, built ad-hoc for the standard.

We are interested in having support built as an incremental change on top of existing browsers.

The two main problems previously identified[7] in the context of multimedia documents are synchronization and automatic editing. We claim, that from a practical perspective, easy authoring of multimedia content and cross platform access to multimedia documents are not less important.

Significant amount of work (e.g., BIBS[10],Virtual Director[6] and Pihkala[8]) addressed the problem of automatic multimedia content synchronization for the purpose of generating a multimedia lecture document. The paper does not discuss the implementation details of the viewer or how the streams should actually be synchronized for viewing purposes after, for example, each slide's begin and end time are found.

Brotherton and Abowd[3] studied the effectiveness, by means of a user study of a lecture capturing system that enables students viewing recorded lectures. The actual implementation of the viewer is not discussed.

6 Conclusions & Future Work

We proposed and instantiated a declarative XML language to structure, annotate and control SMIL documents. SMPL is supported by all browsers de-facto. Unlike web standards, where adding functionality has to be included in a release of a new plug-in or a web browser, SMPL is used to instantiate different languages for different application

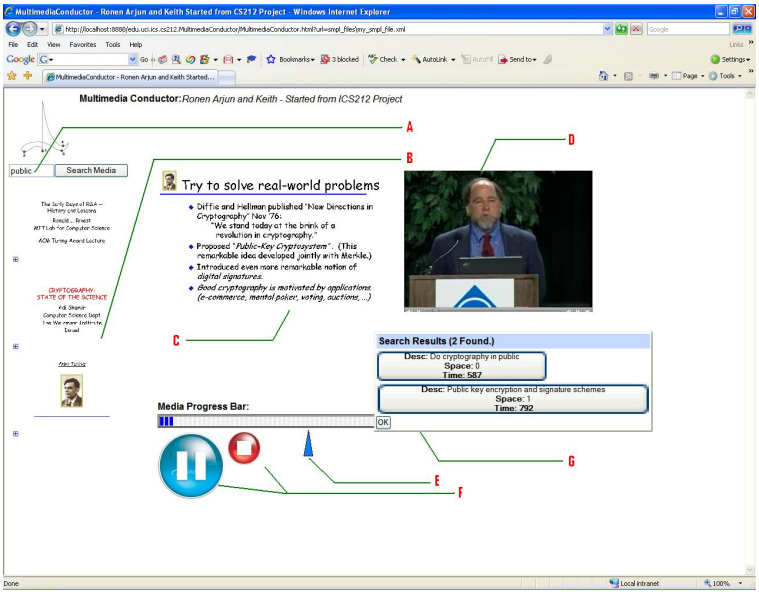

Figure 4. The UI for viewing the DLE Documents - Managed by the DLE Descriptive Language.

needs and yet is supported by all browsers, which support SMIL.

We envision SMPL as the next generation framework to manage the semantic layer of SMIL Multimedia documents.

SMPL can be exploited to facilitate different semantic requirements, much beyond the distant learning context we used as a case study.

Future work should include the implementation of different application needs based on the proposed framework to see what are the limitations and benefits, under different such application needs. Perhaps even extend the framework applicability to other standards beyond SMIL.

A wikipedia based repository of the distant learning XML documents can serve as a ground for a multimedia based, user generated encyclopedia.

7 Acknowledgments

Support of this research by the National Science Foundation under Award Numbers 0331707 and 0331690 and the Department of Homeland Security under award number EMW-2007-FP-02535 is gratefully acknowledged.

Several distinguished students contributed to the implementation of the distant learning language based on the SMPL framework (ordered chronologically): Arjun Satish, Keith A. Mogensen, Zohrab H. Basmajian, Phong Pham and Chiara Chiappini.

References

[1] SAFIRE - Center for Emergency Response Technologies, 2009. `http://www.ics.uci.edu/~projects/cert/`.

[2] StreamViewer - a Specification Language Based System for the Semantic Structure, Annotation and Control of Lectures, 2009. `http://www.ics.uci.edu/~ronen/Site/Teaching.html`.

[3] J. Brotherton and G. Abowd. Lessons learned from eClass: Assessing automated capture and access in the classroom. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 11(2):121–155, 2004.

[4] D. Bulterman et al. Synchronized Multimedia Integration Language (SMIL) 3.0 Working Draft, 2006.

[5] D. Bulterman, J. Jansen, K. Kleanthous, K. Blom, and D. Benden. Ambulant: a fast, multi-platform open source SMIL player. *Proceedings of the 12th annual ACM international conference on Multimedia*, pages 492–495, 2004.

[6] E. Machnicki and L. Rowe. Virtual director: automating a webcast. *Proceedings of SPIE*, 4673:208, 2001.

[7] S. Mukhopadhyay and B. Smith. Passive capture and structuring of lectures. In *MULTIMEDIA '99: Proceedings of the seventh ACM international conference on Multimedia (Part 1)*, pages 477–487, New York, NY, USA, 1999. ACM.

[8] K. Pihkala, P. Cesar, and P. Vuorimaa. Cross platform smil player. *International Conference on Communications, Internet and Information Technology*, 2002.

[9] K. Pihkala and P. Vuorimaa. Nine methods to extend SMIL for multimedia applications. *Multimedia Tools and Applications*, 28(1):51–67, 2006.

[10] L. Rowe, D. Harley, P. Pletcher, and S. Lawrence. BIBS: A Lecture Webcasting System. *Center for Studies in Higher Education. Paper CSHE4-01*, 2001.