Prototype
for a
Media-rich Video Annotation Tool

Philip Desenne

A Thesis in the Field of Information Technology

for the Degree of Master of Liberal Arts in Extension Studies

Harvard University

May 2012

Abstract

By the middle of this decade, two thirds of web traffic will be video. Web-based video content is steadily growing and is increasingly becoming an indispensable critical audiovisual resource for research, teaching and learning. Academic work across all disciplines now requires detailed engagement with all forms of media, of which video analysis and interpretation is a important component. Video annotation systems that allow attaching to fragments of video rich media commentaries, including links to external media resources, provide the essential building blocks for motion picture analysis.

In this project we propose the Media-rich Video Annotation Tool (MVAT), an open-source, multi-platform, stand-alone application that facilitates creating and viewing video annotations. The tool prototype attaches text, image, audio, video, image graphics, geolocated markers and compiled applications to selected segments of the video timeline, and offers an environment where annotations can be stored for future review, edited anytime or previewed by synchronizing them while playing the video.

The MVAT supplies an easy to use framework to create rather than to analyze annotations. This tool furnishes the precursor scaffolding and building blocks necessary for analysis.

Author's Biographical Sketch

Phil is Academic Technologies Senior Product Manager within the Academic Technology Group at Harvard University Information Technology. His focus is on the design and development of innovative educational technologies to enhance and support teaching and learning at Harvard. He collaborates with clients and stakeholders across the university and other institutions, including faculty, teaching staff, students, and administrative personnel to strategize, analyze, prioritize, and engineer academic technology systems. He is the founder and co-chair of the Interactive Media Design and Development user group at Harvard. Phil has over a decade of experience creating instructional applications, concentrating on the subject of media annotations.

Dedication

To all my immediate family members and close friends: thank you for your support and encouragement throughout these years.

To my coworkers in the Academic Technology Group, Harvard University Information Technology: thank you for your continued endorsement.

Acknowledgments

I thank Bakhtiar Mikhak, my Thesis Director, for sharing his expertise to improve the project. Bakhtiar's vision and constructive feedback guided me through the completion of the project.

I also thank Jeff Parker, my Thesis Research Advisor, for his thorough reviews of the manuscript and valuable editorial commentaries.

Table of Contents

List of Figures x

Chapter 1 Introduction 1

Overview of the annotation paradigm 1

Motivation for the project 3

Organization of this document 4

Chapter 2 Background and Related Work 5

Supporting Research Through Video Analysis 5

Video Motion Analysis 7

Sport Biomechanics 8

Video Ethnography 9

Analysis of Motion Picture Content 11

Supporting Teaching with Video Analysis 12

Teaching physics through video analysis 13

Flipping the Classroom 14

Enhancing Foreign Language and cultural comprehension 14

Video Production Feedback 15

Sports Coaching and Training 15

Teacher Assessment 16

Deficiencies of Previous Approaches 16

Short life span of video annotation tools 17

Discipline-specific applications....18
Limited Scholarly Value....18
Lack of support for media rich annotations....19
Closed System and Limited Interoperability....20
Few Open Source or Tools are Platform Specific....21
Summary of previous approaches....22
Chapter 3 Media-rich Video Annotation Tool....24
Media-rich Video Annotation Tool Requirements....24
Requirements for Media-Rich Video Annotation Tool....26
System Overview....29
Design Considerations....29
High-Level Design....30
Code Design....34
Design Patterns....34
Model-View-Controller....35
Data Access Object....36
Singleton....37
Proxy....37
Facade....39
Object Model....41
Model package....42
View package....43

Events package 44
DAO (Data Access Objects) package 45
Miscellaneous packages 47
Data Model 48
MVAT Graphic User Interface 49
High Level MVAT GUI Design 50
MVAT Interface details and operation 51
Authentication module 52
Video List Management View 53
The Video Annotation Editor 55
Editing and Previewing annotations 61
Chapter 4 Summary and Conclusions 66
Lessons Learned 69
Future Work 71
Glossary of Terms 76
References 78
Appendix 1: Video Clients with Annotations 83
Appendix 2: Database Table Schema Definitions 90

List of Figures

Figure 1: Horse in Motion by Muybridge 6

Figure 2: A sample screen from Yanaomamö Interactive from The Ax Fight on CD ..10

Figure 3: YouTube commentary bubbles 19

Figure 4: core system use case 25

Figure 5: core system use case 26

Figure 6: core system use case 28

Figure 7: Overall system architecture of video annotation framework.....31

Figure 8: Core video annotation architecture 33

Figure 9: MVAT interconnected design patterns 35

Figure 10: Database Proxy diagram 38

Figure 11: UML of Model-Graphic Facade.....40

Figure 12: UML of Model sub package content 43

Figure 13: Event-based flow diagram 45

Figure 14: DAO Classes and object relations 46

Figure 15: Data Model: Tables Relations.....48

Figure 16: High-level MVAT GUI design 51

Figure 17: MVAT User registration window 53

Figure 18: MVAT Video Management 54

Figure 19: Loading video and adding metadata 55

Figure 20: Annotation Edit View.....56

Figure 21: Video player time range selector detail.....57

Figure 22: Video player time range selector detail....58
Figure 23: Media annotation manager....59
Figure 24: Media preview window....59
Figure 25: Annotation mapping component....60
Figure 26: Annotation mapping component....61
Figure 27: Example of annotation edit view....63
Figure 28: Previewing Synchronized Annotations....64

Chapter 1 Introduction

> *It is possible to teach every branch of human knowledge with the motion picture.*
> *Thomas Edison (Smith, 1913).*

The Media-rich Video Annotation Tool, or MVAT is a multimedia standalone multi-platform application that enables users to create collaborative rich media annotations for videos. The application allows selecting segments of video and adding textual commentaries along with images, audio, video, maps and precompiled interactive applications. It permits editing, reviewing, sharing, and storing video annotations for future reference. MVAT provides an environment to manage annotated videos and watch videos while playing back synchronized annotations with the timeline.

Overview of the annotation paradigm

Annotating, or adding descriptive commentary notes on the margins of text has traditionally been the critical backbone of research, teaching and learning. It enables a detailed examination of the subject matter and engages an asynchronous critical discourse between scholars (Bargeron et al., 1999); the creator of the work and the annotators. At the personal level it enhances comprehension through close analysis and fosters construction of knowledge. When done collaboratively it stimulates and engages group discussion by honoring different discipline perspectives or personal points of view. Ultimately it spawns new scholarship.

Since antiquity scholars studied manuscripts or printed text by marking passages by hand, placing notes, symbols, underlines and highlights on documents. With the advent of digital text, and the rapid emergence of primary scholarly sources in other media formats, such as photography and film, the traditional annotation paradigm is shifting into a new digital realm.

In the current digital paradigm for research, teaching and learning, there is an increasing need to create, search, discover and interconnect relevant digital resources across all media. Digital video, one of the fastest growing sources of media available online, is becoming an indispensable primary resource in many of the academic disciplines.

Recent forecasts indicate that by the middle of this decade two thirds of all internet traffic will be taken up by video (Cisco, 2011). The number of video clips and hours of video footage consumed and produced has been steadily growing with the advent of self-broadcasting on-line services like YouTube. Video is becoming ubiquitous in the curricula of most academic disciplines (Bossewitch and Preston, 2011).

From the inception of motion picture in the late 1870‘s, scientists have increasingly been relying on this media to investigate or describe physical, biological and cultural phenomena that cannot be accurately recorded by casual observations. Today, humanists, as never before, have access to large collections of digitized film to explore and dissect for their critical studies. In education, online video content is now

reversing traditional teaching methods, where lectures are watched outside the classroom and focused subject tutoring is done during class time (The Economist, 2011).

Motivation for the project

Though an immense corpus of audiovisual material is available to all, scholars still face technological hurdles to engage in a critical discourse about an individual video piece. While many digital video annotation tools currently exist in the open-source and proprietary market, they are either over-simplistic or complex discipline-specific.

The simple, more popular, video annotation clients lack the tools that enable detailed examination and description of the audiovisual elements. In these, mechanisms that facilitate contextualizing or cross-referencing annotations with other relevant scholarly media are usually missing.

On the other hand, complex video annotation clients tend to function more as analytical tools rather than annotation tools. Often these tools target discipline-specific categories of annotations, emphasizing their analysis while neglecting the heuristics of simply creating an annotation.

The impetus for this project came from the need to fill the void left by existing video annotation tools: to enable collaborative scholarly media-rich annotations while minimizing the technological barriers to produce, review and manage them. Nearly a decade of annotation tool development and prototype testing in academic scenarios at

Harvard University provided invaluable application design experience background and common requirements to produce a multi-disciplinary video annotation solution.

By leveraging user-centered design principles, MVAT simplifies the process of creating video annotations while maintaining the core mission to support research, teaching and learning with videos content. It allows the commentator to select and mark a video passage for further analysis. Eventually it provides the framework necessary for expanding profound knowledge about audiovisual resources and supplements the collective academic dialogue across all media.

Organization of this document

The remainder of this document presents information that supports the ideas behind the project and explains details about the application. Chapter 2 provides background information and related work supporting the project, evidence on video analysis used in research and teaching, and exposes the deficiencies of previous approaches. Chapter 3 describes the requirements for MVAT, the high level system overview, characteristics of the code design and detail about the MVAT graphic user interface. Chapter 4 summarizes the thesis project, discusses the lessons learned and identifies future feature enhancements for MVAT. The appendices include a brief description of existing video annotation tools, database table schema definitions and a glossary.

Chapter 2 Background and Related Work

This chapter exposes the evidence supporting the need for video annotation tools in research and teaching. The first section documents the role of video analysis in research. The second section chronicles how video annotation is used for teaching and learning. The third sections reveals the deficiencies of previous approaches to video annotation.

Supporting Research Through Video Analysis

The invention of motion pictures, in the late 1800‘s, introduced a new media to support empirical research. One of the earliest forms of close analysis of sequential images can be traced to 1872 by Eadweard J. Muybridge, a pioneer in the study of animal locomotion and the creator of the first movie projector, the zoopraxiscope. Muybridge was hired by former Governor of California, Leland Stanford, to settle a bet he had about galloping horse: whether all four of a horse's hooves are off the ground at the same time at some point during the trot. Muybridge used multiple cameras to capture motion of a galloping horse (Clegg, 2007). From the resulting images he resolved Stanford’s question pointing to two successive frames in the galloping sequence showing the horse with all its hooves in the air (see frames 2 and 3 in figure 1). Muybridge continued his seminal work on animal and human locomotion publishing several folios on the subject for the University of Pensilvania (Muybridge, 1973) and extensive publications (Muybridge, 1979).

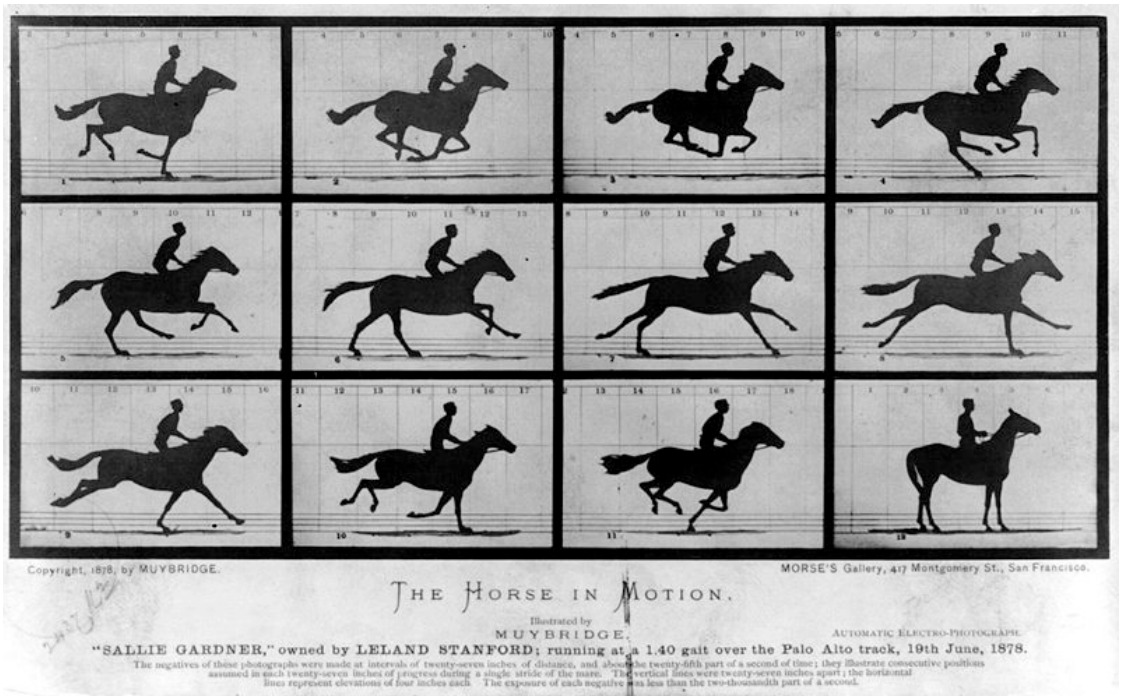

Figure 1: Horse in Motion by Muybridge

Since its conception, scholars from across disciplines have depended on close analysis of motion pictures. Today, with technological advances in high-end videography research possibilities have further expanded to both ends of the visible spectrum, from nanoparticle video microscopy (Geerts, 1991) to deep-sky video astronomy (Massey and Quirk, 2009).

This section describes use case scenarios where a video annotation tool has been used or would be useful in a research environment. The first sub-section looks at use cases for video motion analysis. The second sub-section show examples in video ethnography. The third sub-section looks at sport analysis using viodeo. The last section describes a use case in film research.

Video Motion Analysis

In many scientific disciplines, the use of videography to analyze motion and collision is well documented. New high-speed cameras, some capable of capturing up to a trillion frames per second (Hardesty, 2011), provide invaluable empirical evidence necessary for research in the study of dynamics and kinematics for fields such as continuum mechanics and biomechanics. The resulting super slow motion video produced by the cameras enable very accurate measurements of motion and collision. At the small end of the scale, video microscopy has been used in the pharmaceutical industry to study dynamics of nano-particles (Vásquez et. al., 2008). Slow motion videos of bird and insects in flight have played an instrumental role in studying complex kinematics crucial in aerodynamics used by the aeronautic industry (Valasek, 2012). At the far end of the scale, video is also used in astronomy to study the movement of celestial bodies. The study of meteors is one example where video is indispensable to record their appearance in the earth's atmosphere but also to calculate their velocity, orbit and atmospheric trajectory (Brown et. al., 1994; Molau, 1993).

A video annotation system could streamline the process of analyzing the motion of objects in space. If we look at the astronomy case of Brown et. al. (1994), a video annotation tool would support the video analysis of a meteor shower event. The tool could help users categorize and tag each meteor. Using automatic pixel contrast detection, the tool could trace the trajectory and translated to vector graphics to be part of the annotation. In a collaborative annotation environment researchers could add descriptive metadata about the meteors or engage in a scholarly discussion about the visual data, in preparation for a publication.

To further support the arguments for a publication, a media rich annotation environment allows other media to be included as part of the annotation enabling a comparative analysis of different video sources in a same space to help clarify discrepancies. Video segments of the same object taken from different perspectives could be cross-referenced to the corresponding time segments of the main video. Take for example the work by Brown et al. where the orbit and atmospheric trajectory of the Peekskill meteorite was determined from 14 video records taken from different locations.

In some cases video analysis of motion will require sophisticated software to examine high resolution video images, but even simple video annotation systems using lower resolution videos can assist researches in triaging and indexing segments of the video that can be latter be mapped to higher resolution copies. The annotation tool could then export the traced vector motion data in common file format for further analysis with other software.

Sport Biomechanics

Video motion analysis plays a critical role in the quantitative investigation of athletes and sport activities (Bartlett, 2007). Research in numerous disciplines associated with sports, from ergonomics engineering, physical therapy, podiatry to athletics, examine the details of motion related to humans. Motion video analysis software for studying sport kinematics already exists in the market (see appendix 2). These are sophisticated, sport-specific video annotation applications. They commonly feature rich graphical tools to trace and measure motion of determined activities on the

video screen, and categorize time segments of video based on the activity for that can be further quantified.

Unfortunately, commercially available motion video analysis software can be costly, platform dependent and sport-specific. Because of its flexibility, a media-rich, open source, common collaborative video annotation system could be extended and customized to serve as a motion video analysis software for any sport activity. With features such as video image vector drawing annotations, motion details and trajectory of human or sport objects otherwise too fast for the human vision to distinguish can be highlighted and labeled for further collective examination by coaches and athletes. Moreover such a platform would be broadly available for institutions and individuals from all socio-economic sectors or developing countries with limited technological resources.

Video Ethnography

Visual anthropology is concerned with the study of ethnographic film (Ruby, 1996). Long before anthropology existed as a discipline, ethnologists in the 1880 were using film as a tool of research (Flaherty, 1922). Since then, human interactions and cultural customs of people have been scientifically described and documented using film. Visual anthropologist Napoleon Chagnon, who did extensive ethnographic work among the Yanomamö indigenous tribes of Venezuela during the 60s, is a pioneer of modern annotated ethnographical videography. In the late 90‘s, Chagnon in collaboration with Peter Biella and Gary Seaman published an interactive CD-ROM about his famous film the Ax Fight (Biella et al. 1997) (see figure 2). In the CD the

"film is the center hub around which the user can organize detailed explorations by accessing information... enlargeable graphic works, including frames of the film, photographs, maps and genealogies, ... textual description and analysis of the film and essays related to the film ... biographical information about the individuals identified in the film..." (Biella, 2004).

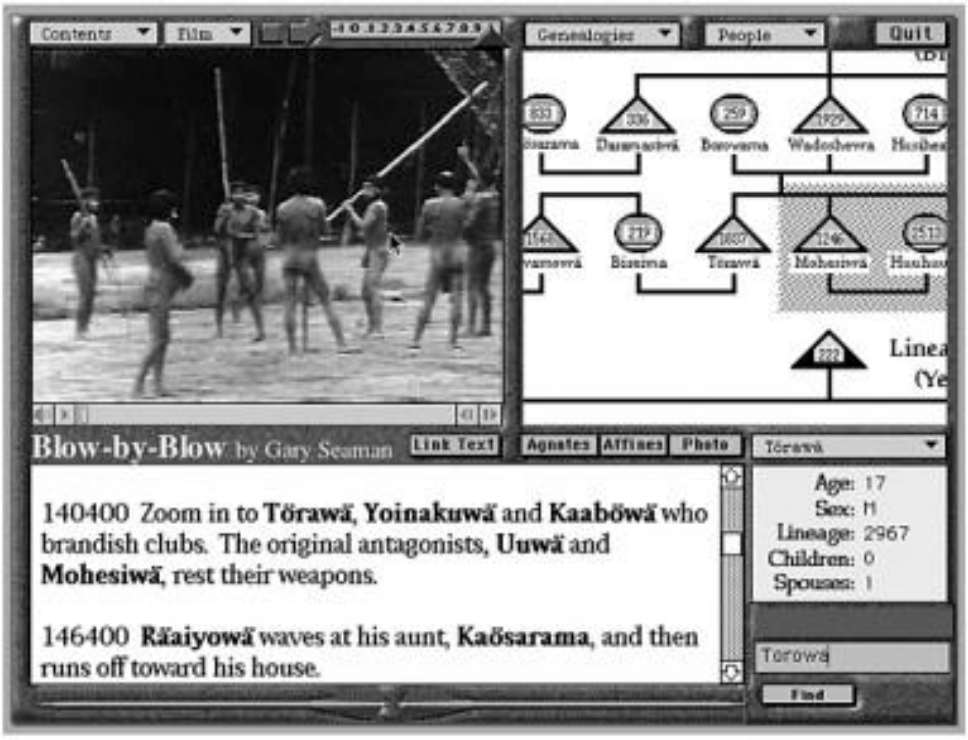

Figure 2: A sample screen from Yanaomamö Interactive from The Ax Fight on CD

Changnon's interactive CD demonstrates the richness and depth of the discourse that one can achieve using contextualized multimedia. Such a case is a exemplary use case testimony for a video annotation system. But a work like this CD is unique. The effort to produce such an interactive piece is costly and requires skilled application developers and instructional technologists to piece together the information. Moreover, the information presented is encapsulated and static with no

option to expand the discourse. An open collaborative media rich video annotation system would be capable of fulfilling the requirements of an interactive application like Changnon's and more. By breaking the production barriers it would enable any visual anthropologist or amateur ethnologist to produce similar pieces of interactive media at a fraction of the cost of an interactive CD-ROM like Chagnon's. At the same time because of the collaborative nature of the annotation system, it would allow other scholars to continue the disquisition about the piece over time.

Analysis of Motion Picture Content

Since the early years of Hollywood, and especially during the cold war era, the detailed analysis of motion picture became an obsession for the federal government. Fearing the influence of movies on the national morale (Digital History, 2012), the government created the Office of War Information (OWI) to supervise the film industry. In Hollywood, OWI's Bureau of Motion Picture had a special Motion Picture Analysis Division established primarily to conduct surveys of movie content to root out communist propaganda. Back in the early 40's, Dorothy Jones, then chief of the division, published the document on "Quantitative Analysis of Motion Picture Content" to "provide an instrument capable of measuring the scientific exactness of the content of each motion picture as it is released" (Jones, 1942).

Nowadays the analysis of motion picture content continues be of significant value for film industry, the focus shifting form war propaganda to motion picture rating and marketing research. The rating classifies films based on quantitative studies of the appearance of issues concerning profanity, violence , sex, substance abuse,

racial issues, etc. Marketing research in broadcasting, film and cable industry focuses on product placement, analyzing the occurrence of branded goods, out of formal advertisement context, within film, music videos and television shows.

Film studies and communication research study content of motion pictures from a socio-cultural and historical perspective. They perform critical explorations of the narrative or detailed dissections of scenes or character interventions, dividing the content into measurable units of analysis (Lorimer 1994) such as spoken words or type of scene and calculating the occurrence of these. Through this analysis they are then able to infer about the meaning of salient themes or other relations.

Film industries, marketing research, film and mass communications all share a common need to perform detailed analysis of content. A media rich video annotation system would serve the research needs of all, enabling efficient categorization of selected segments of video through custom tagging, effectively creating quantifiable units of analysis that can be easily exported for further examination. The annotations system would also enrich the depth of the analytical comparative discourse by allowing relational cross-linking of relevant media resources to each unit of analysis.

Supporting Teaching with Video Analysis

Motion pictures have been used for teaching and learning since the birth of the technology. Modern teaching practices now make extensive use of video to “flip” the classroom. Students can review full lecture videos or clips of lecture topics, visualize classroom cases, reexamine lab practices, learn languages outside of class while dedicating time in class to focus on specific issues or doing exercises. But unlike in a

live classroom, the delivery of the content via video induces passive learning. Because of its unidirectionality, it doesn't allow students to participate and collaborate with peers by enriching the discourse with their perspective or make inquiries about the material being delivered. Video annotation tools in pedagogy introduce a new model of engagement with the video material transforming the discourse into a bidirectional exchange. By being able to add questions about, add enriching commentaries or clarifying marginalia to segments of the video content, the student becomes more engaged.

A video annotation tool can also benefit teachers, not just students. At the personal level it helps the instructor document and support self-analysis with verifiable evidence. Collated video annotation data over a given time period offers the potential to review and examine changes in development of teaching practices (Rich and Hannafin, 2009). Through student commentaries teachers can assess the level of engagement with the video material, by individuals or by the entire class. Teachers can also take note of specific topic areas where students are having difficulties to later address the issues in person or by embedding new reformulated material as commentaries within the video. Teachers can further reflect on their own teaching methods based on the feedback gathered though student commentaries or questions about the material.

Teaching physics through video analysis

Visualization of physical phenomena through video analysis can help students better understand physical concepts. Motion experiments of projectiles, oscillations,

collisions, rotations and Brownian motion have been studied through dedicated video analysis software (Brown and Cox 2009, Laws and Pfister, 1998). Using software to conduct videomeasurements students in physics at Safarik University in Poland performed motion analysis of common objects (Ješková & Kireš, 2007). Studying real life situations close to the students everyday life through video, such as transportation vehicles, rockets or athletes, enabled students to perform analysis without complicated external measuring apparatus – wires, sensors, etc.– and obtain quick and relatively accurate data of multiple object at once (Jeskova and Kires, 2007).

Flipping the Classroom

Professor Gary King, who teaches an advanced course on statistical analysis at Harvard University, uses a video annotation tool to engage students in active learning with online lecture videos. He assigns his students to closely watch prerecorded and annotated lecture videos before the class meets and asks them to comment on video segments where lecture topics might be confusing or need further explanation. Professor King then reviews the student contributions and replies to questions with media rich notes, links to additional resources, statistical code samples, or direct citations to bibliographical records. If a particular topic has been lacking confusing to more than one student he will address the issue during class and engage students in a discussion to clarify the subject.

Enhancing Foreign Language and cultural comprehension

Romance language instructors at Harvard University use video annotations to teach language and cultural comprehension. Instructors assign students to transcribe or

translate conversations by the actors in foreign language movies using simple textual video annotation features. The instructors then review the student contributions and asses the level of comprehension. With a media rich video annotation tool, students could record or upload their own audio or video commentaries thereby providing personal audiovisual pronunciation evidence that can't be captured in a textual form.

An open collaborative video annotation system could be used to crowd-source the translation of films, documentaries, operas, etc, into multiple languages. This would help break language barriers and increase cultural knowledge through videography.

Video Production Feedback

A professor in a documentary film production course can offer frame-by-frame feedback and editing recommendations to students. With the ability to draw simple shapes such as a rectangle on top of the video image the instructor can demonstrate how to reframe the image for a more effective visual impact. In addition, the instructor could include external media clips or fragments as part of the annotations to show an example of camera or film editing techniques that could be adapted by the student.

Sports Coaching and Training

As for research, video annotation systems for coaching and training are well documented (Zhai. et all, 2005; Buchannan, 2009; Cherry et al., 2011). The sports industry is well equipped with video analysis tools (see appendix 1) for coaches to review plays or examine the rendition and efficiencies of athletes or entire teams after practice and games. The video analysis tool are also available for an individual to

perform self-analysis focusing on kinematics to improve their game or condition by correcting inefficient postures or movements (Kitler et al. 2001, Colasante, 2011). By adding notes and drawing on the video frames to highlight direction of movement or forces, athletes and coaches can focus on details of movement, track progress and offer a different perspective to help improve performance.

Teacher Assessment

Interest in video annotation and the teacher assessment seems only to be increasing. At California State University, teachers are assessed via collaborative video annotation tools. As part of the process of licensure or board certification, members of the board are asked to review a sample lecture video given by a candidate and add their commentaries at precise moments of the recorded lecture. The video notes by all the assessors are later collated and weighed in the final evaluation of the teacher.

The previous two sections exposed use case examples of how a media-rich video annotation system can add value to research and teaching. The following section describes how previous approaches have been insufficient in addressing the requirements for annotating videos across disciplines.

Deficiencies of Previous Approaches

In the days of film celluloid, the annotation of motion picture was a relatively tedious manual process, it involved handling of film reels in a specialized manual film editing station, where single frames could be analyzed under a magnifying glass, and commentaries about them recorded on paper by referencing the distance measured

from the beginning of the film or in some cases the frame number. With the evolution of digital video, the paradigm of video annotation changed dramatically. Now referencing a location in the video with time stamps or frame numbers and analyzing a single video image or multiple superimposed frames can be a simple process.

A review of published research on video annotation and video annotation tools tools reveals that over the last decade, numerous approaches have been proposed, ranging from video annotation system design and analysis specifications (Harrison and Baecker, 1992), patents for method and apparatus for annotating full motion video (King and Melis, 1994), to highly discipline-specific standalone video annotation software (Buchanan, 2009; Hagedorn et al, 2008; Kounoudes et al, 2008; Zhai et al. 2005). Some studies have conducted extensive comparison and reviews of video annotation tools useful for research, teaching and learning (Rich and Hannfin, 2009; Chaudhary, 2008), highlighting strengths and weaknesses of each system. Further research discovered video annotation tools, developed as prototypes within educational institutions, that are seldom mentioned in a publication or have yet not been published.

Short life span of video annotation tools

A common trend among video annotation tools is their ephemeral nature. Of the recently published video annotation tool, several were no longer available. For example, since Rich's and Hannfin's review in 2009, tools such as VAST or MediaNotes have already disappeared from circulation. The reasons for the short life span of many video annotation tools can be traced to lack of funding, unsustainable

revenue models or rapid changes in video format technology such as compression codecs. Rapid change will dictate the developments of video annotation tools for years to come. With video annotation tools increasingly relying on video delivered over the World Wide Web, the stability of current and future video annotation tools depend largely on the stability and industry agreement on standard video formats and codecs (Pfeiffer, 2009). Until the video format evolution curve levels out or stabilizes we can expect more video annotation tools to fade.

Discipline-specific applications

Most of the video annotation tools reviewed in the literature were built to satisfy discipline-specific requirements. Some were created for teaching and self-reflection (Rich and Hannfin, 2009) others were designed for psychological and ethnographical studies (Hagerdon, 2008). Interestingly and though seldom mentioned in the literature, those that target sports activities are some of the more sophisticated graphic rich video annotation tools available in the market (see appendix 1) (Buchannan, 2009; Zhai, 2005) and contain features that could satisfy the requirements of other disciplines (see Kinovea, 2012).

Limited Scholarly Value

Given their ubiquity, popular video players such as YouTube, Hulu, Vimeo have the potential to become useful scholarly video annotation tools. Annotation features in YouTube that were designed primarily for marketing purposes, to synchronize display of advertisements messages, or for entertaining social commentaries, to add callout message bubbles on the video image (see figure 1) could

easily be repurposed for research, teaching and learning. But for these annotations to have a scholarly and pedagogical value, they need to be indexable and allow to link to external resources. Currently annotations are not indexed by Google for public consumption, nor can they include a clickable link outside the Google Video Network. Moreover, the annotations can neither be aggregated nor exported to be analyzed in different contexts. However Google most likely indexes user contributed annotations for internal use, mining them for consumer profiling and lucrative contextual advertisement.

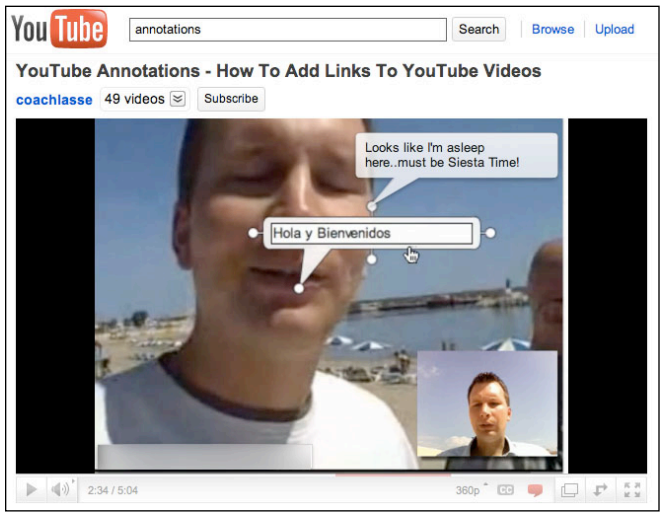

Figure 3: YouTube commentary bubbles

Lack of support for media rich annotations

The majority of video annotation systems reviewed lacked support for multimedia annotations. The main featured shared across all video annotation tools is the ability to add textual commentaries or labels and tags to specific points or segments of the video. Few allow media rich commentaries, where an image, video, audio or other media can be associated with the annotation. Tools such as VideoPaper

or Harvard's Collaborative Annotation Tool (see appendix 1) permit images and plain html as part of the annotation but not much else. Video annotation tools for sport activities, on the other hand, have great graphic drawing tools for highlighting and marking the video image but other than textual notes they lack multimedia notation capabilities.

Media rich annotation are important for ethnographic studies. Yet frameworks such as VCode and VData which are described as valuable for ethnographic studies are missing these features. If we look back at the interactive ethnographic piece by Chagnon, we can clearly identify the need to integrate other media. The addition of photographs, maps and genealogical graphs help to better understand the narrative and aid in determining the causes of the interactions by describing biographical information about the characters and how they relate to each other.

Closed System and Limited Interoperability

Unfortunately the majority of the video annotation frameworks reviewed are closed with limited interoperability. Several promising tools that were built under the auspices of academic institutions are either locked and accessible only from within the institution LMS (see in appendix 1 VITAL at Columbia, Multimedia Annotator at UW or CAT at Harvard) or have a poor support model for deploying them – with little or no instructions available. In the case of proprietary applications, user contributed annotations can be reviewed and sometimes analyzed within the framework, but they can rarely be exported for further examination outside of the framework. The

annotation data is typically locked within proprietary databases and difficult to access (YouTube, VideoAnt, appendix 1).

Interoperable annotation tools would allow the annotation data to be viewed in other video annotation tools, taking advantage of annotation analytical features that might have been missing in the tool where they were originally produced. In the text editing world this would be equivalent to using a simple text editor and opening the file in a feature full text editor as Word. Moreover, if the tools are interoperable, the annotations used by a teacher in one institution could be ported or shared with instructors using a similar tool from another institution.

The lack of interoperability in most of the reviewed tools is understandable. Presently there are no guidelines to describe video annotations in accordance with W3C standards. If there are no annotation standards then there is no foundation on which to adhere a structured API for diverse annotation systems to interoperate. Fortunately, a newly created W3C Open Annotation Community Group is working towards a common, RDF-based, specification for annotating digital resources, merging two existing competing data annotation models: the Open Annotation (OA) (Sanderson and Van de Sompel, 2012) and AO (Annotation Ontology) (Ciccarese, 2012). A common annotation data model could serve as the backbone of an annotation data exchange mechanism.

Few Open Source or Tools are Platform Specific

Very few of the tools reviewed were available as free, standalone applications or as un-compiled open source code. The VCode and VData was the only open source

project available for download as a compiled application and with full source code files. The Kinovea video analysis application is also free, but distributed as a precompiled file. The drawback of these offerings is that the tools are platform specific, therefore limiting adoption. The system requirements for VCode and VData is Mac OSX, and Kinovea is only available for the Windows platform (see appendix 1). Some tools such as VideoAnt are cross-platform web-based applications, also available for free but the application and frameworks remains locked inside the institution where it was created.

For a tool to be widely adopted in a scholarly community, it would need to be freely available and functional across multiple platforms. Ideally it should be part of an open source project supported by an active and committed developer community. An open source framework could enable custom deployments of the tool within an institution and would allow the option of expanding the set of feature sets such as adding new annotation media types, video analysis and diagnostic tools or embed applications for custom processes. Ultimately the tool would be enhanced and its could potentially be extended though regular updates provided by the open developer community.

Summary of previous approaches

The study reveals that not single tool meets the requirements for multiple disciplines. We discovered considerable feature deficiencies throughout all video annotation tools. The majority fall short on meeting the requirements that can serve both equally humanists and scientists. Either the annotation tools are over-simplistic or

they are too complex. The tools predominantly share a way to label or add simple non formatted textual notes to the video timeline, but very few let you attach media-rich, hyper-textual notes, nor highlight the video image as part of the annotation. Some of the more advanced video annotation tools try to encompass annotation creation and video annotation analysis inside the same application, sometimes falling to fulfill the requirement of either.

Interestingly, a few of the outstanding tools complement each other, and if their features were to be combined, as in the case of VCode/VData and Kinovea, a hybrid of both would make an ideal multipurpose scholarly tool.

Chapter 3 Media-rich Video Annotation Tool

This chapter explains the details of how the MVAT is constructed. The first section documents the requirements gathered to build MVAT. The second section describes the high level system overview. The third section looks at the code design characteristics and programing approaches. The fourth section give a run down of MVAT graphic user interface.

Media-rich Video Annotation Tool Requirements

The requirements for the video annotation framework were collected from real use-case scenarios (as described in chapters 3 and 4), interviews with individual scholars, teachers (Annotations at Harvard, 2012) and targeted focus groups with interest in video analysis. The requirements were refined through an iterative prototype development of video annotation tools deployed in undergraduate course websites in sciences and humanities at Harvard University over the course of several years.

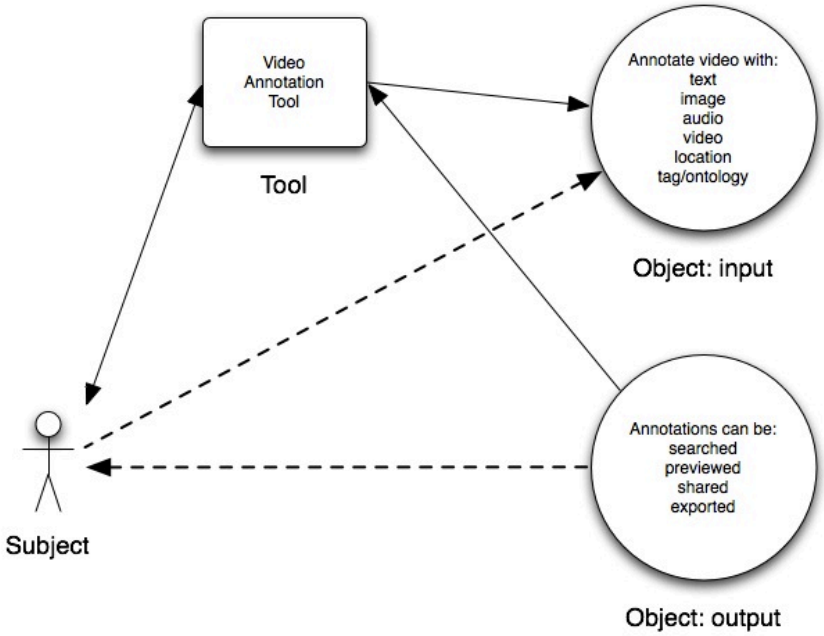

Figure 4: goal-directed actions of MVAT.

The collected use-case data on video annotation was evaluated and the interactive requirements abstracted and broken down into actions and further subdivided into operations. These categories provided the foundation necessary for understanding the steps a user needs to perform a video annotation task. Based on the principles of Activity Theory (Wikipedia, 2012) the goal-directed actions are described at the conceptual level between the subject, the object and the tool (see figure 4). In this context we discern two separate triad of interactions where one is an input process: the object being the user creating the annotation. The other an output process: the object is the user reviewing the annotation.

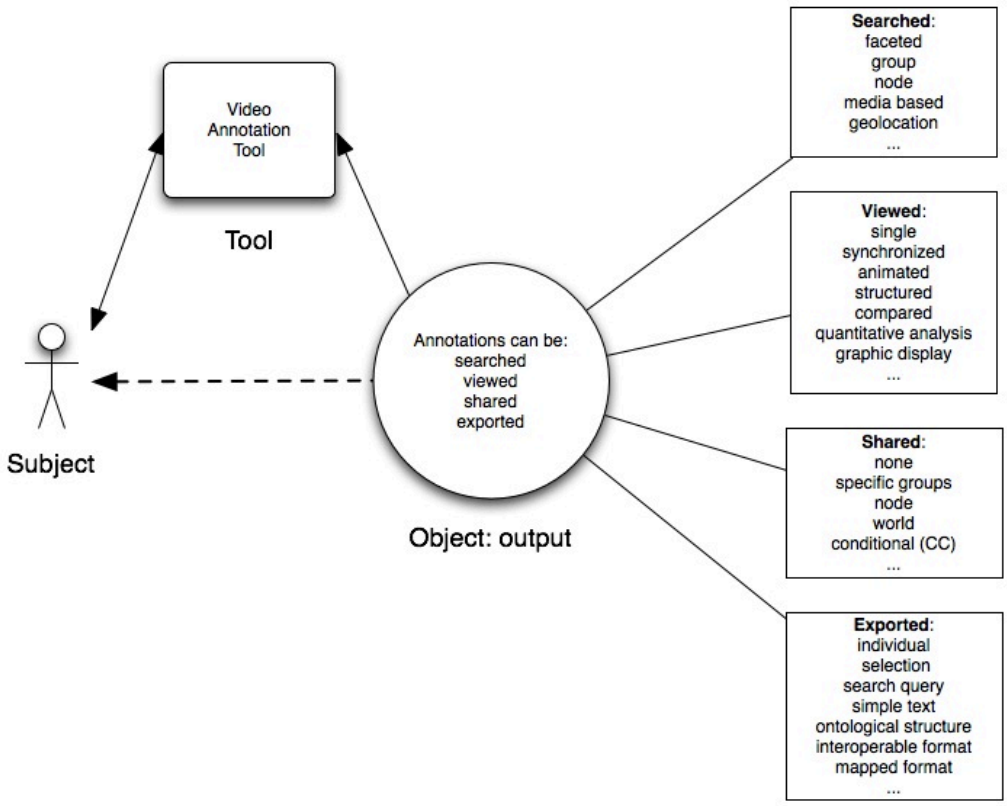

Figure 5: core system use case

Figure 5 illustrates how the task activity is subdivided into the units of operation. Here the output object is split into the search, view, share, and export components of operation. For a more granular comprehension of the interactivity each one of the components is further separated into its constituent sub tasks (see rectangles on figure 5).

Requirements for Media-Rich Video Annotation Tool

The system should allow the user to:

- Deploy the tool in any platform (Windows, Macintosh or Linux)
- Register, or authenticate into the system as an existing user
- Create a list of videos to be annotated from local or online media

- Pick a video from the list and view it
- Add media-rich commentaries to any fragment of the video timeline
- Highlight regions of the video image by drawing or adding simple colored geometric shapes
- Attach tags or keywords to the annotated segment,
- Georeferenced and visualize locations associated with the annotation on a map
- Define privacy settings for each annotation (private, public or shared with predefined group)
- Search and display publicly available annotations on the same video
- Export individual or collections of annotations in common exchangeable file formats (txt, xml, html, etc.)

The video annotation tool should be a stand-alone application deployable in any of the common platforms, Widows or Macintosh, and allow the user to work on line or off-line with video media. The core system use case for a video annotation tool is represented in the diagram of figure 6. The tool allows the user to authenticate into the system, add a video or select one from a predefined collection and create media-rich video marginalia to any segment of its timeline (see figure 6). Besides simple textual notes and tags, the annotation can include images, playable audio and video files, georeferenced map locations, and encapsulated interactive applications.

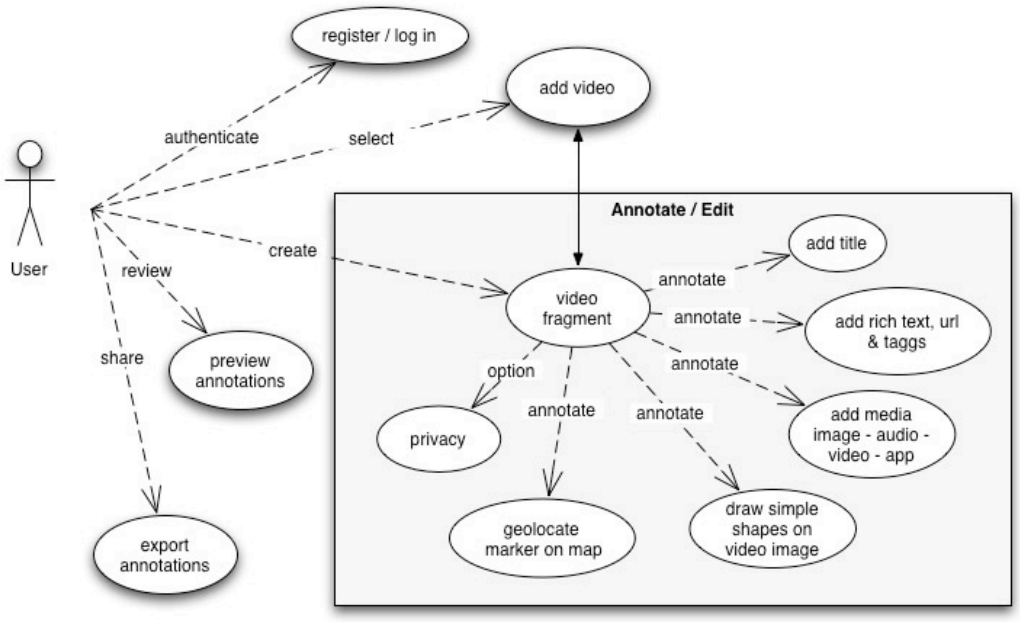

Figure 6: core system use case

Analogous to highlighting or marking text on a document, the tool also enables the user to draw simple shapes to areas of the video image as part of the annotation.

To facilitate reviewing and editing, the user has the option to view the annotations by playing them back synchronous with the playhead of the video player.

Finally the user can further decide to keep the annotation private or share it with the world, and even export the content in a common exchangeable format for further analysis or distribution.

The online database will need to expose an open API that will enable open access to the annotation database and exchange information through interoperable web services using W3C annotation ontology standards.

The tool will need to connect to an online database to synchronize local annotations with the server and download any corresponding new publicly accessible annotations.

System Overview

The Media-richVideo Annotation Tool (MVAT) prototype is a stand-alone Adobe Air application. This chapter describes the software architecture of the existing MVAT. The first section exposes the considerations that influenced the choice of the components and technology that comprise the application. The following section explains the high level design, describing the components of the application and their relationship.

Design Considerations

The purpose of MVAT is to provide an intuitive platform for users to manage a list of videos, play any video from the list, select a segment of the video, annotate it with media-rich content, and view all annotations synchronized with the video playhead. The application needs to be able to access videos and other media from the local file system and from the World Wide Web. The user needs to be able to view and annotate videos even when not connected to the World Wide Web. Annotations need to be exportable in a common file format for archival or further analysis. The design considerations for the application include: ease of use, ease of distribution, compatibility across platforms, application portability, scalability, and interoperability.

Based on the above specifications, a hybrid system developed in either Java or Flash platforms, seemed like the best option. Given the development expertise in Flash and the ubiquity of the technology, a stand-alone Adobe Air application built on the Adobe Flex 3.0 framework was chosen as the most appropriate solution.The Flex framework along with the Flex Builder Eclipse base IDE, offers a convenient development platform for rapidly prototyping rich user interfaces applications. It has the advantage of maintaining one codebase that can be purposed for deploying applications in multiple devices independent of the OS. The Flex framework includes common UI components, native extension libraries, straightforward media connectivity, embedded SQLite database and diverse third party API's. Moreover, Adobe Flex/Air online resources are well documented and the developer community is made up of a vibrant supportive online collective with an open source code sharing spirit, eager to offer help.

High-Level Design

The main component of the system consists of a client-side stand-alone video annotation application equipped with it's own database to store text annotations (see figure 7). All media files that are used by the application remain external, and are located via a local file path or URL. The application can optionally connect to an external online database to synchronizes local annotation data and publicly available shared data. The application is capable of exporting annotation data in a text file in the local file system.

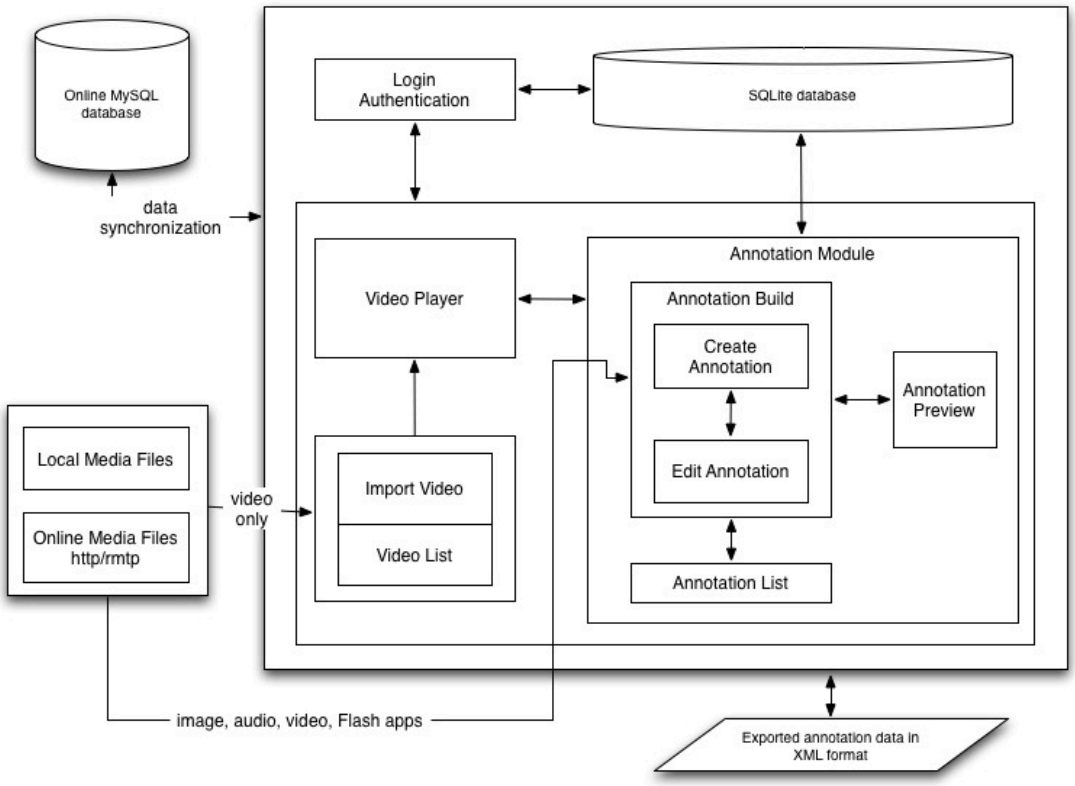

Figure 7: Overall system architecture of video annotation framework

The client-side application contains:

- An authentication module that allows new users to register and existing users to authenticate by loggin in with their registered password
- A local encapsulated database to store user information and annotation data
- A video list module where users can add videos from their own file system or via URL
- An advanced video player where users can preview the videos from the list, or trigger an annotation by selecting time ranges directly on the scrub-bar (range slider UI) video control.

- An annotation build/edit module where users can create media-rich annotation, edit existing annotations, and preview annotations synchronized with the playhead.
- An interactive annotation list module where personal and public annotations are listed and can be selected to preview the annotation content in the annotation build/edit module.

Annotations can be reviewed in the Preview module by playing them back synchronized with the video playhead. The system can back-up the annotation data to an online database automatically or manually. Annotation data can be exported in a standard common exchange annotation format.

At the core of the video annotation tool structure is the annotation build/edit module (see figure 8) which consists of a stack of independent sub-modules that allows the user to associate one or more types of media with the selected video segment to be annotated. Each module can be independently activated at compile time depending on the complexity of the annotation needs.

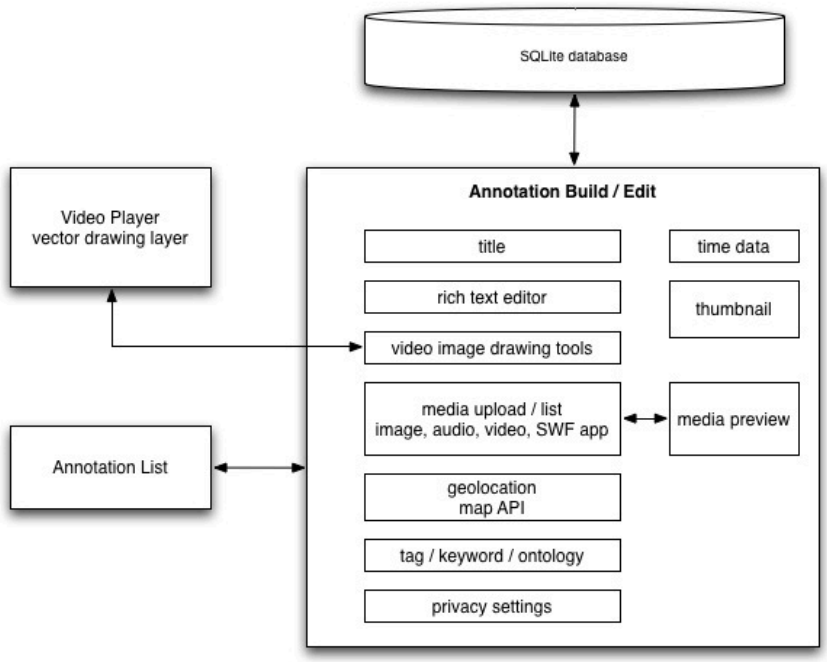

Figure 8: Core video annotation architecture

The sub-modules include:

- Baseline information about the clip fragment – time range and captured thumbnail of first frame in clip
- Title text field. The Annotation List module displays this title to identify the annotation.
- Vector drawing tools for creating basic shapes such as ovals, rectangles and lines on top of the video image, within the video player
- A rich-text editor for entering textual notes – enables rich text formatting and adding URLs
- A multi-media upload / display unit for attaching images, audio, video and compiled Flash applications (in SWF format). The unit is equipped with an option to preview the uploaded media

- A third party map interface enabling the user to add a geolocated marker on a chosen map display –zooming level, and map type
- A tag text field to categorize or subclass annotations
- A privacy selector to keep annotations private or shared with the world

When the annotation is saved, all of the module data is stored in the local database. Modules can be edited and the data updated independently. The user can also update all the the modules at once

Code Design

This section covers details about the software architecture of MVAT. The first section describes the major design pattern implemented by the application. The second section explains the object model components and their interactions. The third section describes the data model.

Programing was primarily done on Adobe's Flex Builder IDE, using the Flex 3.6 SDK. The code was written in ActionScript 3.0 (Grossman and Huang, 2006) and MXML, Adobe's XML-based, user interface markup language (Adobe, 2009).

Design Patterns

MVAT code is structured using various programing design patterns. The more prominent being the Model-View-Controller, Proxy, Data Access Object and the Singleton. Another design pattern used in sub-components was the Facade pattern. These patterns do not operate independently within the application, they are the glue that binds the data and object models by organizing the interactions between the

objects, optimizing the data exchange processes ultimately helping create a more flexible and scalable application. Figure 9 show a schematic representation of how the the model and patterns are interconnected. The following sub-sections describe how each of the patterns were implemented in MVAT.

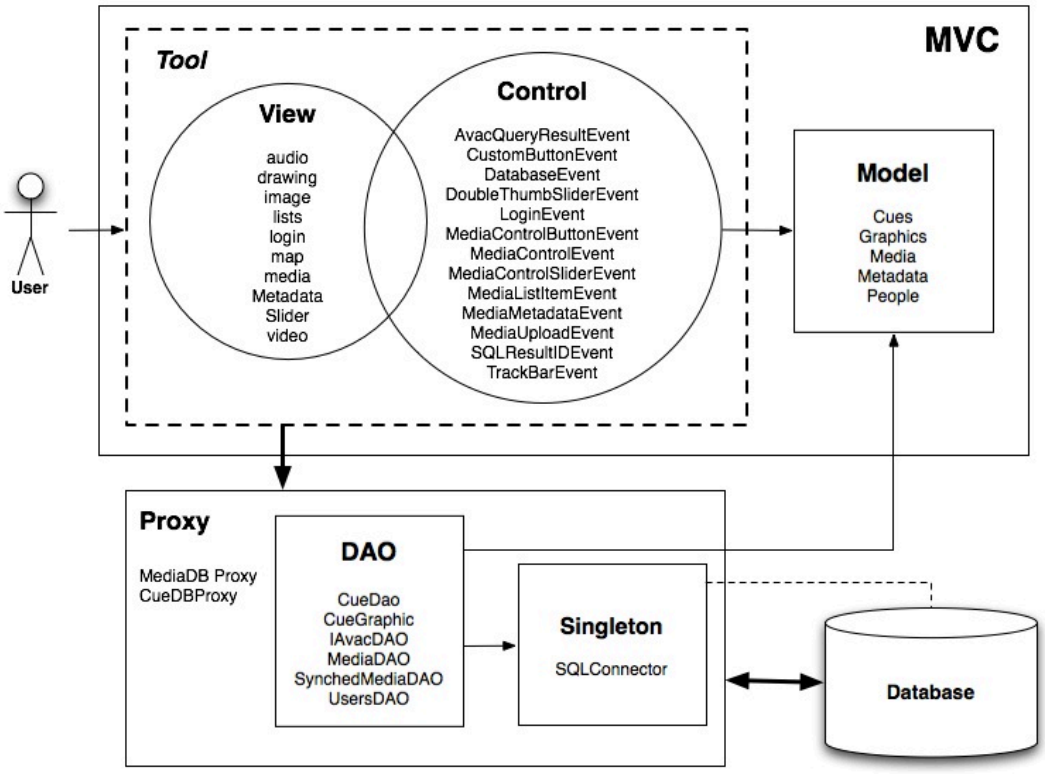

Figure 9: MVAT interconnected design patterns

Model-View-Controller

MVAT being a rich user interface it primarily implements the Model-View-Controller (MVC) design pattern (Reenskaug, 1979) at the core of its architecture. The MVC was created by Trygve Reenskaug to " bridge the gap between the human user's mental model and the digital model that exists in the computer" (Reenskaug, 2010). The MVC pattern provides a loose coupling between the input logic, the business logic

and the user interface logic, reducing the complexity of the overall architecture design and increasing the flexibility and maintainability of the code. The Model component stores the data. The View is the visual representation of the user interface. The Control is responsible for processing the user input and updating the View and the Model correspondingly (see figure 9). The model is an independent component that has no reference to the view or controller. On the other hand, the controller/view, sometimes referred to as the "tool", are tightly coupled.

MVAT comprises a collection of model/view/controller triads, each responsible for a different user interface element within the application (see details under the Object Model section). At the higher level the MVC handles all of the annotation logic, controlling the media and annotation data entered by the user, and updating the corresponding models with their respective data.

Data Access Object

MVAT relies on the Data Access Object (DAO) design pattern to handle all of the data exchange between the User Interface (UI) layer and the database layers (see the DAO under the Object Model Section). The DAO is an object that provides an abstract interface between the UI and the encapsulated SQlite database. It encapsulates the data access logic by mapping the calls from the application to the corresponding persistent data tables without exposing details of the database. The DAO improves the performance and efficiency of the data exchange and allows modifications to the implementation without altering the corresponding decoupled modules.

Singleton

MVAT implements the Singleton design pattern to ensure that only one connection is established by the application to the encapsulated SQLite database. The Singleton is used to limit the instantiation of a class to only one object and provide global access within the application to that instance. This pattern guarantees that UI objects within the application are not creating duplicate connections to the database while adding new or editing existing data information to the related data tables.

Proxy

MVAT invokes the Proxy pattern across several interfaces. The Proxy pattern generally involves a class that functions as an interface to something else. MVAT focuses on programming to interfaces. Most of the model classes rely on interfaces, especially the drawing interface (see Model.graphics package), where the iAnnotation interface stands in for any of the graphic vector object classes by exposing common property methods about each object, such as position, width, height, etc. (see figure 10).

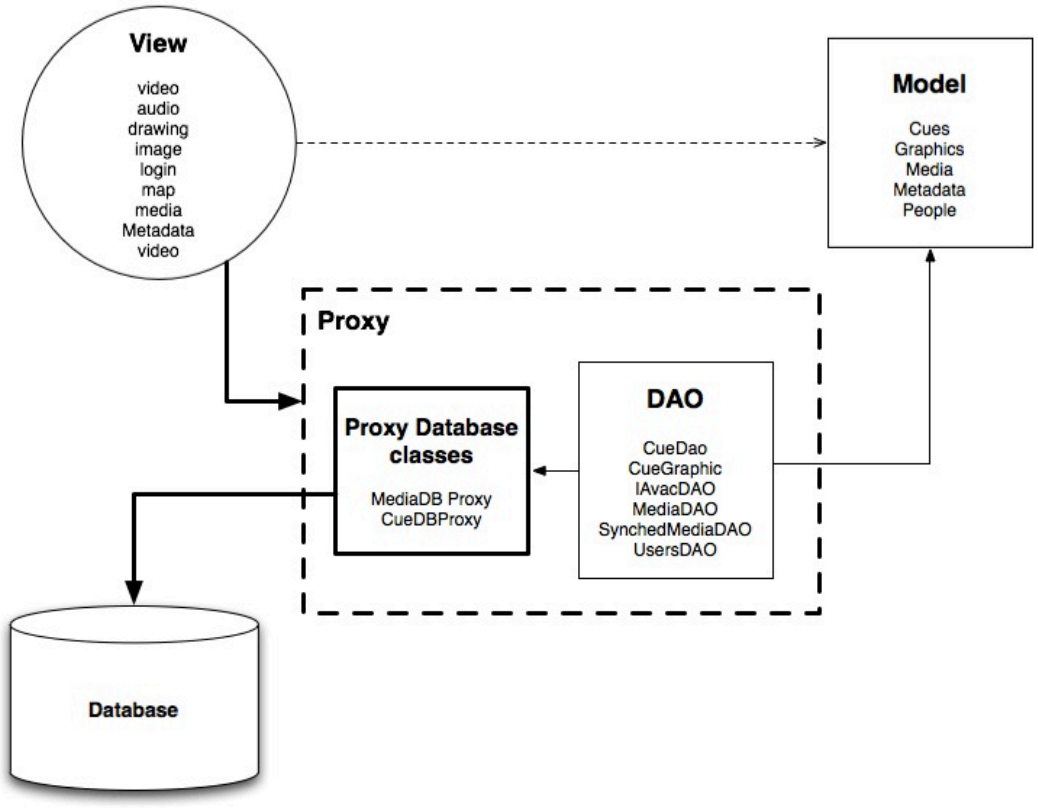

Figure 10: Database Proxy diagram

Perhaps the most notable implementations of the Proxy pattern within MVAT are the database access proxy methods interfacing between the view user interface modules and the database (see figure 10). The media database proxy (MediaDBproxy.as) accesses and exchanges metadata about the video to be annotated. The annotation database proxy (CueDBProxy.as) updates the database with all the annotation metadata associated with a given video. These database proxy classes work as complex proxy objects funneling multiple DAO classes that target specific data table structures within the database.

Facade

The Facade is a structural pattern to "provide a unified interface to a set of interfaces in a subsystem. Facade defines a higher-level interface that makes the subsystem easier to use" (Gamma et al., 1995).

The Facade pattern was instrumental for managing MVAT vector drawing tools and for drawing the annotation vector graphic objects on top of the video image. MVAT graphics model classes (see model.graphics package) make available a unified interface (IAnnotationGraphic.as) to MVAT drawing tool component (DrawingTools.mxml in view.drawing package) that simplifies the process of drawing vector graphics onto a drawing layer (see figure 11).

To illustrate the process, the View DrawingTools.mxml component relies on the single façade IAnnotationGraphic to draw each separate primitive geometric shape available to the tool, such as ellipses, rectangles or lines, onto a drawing canvas. However, when the component is "drawing" the object on the canvas it does not know what type of graphic shape it actually needs to draw: the interface takes care of that.

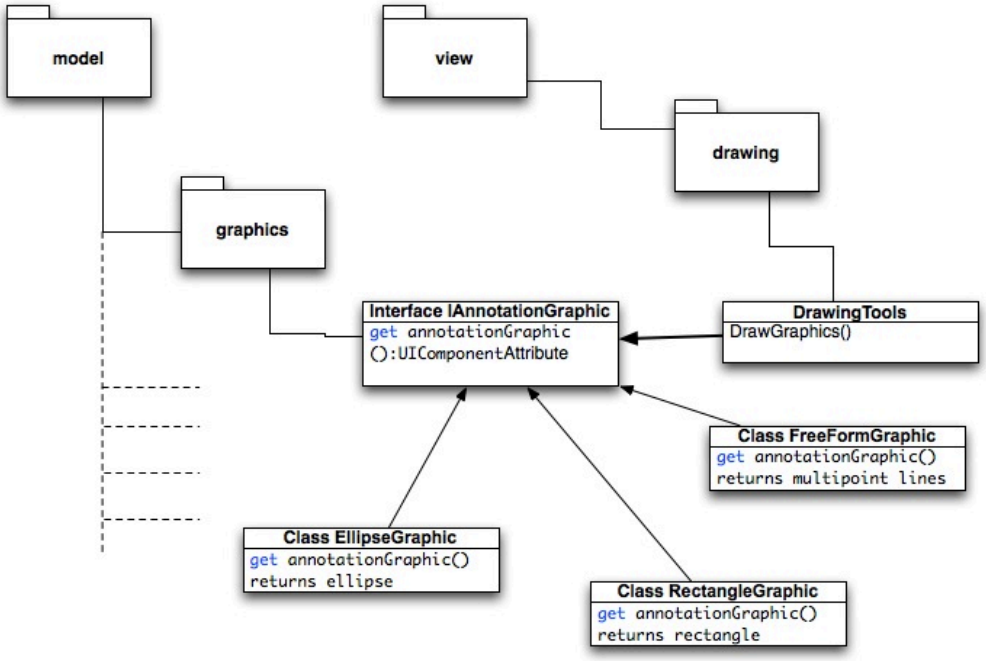

Figure 11: UML of Model-Graphic Facade

Below is a code sample detailing how the interface is implemented by the drawGraphic() method inside the Drawing Tools component.

From DrawingTools.mxml:

```
public function drawGraphics(drawlayer:UIComponent, graphics:Object):void {
  if (graphics != null)
  {
    _graphicObjects = graphics;
    for each (var cueGraphic:IAnnotationGraphic in graphics)
    {
      var newUIComponent:UIComponent = cueGraphic.annotationGraphic;
      ...
      drawlayer.addChild(newUIComponent);
     }
    }
}
```

From interface IAnnotationGraphic.as:

```
function get annotationGraphic():UIComponent;
From Class RectangleGraphic.as
public function get annotationGraphic():UIComponent{
   var newGraphic:UIComponent = new UIComponent();
   newGraphic.graphics.lineStyle(_lineThickness, _lineColor);
   _setFillColor ? newGraphic.graphics.beginFill(_fillColor): null;
  //drawRect creates a rectangle from the stored object model info
   newGraphic.graphics.drawRect(_graphicLocation.x,_graphicLocation.y,
_width, _height);
  newGraphic.alpha = _alpha;
  newGraphic.id = String(_id);
  return newGraphic; }
```

From Class EllipseGraphic.as:

```
public function get annotationGraphic():UIComponent{
   var newGraphic:UIComponent = new UIComponent();
   newGraphic.graphics.lineStyle(_lineThickness, _lineColor);
   _setFillColor ? newGraphic.graphics.beginFill(_fillColor): null;
   //drawEllipse creates an ellipse from the stored object model info
   newGraphic.graphics.drawEllipse(_graphicLocation.x,_graphicLocation.y,
_width, _height);
   newGraphic.alpha = _alpha;

   newGraphic.id = String(_id);
   return newGraphic; }
```

In the case of this graphics model example, the advantage of programming to interfaces allows the application to easily be extended and scaled with additional drawing tools for other geometric shapes, as long as the graphic classes conform to the common interface.

Object Model

MVAT code is founded on object-oriented design and interfaced-based programming. The focus is on programing against an interface rather than an

implementation; this enables polymorphism. Polymorphism is a programing approach emphasizing a flexible code structure that is modular, extensible and interchangeable. It enables a class object to stand in for any other object as long as they use the same interface (Lott and Patterson, 2007). If the interface remains intact, changes can be made to the class implementation without breaking the application. This minimizes the chances of error. This programming approach streamlines the development process and enables the architecture of the application to expand as new features need to be implemented.

The following section explains how the classes are organized, and offers a brief description of the more important ones and their function within the application. The classes and their corresponding interfaces are organized in packages according to their functionality or their role in a design pattern component. Figure 12 is a visual representation of how the class families are organized within the pattern-based workflow.

Model package

The Model package (see figure 12) is composed of sub-packages that group ActionScript classes and interfaces associated with the Model component of the MVC design pattern triad (see figure 9). Each sub package contains Model classes that are referenced by View classes mapped to a user interface component of MVAT.

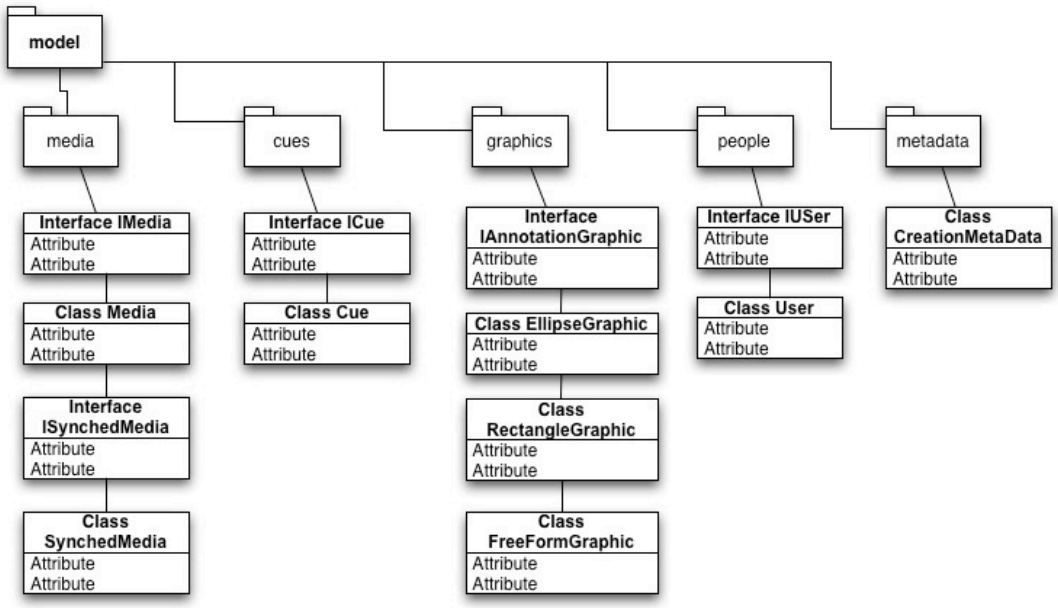

Figure 12: UML of Model sub package content

MVAT Model classes hold data about the client, video media , annotation data , including associated annotation multi-media file data , and video image annotation vector graphics (see figure 12). The classes within each sub package are related and implement a common interface. The model interface consists mainly of getter and setter methods to access or modify properties of the data model. All the model classes are independent and do not make reference to the view or controller component classes.To illustrate how the classes relate to each other between the view and the model packages consider the *graphics* sub package shown in figure 12.

View package

The View package mainly comprises Macromedia eXtensible Markup Language (MXML)(appendix 3) graphic user interface libraries (GUI), including ActionsScript code and stylesheets, responsible for the visual display components of MVAT. These

files are associated with the View component of the MVC design pattern triad. Each view MXML file relies on the corresponding Model class (see Model Package section) to draw itself. For example the Video sub-package contains the necessary components for drawing the video player, and uses the data from the media model class (model.media.Media.as) to properly, position and scale the UI components (play buttons, track-bar, video container) and scale the video image.

It should be noted that the View package is not a pure representation of the View component of the MVC triad. Some of the methods encapsulated within the View package files have Controller responsibilities, hence blurring the division between the Controller and View elements of the MVC.

Events package

The Events package consists of custom event classes extending the Flash.Events class. A rich user interface like MVAT depends on event-based programing and relies on custom events for it's operation.

Events, along with the event-handler methods, are an integral part of the controller component of the MVC. In the MVC, the controller is responsible for handling the input from a process, such as a user input event, and updating the view and model accordingly. In event-based programing, the flow of the program is determined by events, such as user actions or upon the execution of an internal code thread. The process involves a three step control operation:

- An event listener method is registered by the application, and begins "listening" for a specific event

- An event is broadcasted and is received (listened to) by the event listener method
- The event listener then calls the corresponding event handler method to determine what code thread to execute.

Every interactive UI element of MVAT is coupled to an event-based system. For instance, a user clicking on the annotation button of MVAT triggers a custom button event (see figure 13). An event listener captures the event and triggers the event handler method. The handler then launches the annotation editor window, modifies the video player track-bar control and adds new event listeners to the application stack.

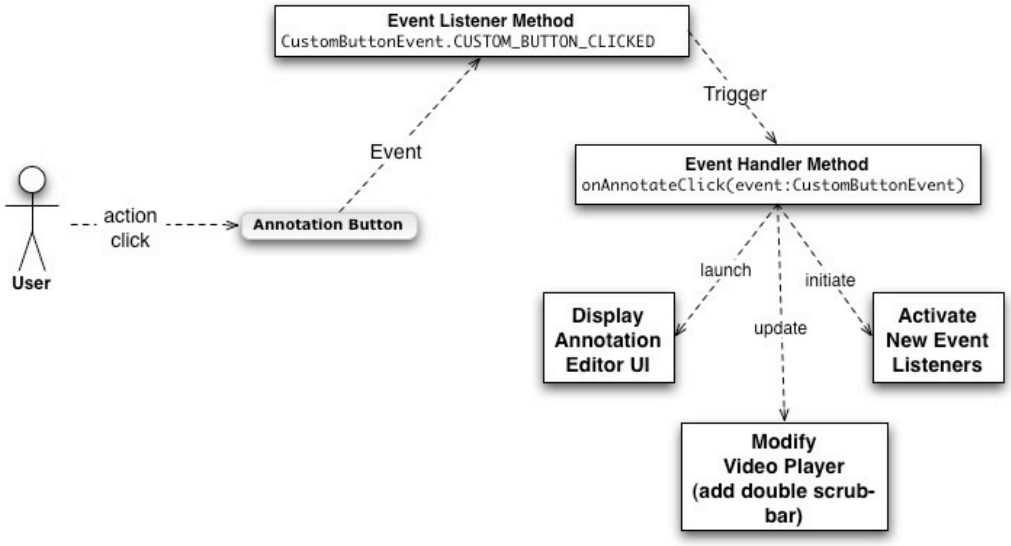

Figure 13: Event-based flow diagram

DAO (Data Access Objects) package

The DAO package, as the name indicates, groups all DAO classes designed to interface with MVAT SQLite database. Each DAO class targets a specific table within

the database. All DAO classes within the package implement a common DAO interface to access local data. (see figure 14). This interface behaves as the abstraction layer between view modules and the database.

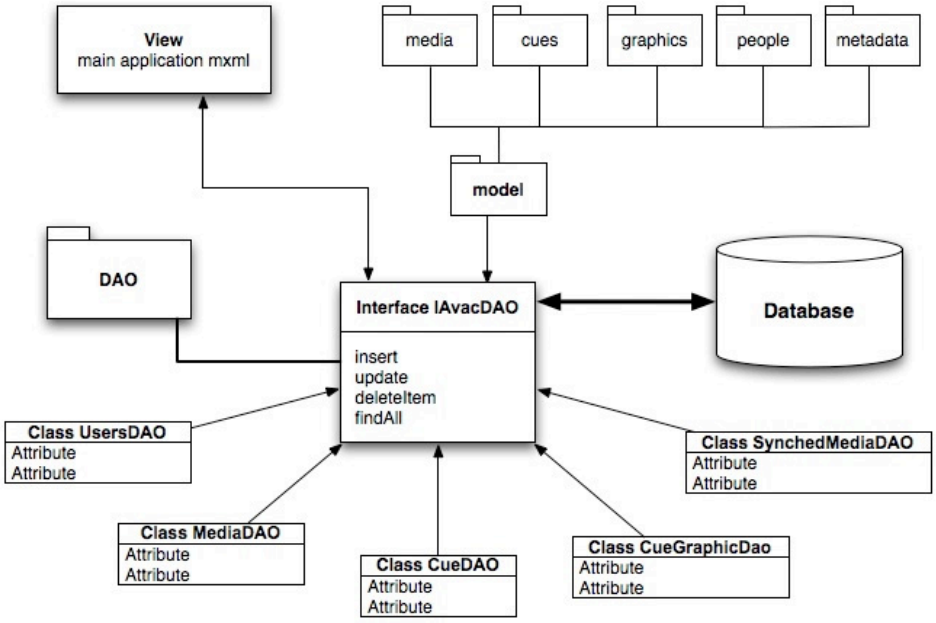

Figure 14: DAO Classes and object relations

The View uses DAO interfaces to exchange data, but it doesn't know anything about the data access logic. On the other hand, the DAO is independent from the view and knows nothing about it. The layer abstraction created by the DAO allows others views to reuse the same data access logic as long as the interface is correctly implemented

The DAO classes optimize access to the database by encapsulating all the SQL statements necessary to exchange data with the database. Upon creation of the DAO, the SQL statements responsible for querying, inserting, updating or deleting data are

instantiated and become available to public database access methods that are ready to connect to the database. These methods rely on data model object properties as attributes mapped to the database table cells.

The limitations of the DAO approach is that SQL statements are static and hard coded into the methods and don't allow easy expansion of the object to accommodate new data access options. Coenraets (2009) suggests creating "a mini DAO framework where a base DAO class would take care of all the boilerplate code to set up and execute SQL statements" or an Object Relational Mapping (ORM) framework.

Miscellaneous packages

The SQL package contains the singleton class (Singleton pattern) that ensures that only one connection is made to the database. MVAT requires only one instance of this class for all the data exchange with the database. This unique class instance is a required attribute for any DAO instantiation.

The Security package contains classes responsible for logging into the application via an encrypted local store. These take care of user authentication and user registration.

The Utils package contains utility classes commonly used by various classes. These include useful method for cloning objects, manipulating arrays, converting time data.

Data Model

The data model for MVAT is founded on a relational database structure consisting of five related tables. The tables are managed through SQLite, an embedded relational database management system. The SQLite is ideal for MVAT because the data is encapsulated within the application and therefore portable ; it implements most of the SQL standard, which makes the data interoperable with other databases such as mySQL; it is contained in a small programming library, hence it has a small footprint.

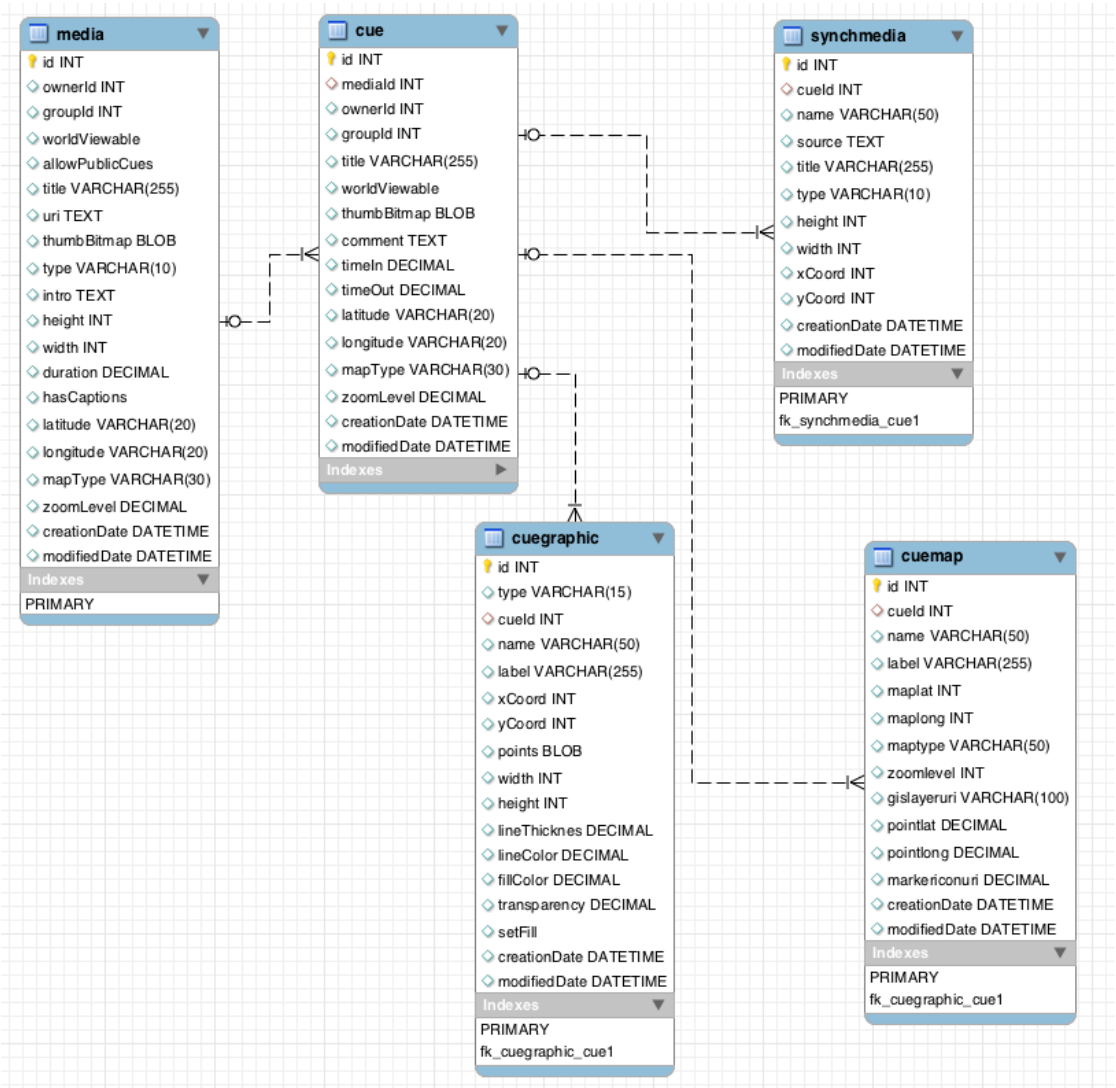

Figure 15: Data Model: Tables Relations

The data model (see figure 15) begins with the Media table as the main trunk. This is the main table that contains the reference –URI– and basic metadata about the video to be annotated. Branching from the main trunk is the the central piece of MVAT: the Core Annotation table (titled cue table). It stores kernel annotation data that drives the application. It relates to the Media table via primary key id. The three remaining tables relate directly to the Core Annotation table.

The Synchronized Media Table (titled synchmedia table) stores the references to images, audio, video files, and SWF applications that are associated with the annotation –excluding textual notes.

SVG Object Annotation table (titled cuegraphic table) stores vector graphic data for each shape–lines, spheres and rectangles–drawn on the video screen canvas layer. It also contains style info–line, fill, line thickness, color, and alpha transparency.

Finally the Geospatial Annotation table (titled cuemap table) stores maps and geolocated reference points or shapes along with associated active GIS map layer data.

The last three tables relates to the Annotation Table (titled cue table) via primary key id. The schema definition for each table is outlined in appendix 2.

MVAT Graphic User Interface

This section offers a detailed description of MVAT graphic user interface (GUI). The first sub-section gives a high level overview of MVAT GUI design, including the applications states and major components of the interface. The second

sub-section describes each UI module in detail. The third sub-section discusses the implementation challenges.

MVAT interface was developed using common UI components and native extension libraries from the Adobe Flex 3.0 framework as well as third party libraries. The Flex UI component architecture was chosen for its flexibility and modularity, permitting the rapid prototyping of complex interfaces. Flex UI components are easily interconnected through internal APIs to create reusable larger custom UI modules. MVAT is primarily driven by custom built UI components.

High Level MVAT GUI Design

MVAT is operated via four GUI states with distinct functionalities (see figure 16). The application flow begins with a gateway Authentication state, for user authentication or registration. Passed the authentication, the application opens to an initial Video List state, where videos can be added with their corresponding metadata or selected to be annotated. After selecting a video from the list, the application moves to the Annotation Edit state, where pre-existing video annotations can be selected and edited or where new ones can be created. The last sate is the Annotation Preview, where annotations can be previewed statically or dynamically displayed, synchronized with the video playhead.

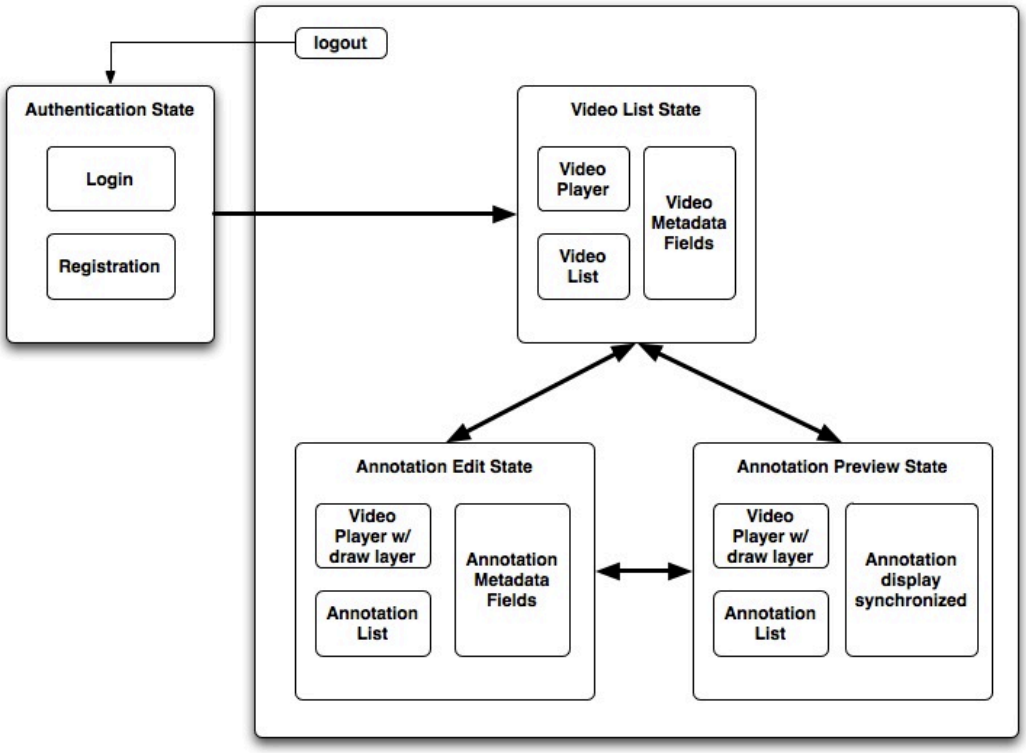

Figure 16: High-level MVAT GUI design

Switching between states is possible in all but the Authentication state. A return to the initial Authentication state is achieved by logging out of the system. Aside from the Authentication state, toggling from one state to another is a seamless process to the user, as the changes are triggered by natural user interactions with UI list and tabbed panel with minimal rearrangement of the main UI components.

MVAT Interface details and operation

Contrary to the conceptual state diagram exposed in the preceding high level GUI design section, the actual live MVAT masks the distinct annotation state views in one unified interface. Aside from the authentication module and logging in, the user

navigates between view states seamlessly by clicking video or annotation list items or tabs in tabbed window layouts.

To streamline the interface flow between the different views, MVAT shares common UI components throughout the application. For example the video list and annotation list views share the same grid display component. Likewise the video and annotation metadata share common UI common metadata components, such as the title, rich text editor and mapping interfaces.

The following sub sections describes the details of each UI component the main role they play within the application.

Authentication module

Upon launching, MVAT displays a login window prompting the client to authenticate into the system or register as a new user (see figure 17). The login interface relies on the Adobe Air Encrypted LocalStore class (Adobe, 2011) providing a persistent, encrypted data storage mechanism for storing user information within the application. When a user registers, the login data is encrypted to the local store using AES-CBC 128-bit encryption. The encrypted local store can only be accessed from the application security sandbox. Given the encrypting features, the system is ideal for deploying MVAT in a collaborative multiuser lab or museum kiosk environment.

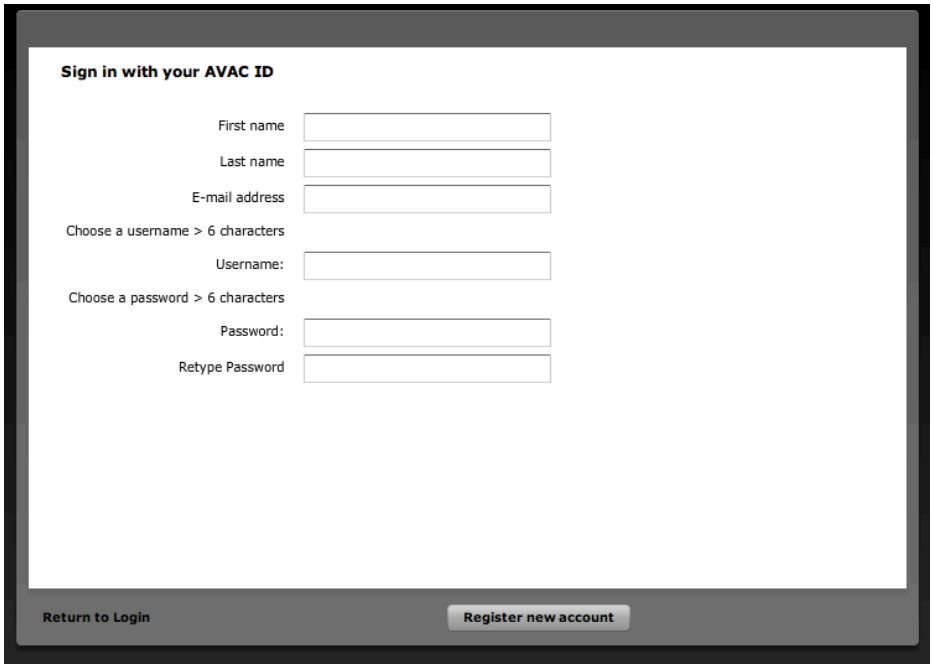

Figure 17: MVAT User registration window

Video List Management View

The Video List Management view is the first window displayed after logging into the application. It shows a split window with a grid list component on the left side and a tabbed widow panel component on the right. These components are the main containers for the tool and will remain in the same layout position throughout the application view changes.

If videos have been previously added to the tool, they will appear listed in the grid component on the left (see figure 18). In this view the user has the option to add a new video or chose to preview a preexisting one. MVAT interface simplifies the process of adding new videos by enabling a drag-drop interface that allows the user to simply drag a compatible video file (FLV or H264) from the file system on top of the video list component located on the left side of the window. As an alternative to file

drag-drop, MVAT also gives the option to enter a local or World Wide Web video file URL.

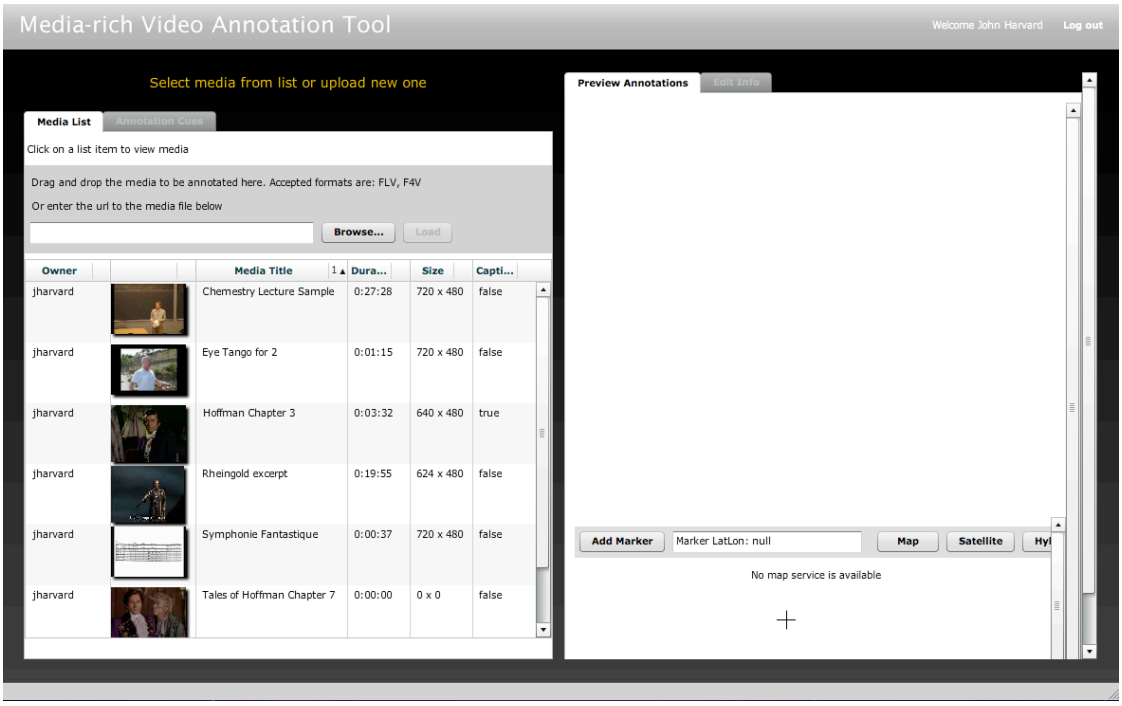

Figure 18: MVAT Video Management

When a new video file is added, the VideoPlayer component launches and loads the video ready to be played (see figure 19). To the left of the player, the tabbed information window displays basic metadata fields about the video for the user to complete. The user can provide a title for the video and a descriptive rich text annotation about the entire media piece. Moreover, the video can be geolocated in a map interface. It should be noted that some of the video metadata, such as the duration and size of the video, is automatically captured by the tool. Upon loading the video, the tool also captures a thumbnail of the first video frame image. The thumbnail is used in the video list component as a visual attribute to facilitate recognition from other videos in the list. If desired the thumbnail can be changed, by positioning the

playhead in a different location of the video playhead and pressing the “Capture Thumbnail” button.

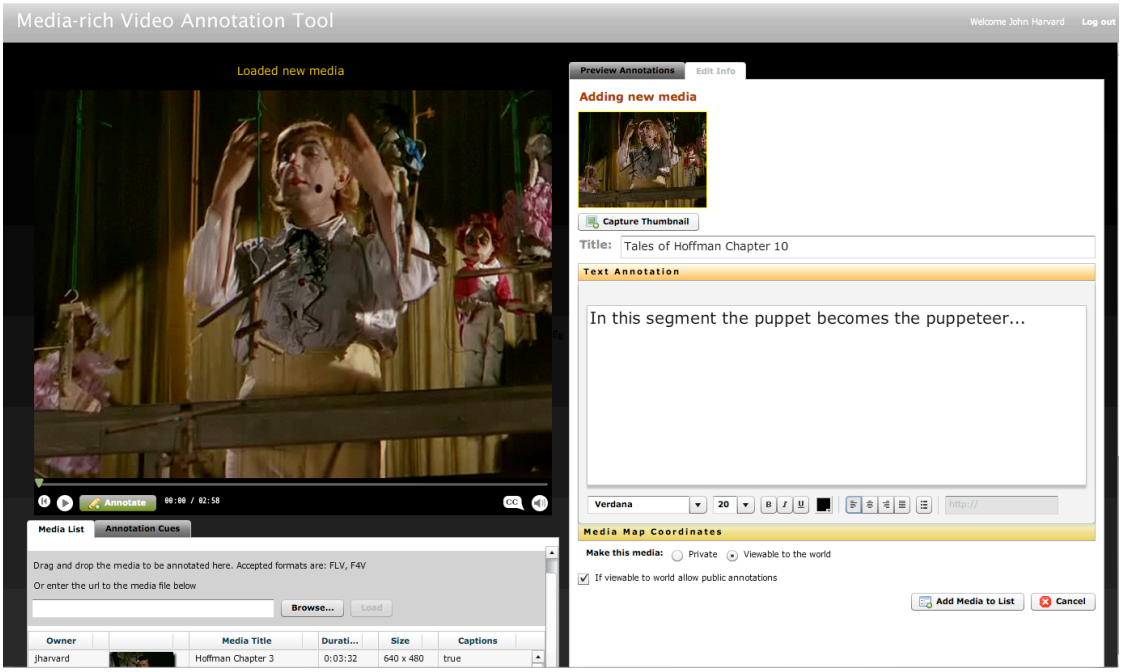

Figure 19: Loading video and adding metadata

Once the metadata has been filled the user can establish privacy settings for sharing the video, with the option to keep it private or let anyone that logs into the tool interact with it. If the video is shared with the world, the user can allow other users to annotate it or not. As a final step the user must confirm adding the video to the video list by clicking the add media button.

The Video Annotation Editor

One of the key view building blocks of MVAT is the video player (VideoPlayerCaptionsDraw.mxml), a highly customized video component with features built specifically to handle the creation and display of annotations. At the core it consists of a basic video player with standard control buttons, but the outer shell

encloses additional customized controls, such as the annotate button (an event dispatcher), an optional double scrub bar to select the range of the annotation, and a vector drawing layer that permits drawing annotation graphics on top of the video image. This component is modular and can be used repurposed for other applications or deployed as an independent video player embedded in an HTML page with its custom APIs accessible through a Javascript bridge.

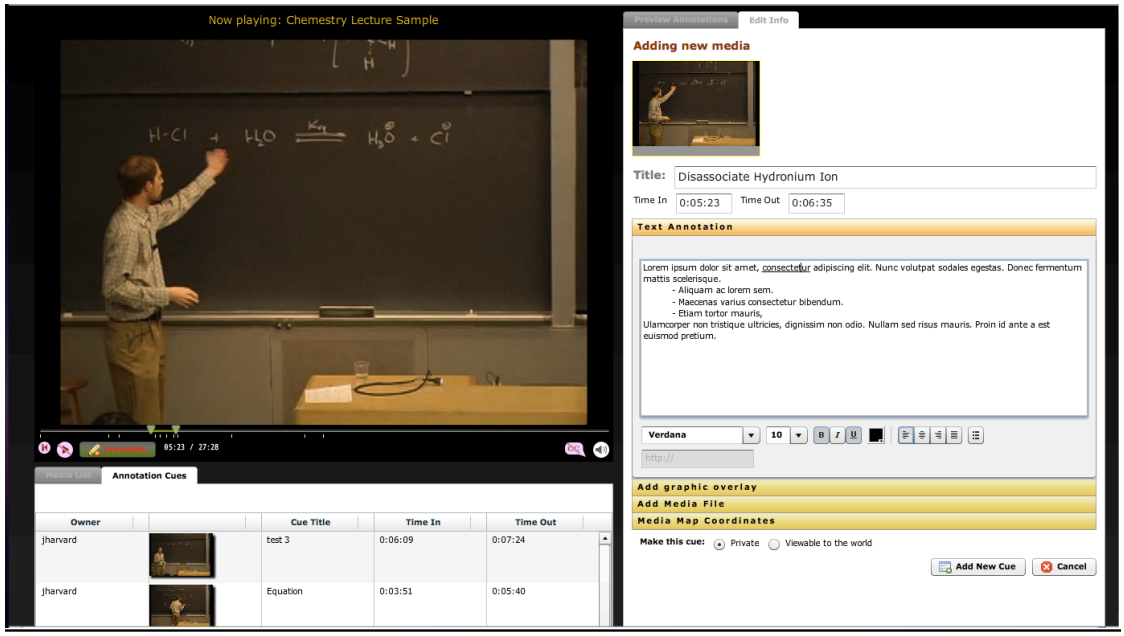

Figure 20: Annotation Edit View

As soon as a video is loaded into the player it is ready to be annotated. To create a new annotation the user simply needs to position the playhead of the player at the desired location on the video timeline and click the Annotate button on the video player. After releasing the button, MVAT seamlessly switches to the video annotation edit view, displaying the annotation input components and modifying the video player by transforming the single playhead control scrub-bar into a double double handle control time range selector (see figure 21).

Figure 21: Video player time range selector detail

The double handle time range selector UI, though small in appearance, is a key instrument of the annotation process as it enables the user to choose when the annotation should begin and when it should end. Upon releasing any of the range selector handles, the start and end timestamp data is automatically registered by the tool and the time input fields in the annotation edit panel are automatically updated. Releasing the start time handle of the range slider also updates the first frame thumbnail captured for that video segment. Like the video thumbnail in the video list component, the annotation thumbnail is used as a visual attribute to facilitate recognition among other listed annotations.

The annotation edit panel (see right side of figure 20) is where all of the annotation metadata associated with the selected video segment is entered. In addition to the previously mentioned time input fields and the thumbnail display, the edit panel contains a simple input text field to add a title; a rich text editor to enter descriptive notes; a multi-media selector to attach images, audio, video and Flash applications; vector drawing tools to draw shapes on top of the video image; and a mapping interface to add a geolocated marker; and lastly privacy controls for sharing the annotation. The panel distributes the components in a UI accordion container to maximize the space utilized by each component within the panel.

The title input text field and the rich text editor are standard Flex 3.0 UI component with no modifications. The rich text editor handles all of the text style formatting and URL linking for the textual annotation.

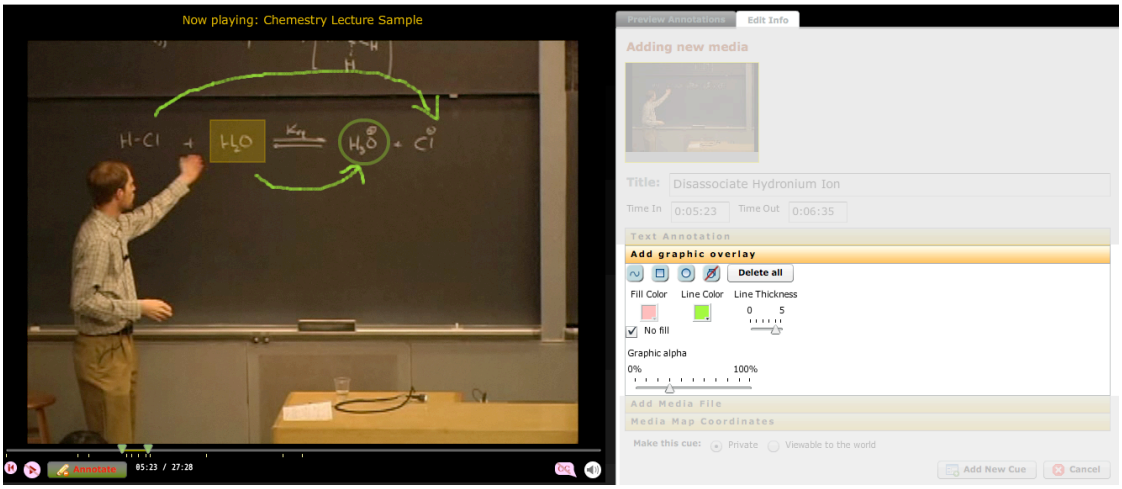

Figure 22: Video image graphic drawing tools

The vector graphic drawing tools component (see figure 22) enables the user to draw vector graphic overlays on top of the video image to highlight or point to specific regions on the image as a compliment to the annotation. The tool enable the user to draw simple ellipse and rectangle geometric shapes, and freeform lines with the mouse much like common digital vector graphic drawing tools (see video image on figure 22). Style attributes such as line and fill colors, line thickness and object transparency can be defined through the interface.The drawing tools function in conjunction with the canvas draw layer of the video player component. Specifics about the code that drives these tools is detailed in the Code Design chapter.

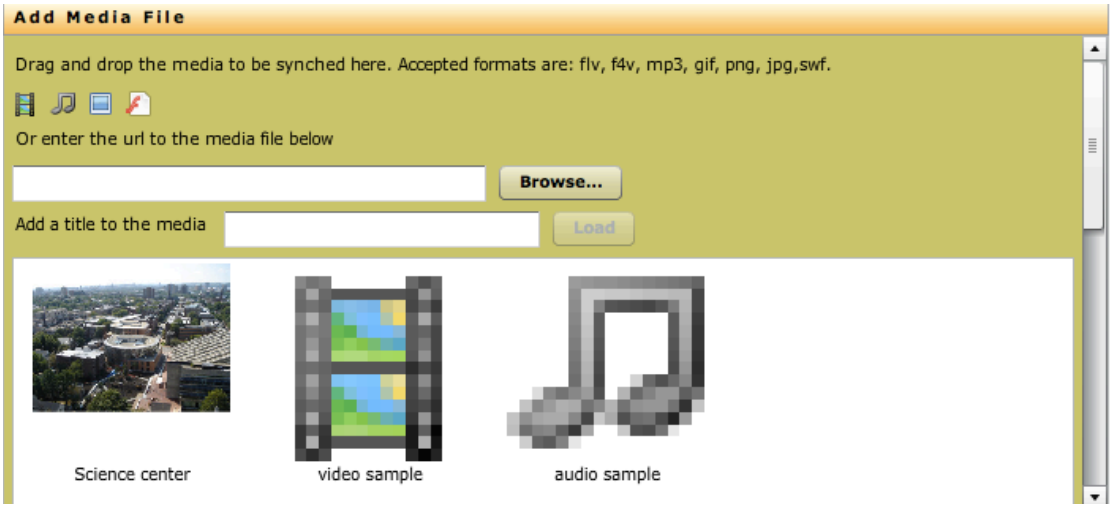

Figure 23: Media annotation manager

The annotation multi-media selector is a custom component that manages multiple media types associated with the annotation (see figure 23). The component is constructed from the same drag-drop media uploader UI component used to add videos to MVAT (MediaUploadUI.mxml) and a tile display container, the Flex 3.0 mx TileList component, to store the media files.

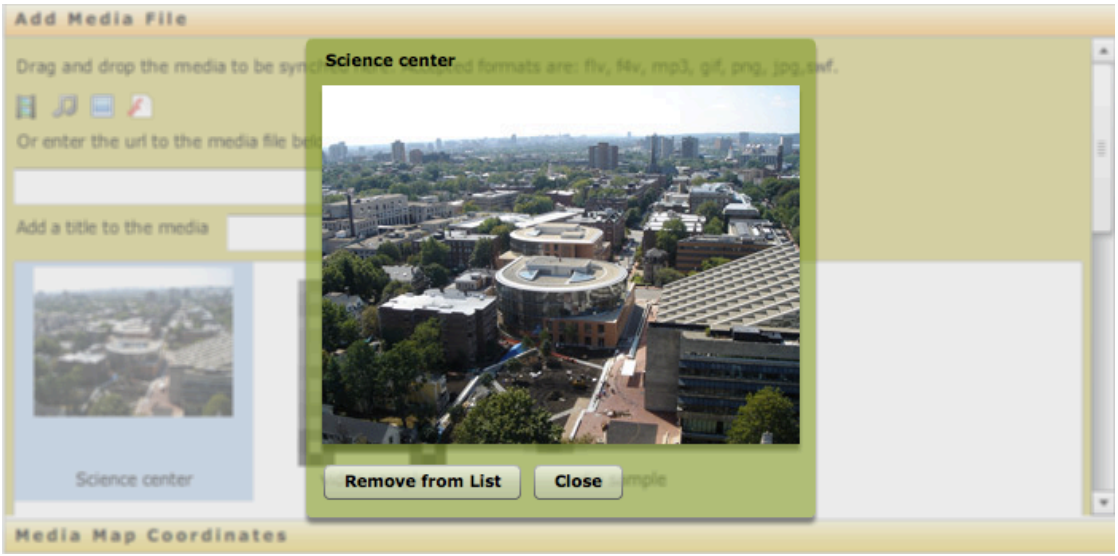

Figure 24: Media preview window

Users can attach images (JPG, PNG), audio (MP3), video (FLV, H264) and Flash applications (SWF) to an annotation by simply dragging the files from the file system, or by entering a valid URL in the input text field. Media files that are uploaded to component are listed the Tilelist. Media items can be reviewed in a preview pop-up window by double-clicking on the item (see figure 24). Items can be removed from the list from the preview window by clicking on the "Remove from list" button.

Lastly, the mapping component gives the option to georeference the annotation, by adding a marker on a map. In addition to the latitude and longitude data, the tool captures the map type – map, satellite and hybrid –and zoom viewing level. The component is built on top of the now defunct Yahoo flex mapping API (http:// developer.yahoo.com/maps/flash/).

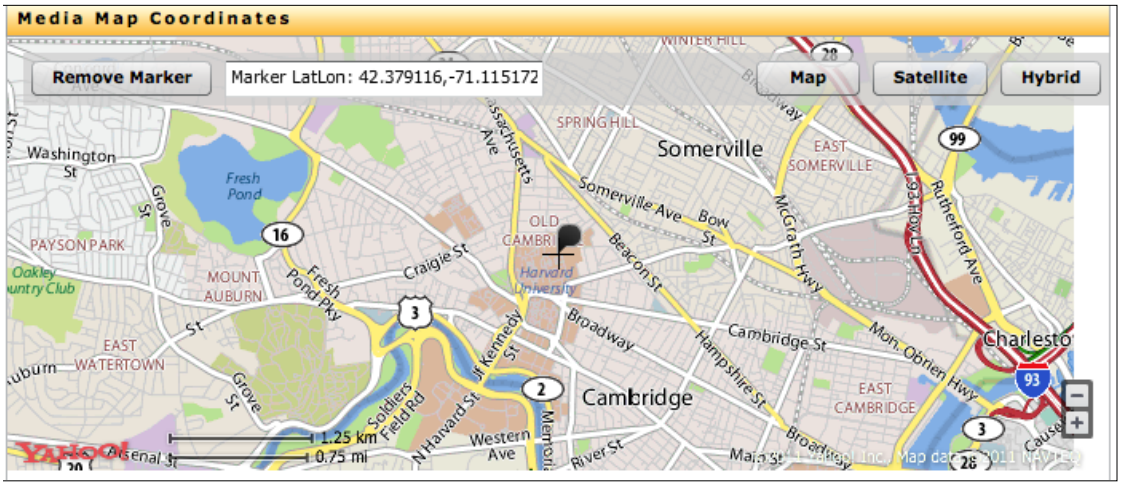

Figure 25: Annotation mapping component

Before submitting the annotation, the user has the option of preserving the content of the annotation private for personal review or sharing it with the world to foster collaboration.

As soon as the annotation is saved, the new record is added to the annotation grid list (see figure 26) and a tick mark is notched along the video player progress bar, identifying the location of the beginning of the annotation (see figure 20). One can view or edit an annotation by clicking anywhere on the row of the item in the grid list.

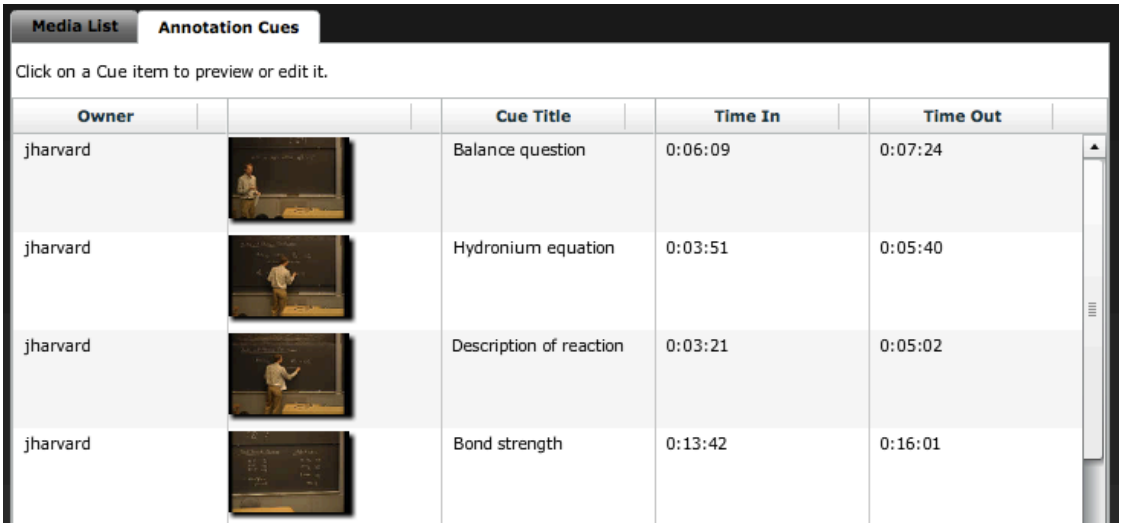

Figure 26: Annotation mapping component

Editing and Previewing annotations

Annotations can be reviewed in the editor window or alternatively in the preview window depending on which tab is selected on the tabbed annotation/preview panel on the right side of the application window. Each view option has different functionalities.

Editing Annotations

To edit or delete the content of an annotation one first needs to select the edit tab on the annotation/preview panel. Then one must click on the annotation item on the grid (see figure 26) to load the annotation data from the database into the corresponding UI modules of the annotation editor panel. The edit panel (see figure

27) is identical to the panel for creating new annotation –uses the same components– but the interface in edit mode displays database update buttons within each module. Likewise, if any vector graphic is part of the annotation, the shapes are drawn on top of the video image by the drawing component.

Data items can be modified and the changes saved to the database individually from each module by clicking on the corresponding update button. All the modules can also be updated at once by clicking the global update button at the bottom of the panel.

If needed the entire annotation can be deleted from the database by clicking on the delete button. Before permanently deleting an item a warning message offers the option to cancel or continue.

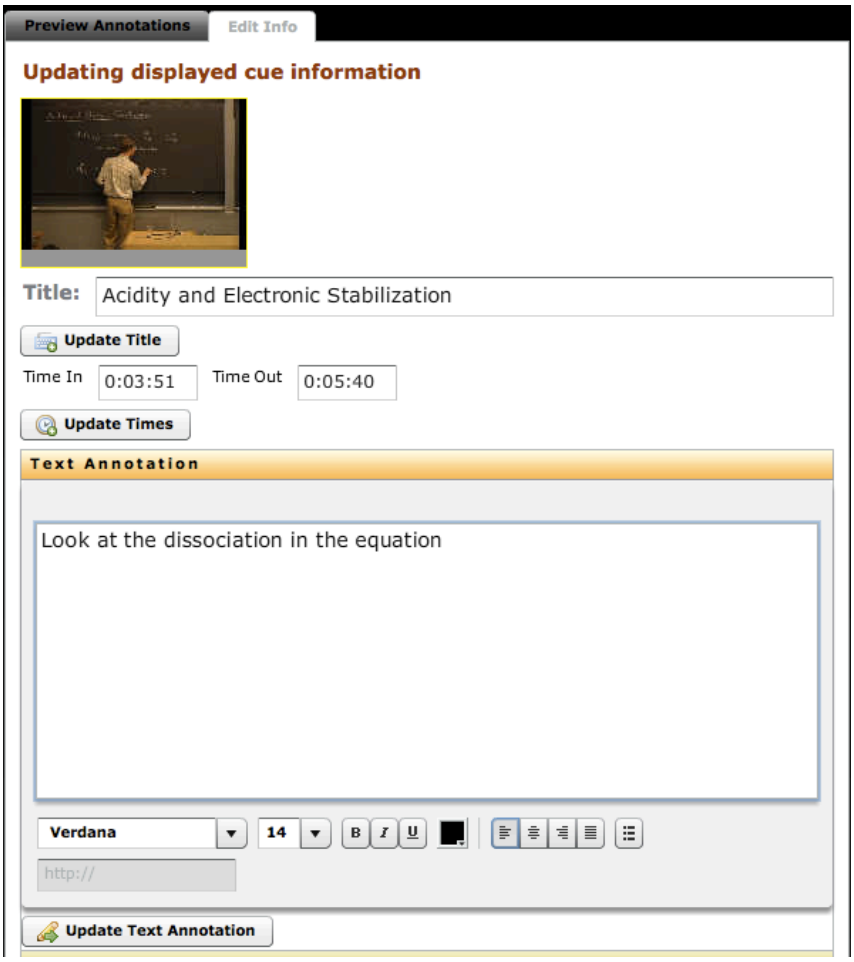

Figure 27: Example of annotation edit view

Previewing Annotations

To preview annotations, the user must select the Preview/Annotations tab on the annotation/preview panel. By clicking on this tab the application loads the annotation preview view. The preview panel appears displaying the initial metadata about the entire video. At this point the user has the option of reviewing the annotations one by one or engage the player to “play” the annotations synchronized with the video playhead.

To check individual annotations, the user can click on any of the listed annotations to loads the content into the preview panel. Once loaded, the textual notes appear first, followed by the media files and at the bottom of the panel geolocated markers in the map interface. Each media annotation file is loaded into it's corresponding container. Images are scaled to fit into the window and can be expanded to full size by clicking on them. Audio files are loaded into a basic custom AudioPlayer component, ready to be played from within the panel. Video files are loaded into a basic video player – a version of the main video player component– also ready to be played. Compiled interactive SWF application widgets that are self contained are also loaded and become fully interactive within the panel.

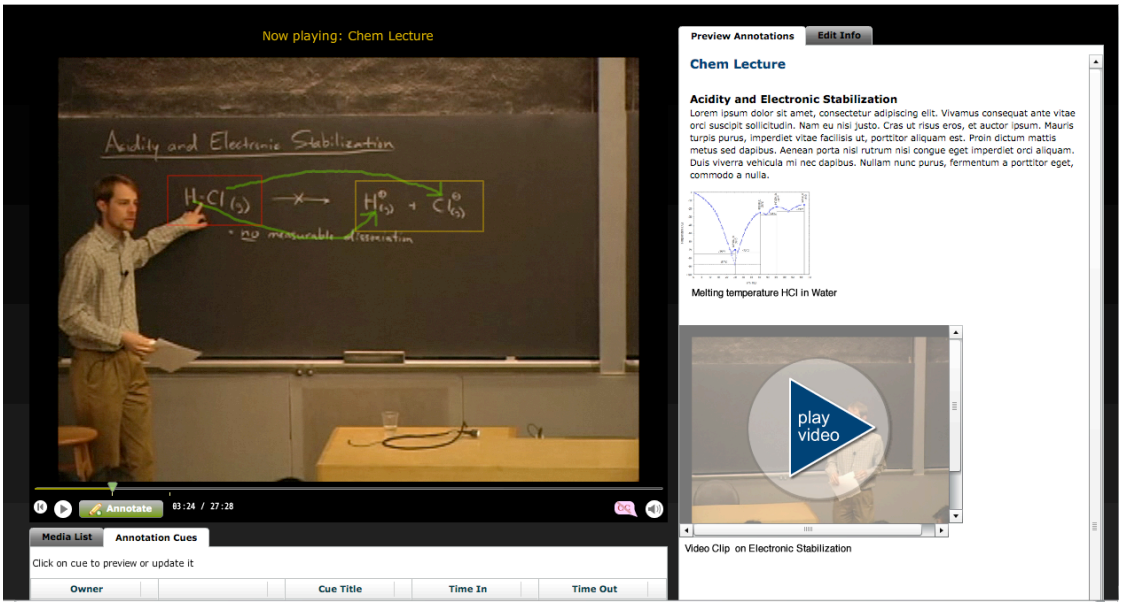

Figure 28: Previewing Synchronized Annotations

When the Preview tab is active (figure 28), the video player is automatically engaged to synchronize the annotations with the playhead. As the video clips plays, the annotations are loaded into the preview panel as soon as the time of the playhead

coincides with the the in-point timestamp of the annotation. The annotation is then removed when the playhead coincides with the out-point time stamp of the annotation. This system allows multiple annotations to be displayed at once, nesting annotations depending on how the respective time stamp ranges overlap. Stoping the video player and scrubbing the playhead or clicking the video control bar also displays all the annotations corresponding to the point on the video timeline.

Chapter 4 Summary and Conclusions

Given the ubiquity and increasing growth of video-based material used in research, teaching, and learning, a better video player with enhanced annotation capabilities is needed to fulfill scholar and pedagogical needs. MVAT is a multi-platform, open-source, stand-alone application that facilitates creating and previewing media-rich video annotations. MVAT provides a simple video player exposing annotation tools that enable text, image, audio, video, image graphics, geolocated markers and compiled applications to be attached to selected segments of the video timeline. The tool provides a framework for storing annotations for future revision or review. It also works as a presentation tool to previewed annotations by synchronizing them while playing the video. The tool satisfies requirements for teaching, research, and learning across multiple disciplines. No other video annotation tool, commercial or open source, in the market today encompass these annotation features.

The main purpose of MVAT is curatorial, not analytical. It was conceived to create, edit and review video annotations, not to analyze annotations. However MVAT provides the scaffolding and the building blocks necessary for analysis.

MVAT was founded on user-centered design principles. It is the result of various design iterations that were field-tested in in sciences and humanities undergraduate courses at Harvard University. The design aim was to develop a heuristic tool that would simplify the process of forming and managing media-rich interactive video marginalia while adhering to scholarly principles.

From the technological perspective, MVAT was devised as a multi-platform application intended for wide distribution. Flex, the development framework chosen to build MVAT, offered the most adaptable environment to produce a rich interface application, from one code base to multiple devices in the desktop, web and mobile spaces. Unfortunately, changes in the business model of the Flex product and the rapid evolution of challenging alternative Software Development Kits (SDKs), sucha as those for HTML5 and mobile, paint an uncertain future for Flex based applications like MVAT.

Regardless of development platforms and the future of MVAT within the Flex framework, the important take-away is the proof-of-concept and the design architecture of MVAT project prototype. The technology for software application development is rapidly and constantly evolving. New and better SDKs and frameworks will continue to replace old outdated ones, but the conceptual application design models for MVAT will still be valid and implementable in new development environments. Recent evidence is that MVAT served as model for the development of the video annotation component of the Collaborative Annotation Tool built for Harvard's FAS LMS –the course iSites platform.

A noteworthy aspect of MVAT application design is its modular and extendable architecture. MVAT prototype was constructed using standard and custom GUI modular components interconnected through their public API's. Components such as the video player and the drawing tools, for example, come from a collection of custom components shared by other tools. Because each annotation component is modular and

self-standing, it can be removed or turned off without affecting the operability of MVAT. Moreover, if other annotation features need to be added to MVAT, such as a live video recording annotation module or a semantic tagging module, these can easily be connected to the application. Likewise for the non GUI back end of the tool, new data models required by the new GUI components can be integrated and coupled to the database via DAO interface classes. The extendability is not limited to new annotation features: annotation data analysis and visualization modules could also be added.

The decision to release MVAT as an open source project was critical to ensure the potential growth and enhancement of the application. By opening the project the opportunities for new development become more viable. Additional discipline-specific annotation modules could be made available.

With rapid changes in technology, the original development platform may become outdated or obsolete. When the RIA industry recently began shifting from Flash to HTML5, the result was a slowdown or shutdown of third party Flash/Flex APIs (see Lessons Learned section). This of course affects the application that depend on it, such is the case for MVAT today. However with the Flex SDK moving further into the open source space and in the hands of the Apache foundation there is a good chance the framework will persevere and be a viable option for the foreseeable future thus potentially ensuring the continuity for MVAT.

But even if the Flash and Air platforms disappear into the framework junkyard, the files from the the open code repository will remain available, with algorithms, design models and database structures that could still be valid and repurposed in a new

tool. The open code repository has the potential to promote the life expectancy of MVAT.

Lessons Learned

One big lesson learned from developing MVAT is that one should be cautious about what development platform to chose. Relying on proprietary development platforms such as Adobe Flex and Air and supported third party API's though at first may seem ideal, has shortcomings.

In the beginning, the application was developed using well supported, cutting edge, cross-platform technology with robust GUI libraries that allowed rapid prototyping of rich internet applications (RIA). At the time, the decision to use Adobe Flash, along with it's Flex and Air development environments, seemed the most appropriate solution, even more so, if one considered that the majority of video delivered online was in a Flash compatible format (FLV, H264). Flash was an industry standard for creating RIA's with with proven cross-platform long term stability, arguably better than JAVA applets. However, between the time of the product's conception and the time the prototype was completed – which took over a year– Flex was unleashed from Adobe, Flash support for the mobile space was dropped and Flash is no longer in vogue and is being replaced by native HTML5 RIA alternatives.

This rapid technology shift triggered a series of events that directly affected MVAT. The announcement by Apple in 2010 that they would not support Flash on their iOS for mobile devices precipitated the decline of Flash's dominance of the RIA

market. In November of 2011, the announcement by Adobe of the open sourcing of the Flash-based Flex SDK, and turning it over to the Apache foundation, further precipitated the private software industry confidence in the future of Flash and eroded a well established support infrastructure.

The downtrend for Flash also dissolved third party industry support for new Flash-based development. Third party Flex API have since been dropped by leading software companies. Such is the case for the Flex mapping API from Yahoo used by MVAT. The mapping service is no longer available from Yahoo and hence MVAT mapping tool no longer is operational. For now, until a new mapping API is implemented, MVAT will have one less annotation parameter that can be recorded or displayed. The positive side is that given the modularity of the component, the broken mapping component can be removed and MVAT will remain fully operational while alternative mapping API can be found. On the flip side, if this tools was deployed in a live environment where research teaching or learning depended on the annotation mapping features, projects could be delayed or irreversibly affected.

Regardless of the recent bad press that Flash, and its dependent technologies such as Flex, have received, it should be noted that the much touted HTML5 alternative, with native interactive applications and media playing capabilities, is not yet mature enough to replace it – for example a common agreement on video formats is one of the main areas of contention between competing software companies. When it comes to building rich interactive application such as MVAT, Adobe Air and Flash technologies still offer the best single code base cross-platform solution.

Other lessons learned were about the limitations of the current Flash video format (FLV wrapper) and how it affects MVAT. The requirement for MVAT is that the time selection of the video segment to be annotated be accurate. Unfortunately, in a progressive downloaded Flash video–which is what occurs on the desktop–a selected video segment can only de defined between two video encoded keyframes (not timestamps). This means that if the video was encoded with a variable keyframe to optimize its compression, a selected segment to be annotated could be off by a few seconds. To overcome this issue it is possible to force the video encoder to add a keyframe every half second while encoding the video. Using this workaround it was possible to get an accuracy of half a second in video selections. However, this solution might not be adequate in research that needs more accuracy in their experiments.

Slowing down the motion is sometimes the only way to determine or detect when an event is occurring. Slow motion Flash video playback is nearly impossible to achieve. This issue poses additional limitations on MVAT and on use cases that depend on it.

Future Work

MVAT could benefit of several feature enhancements, but before future development takes place the tool should be thoroughly field tested and outstanding technical bugs and operation issues addressed. Several problems have already been identified concerning interface usability, video image graphics, and annotation modules that are no longer supported. Listed bellow are a set of features that remain to

be developed that would make the tool more useful. Because of the modular architecture of the tool the enhancements could be easily implemented.

The MVAT only allows collaboration within the same instance of the tool installed on a given computer. For MVAT to be truly collaborative it needs to connect and synchronize the annotation data with a common online MySQL database. This would allow other MVAT instances to share the annotation data and make it available to any connected MVAT user. To make this possible, an online database would have to be created coupled with an online Common Gateway Interface (CGI) script. Creating the database should be a simple linear operation since there are many similarities between the mySQL and SQLite database structures. After all, SQLite was designed as a light portable self-contained transactional SQL database engine with very similar descriptions and database creation methods as mySQL.

Another missing piece of the collaboration feature is allowing threaded discussions on annotations. In the current MVAT, a user cannot reply to a comment made a by another user. By enabling threaded discussions the collaboration and conversation around a specific commentary is completely contextualized.

MVAT would greatly benefit from a streamlined authentication and user registration system. MVAT uses a localized –encapsulated –registration and encrypted password authentication. Ideally this process should be externalized, relying on an open standard for exchanging authentication and authorization data such as the Security Assertion Markup Language (SAML). Alternatively an authentication scheme using common APIs such as Facebook user authentication API could be used in place

of this but it would not be functional unless the application is connected top the World Wide Web.

A critical missing feature in MVAT to ensure the longevity of the annotation data and foster collaboration is interoperability. For MVAT to be interoperable with other tools it needs to be able to exchange annotations –import and export– using standard annotation data models. Data exchange models are also needed for cross-referencing annotation data. One caveat is that no standard annotation data model for scholarly annotations exists at the moment. Nevertheless, a newly formed W3C Open Annotation Community Group is working on merging two partially defined annotation data models: Open Annotation (Ciccarese , 2012) and the Annotation Ontology (Sanderson and Van de Sompel 2012). The unified model will follow the Resource Definition Framework (RDF) specifications derived from W3C recommendations for data interchange on the Web. The RDF model relies on Web identifiers (URIs) and describes resources in terms of simple properties and property values (W3C RDF). Once the specifications for video annotation data models are defined, MVAT and other annotation tools that adhere to the model should be fully interoperable.

While the annotation data models are being defined, MVAT could implement other data export options. In many cases a simple text file with time stamps, textual commentaries and associated urls to media would suffice. Other options to consider would be exporting sets of annotations in formats using XML standards such as the Text Encoding Initiative (TEI).

MVAT could be improved by allowing annotations to be tagged with keywords or definitions. The option to categorize or tag an annotation is a requirement for selecting, visualizing and analyzing annotations. The tagging enables annotations to be identified as units of analysis, which are essential for ethnographical or behavioral studies. Tagging in MVAT should not be limited to simple keywords, it should include the option to add semantic tags or definitions based on predefined glossaries, taxonomies or web ontologies, even pictorial enriched ontologies as described by Bertini et. al. (2005). MVAT could include a way to point to a specific ontological definition, such as an RDF XML file, and make available the taxonomic hierarchies for labeling the annotations.

MVAT does not have a way to search and display annotations by content. A mechanism to do faceted searches of annotations based on word queries, tags, users, locations, is a necessary upgrade for MVAT. Without this feature MVAT has limited usefulness for research teaching and learning.

A simple annotation visualization module inside MVAT could plot the density of annotations –using basic histogram graphs– offering a high-level overview of the commented areas of the video that are of most interest to different commentators. The visualization would be tied to patterns matched by the search query. The visualization module could be interactive allowing the user to click on a bar of the histogram and reveal the associated annotations.

MVAT already includes media-rich annotation features, where users can upload or point a video or audio clip file as part of the annotation. This process can be

problematic if the media file is not encoded in the appropriate file format to be consumed by MVAT. An option to record a video or audio note from within MVAT would minimize these errors. The recording feature would be beneficial in foreign language courses that make extensive use of videos to learn language, enabling educators to create assignments within the video to asses spoken language skills by evaluating students replying to prompts with their own voice recordings.

MVAT implemented a mapping API from Yahoo for its annotation geolocation module that is no longer in service. The Google Maps API for Flash also has been officially deprecated. ArcGIS is one of the few currently available mapping APIs for Flex that could replace the deprecated one (ArcGIS, 2012), but it would require the Flex 4 SDK (MVAT was built using Flex 3 SDK) and updating the MVAT map API methods calls to access the API.

Finally MVAT could include new vector graphic drawing tool to enhance video image markup. A new suite of drawing tools such as a line tool, arrow tool or text label would be a useful compliment to the existing ones. With these new tools one could show delineate divisions within the video image, highlight direction of motion or label characters or object within a video scene.

Glossary of Terms

AO: Annotation Ontology.

API: Application Programing Interface.

CAT: Collaborative Annotation Tool.

DAO: Data Access Object.

FLV: FLash Video container file format.

GIS: Geographic Information Systems.

GUI: Graphic User Interface.

HTML: HyperText Markup Language.

IDE: Integrated Development Environment.

LAMP: Linux (operating system), Apache HTTP Server, MySQL (database software) and PHP (or sometimes Perl or Python).

LDAP: Lightweight Directory Access Protocol.

MAMP: Mac OS X, Apache HTTP Server, MySQL (database software) and PHP (or sometimes Perl or Python).

MVAT: Media-rich Video Annotation Tool.

MVC: Model-View-Controller programing design pattern.

MXML: Macromedia eXtensible Markup Language, XML-based user interface markup language first introduced by Macromedia in March 200.4

mySQL: is a relational database management system that runs as a server providing multi-user access to a number of databases. It is named after developer Michael Widenius' daughter, My. The SQL phrase stands for Structured Query Language.

OA: Open Annotation Collaborative.

PHP: is a general-purpose server-side scripting language originally designed for web development to produce dynamic web pages. PHP originally stood for "Personal Home Page", it is now said to stand for "PHP: Hypertext Preprocessor.

RDF: Resource Definition Framework.

RIA: Rich Internet Application

SAML: Security Assertion Markup Language (SAML) is an XML-based open standard for exchanging authentication and authorization data between security domains.

SDK: Software Development Kit is a set of software development tools that allows for the creation of applications for a certain software package.

SQL: Structured Query Language.

SQLite: is an ACID-compliant embedded relational database management system contained in a relatively small (~275 kB) C programming library.

SVG: Scalable Vector Graphics.

SWF: is an Adobe Flash file format used for multimedia, vector graphics and ActionScript. SWF was used as an abbreviation for ShockWave Flash. This usage was changed to the backronym Small Web Format to eliminate confusion with a different technology, Shockwave, from which SWF was derived.

TEI: Text Encoding Initiative.

UI: User Interface.

URI: Uniform Resource Identifier.

URL: Uniform Resource Locator.

W3C: World Wide Web Consortium.

WYSIWIG: is an acronym for what you see is what you get.

XML: Extensible Markup Language.

References

Adobe (2011). *flash.data.EncryptedLocalStore - ActionScript® 3.0 Reference for the Adobe® Flash® Platform*. Retrieved February 2012 from <http://help.adobe.com/en_US/FlashPlatform/reference/actionscript/3/flash/data/EncryptedLocalStore.html>.

Adobe (2009). *About MXML*. Retrieved February 2012 from <http://livedocs.adobe.com/flex/3/html/help.html?content=mxml_2.html>

Annotations at Harvard (2012). *Testimonials § Annotations at Harvard*. Retrieved February 2012 from <http://www.annotations.harvard.edu/icb/icb.do?keyword=k80243&tabgroupid=icb.tabgroup139304>.

ArcGIS (2011). *ArcGIS API for Flex*, Retrieved February 2012 from <http://help.arcgis.com/en/webapi/flex/index.html>.

Bargeron D., Gupta A., Grudin J., & Sanocki E. (1999, May). Annotations for streaming video on the Web: system design and usage studies. *Computer Networks* 31 (11–16) 1139-1153.

Bartlett R. (2007). *Introduction to Sports Biomechanics: Analyzing Human Movement Patterns*. New York , N.Y.: Routledge

Bertini, M., Cucchiara, R., Del Bimbo, A. & Torniai, C. (2005). Video Annotation with Pictorially Enriched Ontologies. *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*. 1428-1431.

Biella, P., Chagnon N. & Seaman, G. (1997). *Yanomami Interactive. The Ax Fight*. Forth Worth TX: Harcourt Brace College Publishers.

Biella, P. (2004). The Ax Fight on CD ROM. In E.D. Lewis (Ed.), *Timothy Asch and ethnographic film (*pp. 239-262). London: Routledge

Bossewitch J. & Preston M. (2011). *Learning Through Digital Media; Teaching and Learning with Video Annotations*. Retrieved October 2011 from <http://learningthroughdigitalmedia.net/teaching-and-learning-with-video-annotations>.

Brown B. & Cox A. (2009). Innovative Uses of Video Analysis. *Physics Teacher*, 47 (3) 145-150.

Brown, P., Ceplecha, Z., Hawkes, R. L., Wetherill, G., Beech, M. & Mossman, K. (1994). The orbit and atmospheric trajectory of the Peekskil meteorite from video records, *Nature*, 367.

Buchannan, S. (2009) *Gymnastics Video Analysis: A Tool for Women's Gymnastics Judges (GVA)*. Master Thesis in Information Technology, Degree of Master of Liberal Arts in Extension Studies, Harvard University. Boston.

Ciccarese, P. (2012). Annotation-Ontology - AO - Annotation Ontology - Google Project Hosting. Retrieved on February 2012 from <http://code.google.com/p/annotation-ontology/>.

Clegg, B. (2007). *The man who stopped time: the illuminating story of Eadweard Muybridge : pioneer photographer, father of the motion picture, murderer*. Washington D.C.: Joseph Henry Press.

Cisco (2011). *Cisco Visual Networking Index: Forecast and Methodology, 2010-2015 [Visual Networking Index] - Cisco Systems*. Retrieved March 2012 from <http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360_ns827_Networking_Solutions_White_Paper.html >.

Chaudhary, A (2008). *Video annotation tools*. Master's thesis, Master of Science. Texas: Texas A&M University.

Cherry G., Fournier J., & Stevens R. (2011). Using a Digital Video Annotation Tool to Teach Dance Composition. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning*. Retrieved March 30, 2012, from <http://imej.wfu.edu/articles/2003/1/01/index.asp..

Coenraets C, (2009). *Using the SQLite database access API in Adobe AIR | Adobe Developer Connection*. Retrieved March 30, 2012, from <http://www.pdfstandard.org/devnet/air/flex/articles/sqlite_db_api_in_air.html>.

Colasante M., (2011). Using video annotation to reflect on and evaluate physical education pre-service teaching practice. *Australasian Journal of Educational Technology*, 27(1), 66-88.

Digital History (2012). *WWII Guide: Wartime Hollywood*. Retrieved February 2012 from <http://www.digitalhistory.uh.edu/modules/ww2/wartimehollywood.html>.

Gamma, E., Helm, H., Johnson, R. & Vlissides, J. (1995). *Design patterns: elements of reusable object-oriented software*. Boston: Addison-Wesley.

Geerts, M. De Brabander, R Nuydens & Nuyens R. (1991). The Dynamic Study of Cell Surface Organization by Nanoparticle Video Microscopy. In R. J. Cherry (Ed.) *New Techniques of Optical Microscopy and Microspectroscopy (*pp. 119-136). Boca Raton: CRC Press.

Grossman G. & Huang E. (2006). *ActionScript 3.0 overview | Adobe Developer Connection*. Retrieved February 2012 from <http://www.adobe.com/devnet/actionscript/articles/actionscript3_overview.html>.

Hagedorn J., J. Hailpern, and K. & Karahalios G. (2008). VCode and VData: illustrating a new framework for supporting the video annotation workflow. In *AVI '08 Proceedings of the working conference on Advanced visual interfaces (*pp. 317-321). New York: ACM

Hardesty L. (2011). *Trillion-frame-per-second Video - MIT News Office*. Retrieved February 2012 from <http://web.mit.edu/newsoffice/2011/trillion-fps-camera-1213.html>.

Kinovea. (2012). *Kinovea - Video Analysis Software for Sports.* Retrieved February 2012 from <http://www.kinovea.org/en/>.

Kounoudes, A, Tsapatsoulis, N. Theodosiou, Z., and Milis, M. (2008). A multi-level video annotation tool based on XML-dictionaries. In Nikos E. Mastorakis, Marios Poulos, Valeri Mladenov, Zoran Bojkovic, Dana Simian, Stamatios Kartalopoulos, Argyrios Varonides, and Constantin Udriste (Eds.) *Proceedings of the 10th WSEAS international conference on Mathematical methods, computational techniques and intelligent systems (MAMECTIS'08) World Scientific and Engineering Academy and Society (WSEAS) (*pp. 455-460). Stevens Point.

Ješková Z. & Kireš N. (2007). Video measurements as a means of physical phenomena visualization. In *Proceedings 12th International Conference on Multimedia in Physics Teaching and Learning Wroclaw, Poland*. Retrieved March 2012 from <http://mptl12.ifd.uni.wroc.pl/papers/37.pdf>

Laws. P, & H. Pfister. (1998). Using Digital Video Analysis in Introductory Mechanics Projects. in *The Physics Teacher*. Vol. 36 (5) pp. 282-287.

Lorimer, R., & Scannell, P. (1994). *Mass communications: A comparative introduction*. Manchester, England: Manchester University Press

Massey S. and Quirk S. (2009) *Deep-Sky Video Astronomy*. New York: Springer Science +Business Media.

Molau, S. (1993). MOVIE: Meteor Observation with Video Equipment. In Roggemans P. (Ed.) *Proceedings of the International Meteor Conference, Puimichel, France, 23-26 September 1993 (*pp. 71-75).

Muybridge, E. (1973). *Animal Locomotion; the Muybridge Work at the University of Pennsylvania*. New York: Arno Press.

Muybridge. E. (1979). *Muybridge's Complete Human and Animal Locomotion*. New York: Dover Publications.

Pfeiffer, S. (2009) 'Patents and their effect on Standards: Open video codecs for HTML5'. In *International Free and Open Source Software Law Review*. Vol. 1 (2), pp. 131 – 138

Reenskaug T. (1979). *Trygve/MVC*. Retrieved February 2012 from <http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html.

Reenskaug T. (2010). Trygve Reenskaug personal website. Retrieved February 2012 from <http://heim.ifi.uio.no/~trygver/>.

Richard D. (2000). *Film Studies: Critical Approaches*. Oxford: Oxford UP.

Ruby J. (1996). Visual Anthropology. In D. Levinson and M. Ember (Ed.) *Encyclopedia of Cultural Anthropology*. New York: Henry Holt and Company. Vol. 4:1345-1351.

Sanderson R. & Van de Sompel, H. (2012). Open Annotation: Beta Data Model Guide. Retrieved February 2012 from <http://www.openannotation.org/spec/beta/>

Smith F. J (1913, July 9). The Evolution of the Motion Picture: VI – Looking into the Future with Thomas A. Edison. *The New York Dramatic Mirror*, p. 24, col 3.

The Economist (2011). *Electronic Education: Flipping the Classroom | The Economist*. Retrieved March 2012 from <http://www.economist.com/node/21529062>.

Valasek, J. (2012). Morphing Aerospace vehicles and structures. In Valasek, J (Ed.) *Aerospace Series.* Vol. 57. Chichester, UK: John Wiley and Sons, Ltd.

Vásquez N., Jacob K., Cocco R., Dhodapkar S., & Klinzing G. (2008) Visual analysis of particle bouncing and its effect on pressure drop in dilute phase pneumatic conveying. In *Powder Technology*. Vol.179 (3) pp. 170-175.

Wikipedia (2012). *Activity Theory - Wikipedia, the Free Encyclopedia*. Retrieved February 2012from <http://en.wikipedia.org/wiki/Activity_theory>

Yamamoto, D., Masuda, T., Ohira, S., & Nagao, K., (2008). Video Scene Annotation Based on Web Social Activities. In *Multimedia, IEEE* . Vol.15 (3), pp. 22-32.

Zhai G., Fox G., Pierce M., Wu W. & Bulut H., (2005). eSports: collaborative and synchronous video annotation system in grid computing environment. In *Multimedia, Seventh IEEE International Symposium on*. Vol 1 pp. 9 pp., 12-14

Appendix 1: Video Clients with Annotations

Advene project

http://liris.cnrs.fr/advene/index.html.

It aims at providing a model and a format to share annotations about digital video documents (movies, courses, conferences...), as well as tools to edit and visualize the hypervideos generated from both the annotations and the audiovisual documents.

Anvil, the video annotation research tool

http://www.anvil-software.de/.

ANVIL is a free video annotation tool. It offers multi-layered annotation based on a user-defined coding scheme. During coding the user can see color-coded elements on multiple tracks in time-alignment. Some special features are cross-level links, non-temporal objects, timepoint tracks, coding agreement analysis and a project tool for managing whole corpora of annotation files. Originally developed for gesture research in 2000, ANVIL is now being used in many research areas including human-computer interaction, linguistics, ethology, anthropology, psychotherapy, embodied agents, computer animation and oceanography.

Collaborative Annotation Tool

http://atgportfolio.fas.harvard.edu/node/4

The Collaborative Annotation Tool (CAT) is an online pedagogical tool for annotating text passages, large images, audio and video files using rich text commentaries and images. The tool is currently available to faculty, instructors and students at Harvard University, where it can be installed inside course websites, through Harvard's iSite web publishing platform (LMS). It was created by the Academic Technology Group (Harvard University Information Technology) for the Harvard community. It is scheduled to be released to the world as an open-source application sometime in 2012.

dotSUB - Online Crowdsourced Video Translation

http://dotsub.com/.

dotSub is an application that allows watch videos with subtitles in any language, upload your videos, create your own subtitles

Kaltura Video Platform

http://www.kaltura.org/

Full featured open source video platform running on your own servers or cloud.

Kinovea

http://www.kinovea.org/en/

Kinovea is a free and open source solution for video analysis.

It is mostly used by sports coaches and athletes to explore, study or comment a performance. In addition to this primary focus, Kinovea is also used by animation artists, podiatrists, and ergonomics engineers.

MotionClip Game Film Analysis Software

http://www.allsportsystems.com/

Video Tagging Software for Team Sports, Science, and Research. Game film breakdown and video analysis software for sports. Provides video analysis software for team sports and research requiring association of data with specific moments in video. Used by football coaches, soccer coaches, researchers and sports professionals requiring video analysis, data tagging, filtering, and statistical analysis of video recordings.

Motionpro

http://www.motionprosoftware.com/

MotionPro provides swing analysis and motion analysis software for golf, tennis, bowling, baseball, and all other sports.

Multimedia Annotator

http://languageinstitute.wisc.edu/content/projects/authoring_tools.htm.

The University of Wisconsin-Madison has created an instructional application called the Multimedia Annotator (MmA). The Multimedia Annotator is an easy-to-use tool designed by and for language teachers to provide students with a variety of kinds of help as they view a video clip.

Pad.ma

http://pad.ma/

Short for Public Access Digital Media Archive - is an online archive of densely text-annotated video material, primarily footage and not finished films. The entire collection is searchable and viewable online, and is free to download for non-commercial use.

Project Pad

http://projectpad.northwestern.edu/ppad2/.

The Video and Audio Tools lets you attach comments to time segments of Flash FLV video and MP3 audio streams. The tools can be used by instructors and / or student

teams to critique student-produced video and audio or to provide a way for students to analyze scientific, historic, or artistic recordings.

Quintic Software

http://www.quintic.com/software/

Software provides Video Capture and Biomechanical analysis of an individual's technique for athletes, coaches, physiotherapists, podiatrists and sports scientists.

Simple Video Annotation tool

http://videoannotation.codeplex.com/

This simple tool allows you to add tags and annotations to video, similar to YouTube video annotation. This is a working application, but there are plenty of extra features that could be added. Its written in c#

Siliconcoach Pro

http://www.siliconcoach.com/products/siliconcoach_pro#introduction

Siliconcoach Pro is a video analysis software designed analyze movement and providing detailed feedback to enhance performance. It can measure and highlight key points or step through the video frame by frame to see precise details. Provides dual screen, quad screen or overlay movies to compare videos. Allows to record verbal and visual feedback. Rather than describing movement, you can visually show it.

Sportstec

http://www.sportstec.com/

Sportstec provides a suite of sophisticated video analysis software for sports coaching. The company labels their product as "Performance Analysis" software that covers the demands of coaching staff.

Studiocode

http://www.studiocodegroup.com/

Studiocode is a video analysis tool used within the healthcare, education, research and corporate sectors to improve, measure and analyze performance.

Vannotea

http://itee.uq.edu.au/~eresearch/projects/vannotea/

Application developed by the School of information Technology and Electrical Engineering e-Rsearch Group for collaborative indexing, annotation and discussion of audiovisual content over high bandwidth networks

Video Annotation and Reference System (VARS)

http://vars.sourceforge.net/.

The Video Annotation and Reference System (VARS) is a software interface and database system that provides tools for describing, cataloging, retrieving, and viewing the visual, descriptive, and quantitative data associated with video. Developed by the Monterey Bay Aquarium Research Institute (MBARI) for annotating deep-sea video data, VARS is currently being used to describe over 3000 dives by our remotely operated vehicles (ROV). VARS has allowed MBARI scientists to produce numerous quantitative and qualitative scientific publications based on video data.

VideoAnnEx annotation tool

http://www.research.ibm.com/VideoAnnEx/

The *VideoAnnEx* annotation tool assists authors in the task of annotating video sequences with MPEG-7 metadata. Each shot in the video sequence can be annotated with static scene descriptions, key object descriptions, event descriptions, and other lexicon sets. The annotated descriptions are associated with each video shot and are stored as MPEG-7 descriptions in an output XML file. *VideoAnnEx* can also open MPEG-7 files in order to display the annotations for the corresponding video sequence. The annotation tool also allows customized lexicons to be created, saved, downloaded, and updated.

Video Capture of Practice

http://video.ceit.uq.edu.au/
A video sharing application built by Center for Educational Innovation and Technology, University of Queensland, intended to support the use of Video Capture of Practice as a learning activity. The application allows upload of video capturing practice and collaborative assessment.

VCode & VData: Video Annotation Tools

http://social.cs.uiuc.edu/projects/vcode.html.

VCode and VData are a suite of "open source" applications which create a set of effective interfaces supporting the video annotation workflow . The system has three main components: VCode (annotation), VCode Admin Window (configuration) and VData (examination of data, coder agreement and training). The Design of VCode and VData was grounded in existing literature, interviews with experienced coders, and ongoing discussions with researchers in multiple disciplines.

Viddler

http://www.viddler.com/.

- Allows adding Tags, text, links, and video comments on the video timeline
- Allow others to add timed comments

- Threaded conversations

Video Interactions for Teaching and Learning (VITAL)

http://vital.ccnmtl.columbia.edu/ccnmtl/vital3/login.smvc

Video Interactions for Teaching and Learning (VITAL) is a Web-based learning environment that enables students to view, analyze, and communicate ideas with video. VITAL was originally created to help students practice their observation and interpretation skills in developmental psychology courses at Teachers College, Columbia University. Currently, VITAL is deployed in a wide range of courses and disciplines across Columbia University, from the School of Social Work to the School of the Arts.

VideoPaper

http://vpb.concord.org/

Developed in 2000, VideoPaper was funded by National Science Foundation as part of the Bridging Research & Practice project at Technical Education Research Centers. VideoPaper has been used in a number of national and international settings, with focus ranging from mathematics to teacher education. VideoPaper allows users to associate comments as well as images to a specific portion of video. Recent versions support captioning providing a synchronized written transcript or other written elaboration while video is displayed. The user can toggle between video and text comments in real time by selecting the corresponding links. VideoPaper also allows the linking of images to specific locations in a video segment, such as a video depicting a preservice teacher helping a student with classwork accompanied by images of the student working on the problem. (description from Rich et al. 2009)

Video Traces

http://depts.washington.edu/pettt/projects/videotraces.html

VideoTraces was conceived and designed by Dr. Reed Stevens of the College of Education and has been developed and studied in collaboration with PETTT. Video Traces is a system that makes it easy to capture a piece of rich digital media, such as video or a digital image, and to annotate that media both visually (using a pointer to record gestures) and verbally. The resulting product is a "video trace": a piece of media plus its annotation--in essence, a recorded "show & tell". Traces can be viewed by their creator, exchanged with others, and further annotated for a variety of teaching and learning purposes.

Vertov

http://digitalhistory.concordia.ca/vertov/

Vertov is a free media annotating plugin for Zotero, an innovative, easy-to-use, and infinitely extendable research tool. Both are Firefox extensions. Vertov allows you to

cut video and audio files into clips, annotate the clips, and integrate your annotations with other research sources and notes stored in Zotero.

VideoAnt - Video Annotation Tool

http://ant.umn.edu/

Synchronizes web-based video with timeline-based text annotations.

VAT - Video annotation tool | Bohemie Research Project

http://www.boemie.org/vat.

VAT is a tool to manually annotate MPEG video files. It provides a user friendly interface for the accurate and live and "frame by frame" annotation of video. VAT imports its descriptors from predefined OWL ontology files. In addition, VAT enables the definition and employment of user defined descriptors. Three different workspaces opening as different tabs allow for the annotation of the video as a whole, the annotation of specific shots and the annotation of specific regions. As regards to regions, size adjustable region marking boxes enable the captivation of movement trajectories, simply by using mouse dragging and dropping. Originally developed within the BOEMIE research project, VAT has already been successfully used for the annotation of sports events videos. Its augmentation is considered as an ongoing research effort for MKL.

Video Image Annotation Tool | Informatics and Telematics Institute

http://mklab.iti.gr/via/.

Video Image Annotation Tool (VIA) is a windows application to manually annotate video and images. It provides a user friendly interface for the accurate and undemanding live and "frame by frame" annotation of video and still images.

The development of this tool has been supported by the Bootstrapping Ontology Evolution with Multimedia Information Extraction (BOEMIE)

XMAS

http://web.mit.edu/shakspere/xmas/index.html

Cross Media Annotation System (XMAS), developed under the MIT-Microsoft iCampus Initiative. The MIT Shakespeare Project has been developing tools to aid in the study and comparison of Shakespeare texts, images and films. By combining on-line discussion and text editing with video sequences on DVD, XMAS makes possible a range of new teaching and learning techniques in which text, video and images can be studied, excerpted, and shared remotely.

YouTube video annotation

http://www.youtube.com/t/annotations_about.

- Web application.
- Free to use.
- Only annotates videos on YouTube. You can only annotate videos you have uploaded, while others can see the annotation.
- Text annotation (“text bubbles” or notes), highlight part of the screen.
- All annotations are editable.

Appendix 2: Database Table Schema Definitions

Media table

id INTEGER PRIMARY KEY AUTOINCREMENT,
ownerId INTEGER,
groupId INTEGER,
worldViewable BOOLEAN,
allowPublicCues BOOLEAN,
title VARCHAR (255),
uri TEXT,
thumbBitmap BLOB,
type VARCHAR (10),
intro TEXT,
height INTEGER,
width INTEGER,
duration NUMERIC,
hasCaptions BOOLEAN,
latitude VARCHAR (20),
longitude VARCHAR (20),
mapType VARCHAR (30),
zoomLevel NUMERIC,
creationDate DATETIME,
modifiedDate DATETIME

Core Annotation Table

annotation id INTEGER PRIMARY KEY AUTOINCREMENT,
mediaId INTEGER,
ownerId INTEGER,
groupId INTEGER,
title VARCHAR (255),
worldViewable BOOLEAN,
thumbBitmap BLOB,
comment TEXT,
timeIn NUMERIC,
timeOut NUMERIC,
latitude VARCHAR (20),
longitude VARCHAR (20),
mapType VARCHAR (30),

zoomLevel NUMERIC,
creationDate DATETIME,
modifiedDate DATETIME),

Synchronized media table

id INTEGER PRIMARY KEY AUTOINCREMENT,
cueId INTEGER,
name VARCHAR (50),
source TEXT,
title VARCHAR (255),
type VARCHAR (10),
height INTEGER,
width INTEGER,
xCoord INTEGER,
yCoord INTEGER,
creationDate DATETIME,
modifiedDate DATETIME,

SVG Object Annotation Table

id INTEGER PRIMARY KEY AUTOINCREMENT,
type VARCHAR (15),
cueId INTEGER,
name VARCHAR (50),
label VARCHAR (255),
xCoord INTEGER,
yCoord INTEGER,
points BLOB,
width INTEGER,
height INTEGER,
lineThicknes NUMERIC,
lineColor NUMERIC,
fillColor NUMERIC,
transparency NUMERIC,
setFill BOOLEAN,
creationDate DATETIME,
modifiedDate DATETIME