# General and Specific Interfaces: Experiences with a Multimedia Platform

**David Bargeron, Jonathan Grudin and Anoop Gupta**

October 2, 2001

Technical Report
MSR-TR-2001-90

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA  98052

# General and Specific Interfaces: Experiences with a Multimedia Platform

**David Bargeron, Jonathan Grudin, Anoop Gupta**
Microsoft Research
One Microsoft Way
Redmond, WA 98052-6399 USA
+1 425 706 0784
{davemb, jgrudin, anoop}@microsoft.com

## ABSTRACT

Tradeoffs have always existed between a consistent, widely-used interface and interfaces optimized for specific purposes. However, the balance may be shifting toward a focus on specialized interfaces due to increased capabilities of software and rising expectations of users. We illustrate these effects through the description of a multimedia annotation system deployed in a series of laboratory and field studies. Laboratory studies identified more generic problems and guided general improvements. Field studies invariably identified context-specific requirements that presented challenges for a general interface. These studies have implications for the designers and users of multimedia systems and suggest a broader trend away from applications and toward platforms.

**Keywords**
Multimedia, interface optimization, annotation

## INTRODUCTION

The tradeoffs between a single, widely-applicable user interface and multiple, task-specific interfaces are well known. They occur at all levels, from enforcing “look and feel” across the features of a single application to ensuring brand uniformity across an entire corporate product line. They apply to hardware as well as software.

Traditionally, software user interfaces have been aimed at as wide an audience as possible to offset the high cost and complexity of producing more specialized interfaces. Today, however, software is more flexible, and users’ expectations are higher. This is tilting the balance away from more “generic” user interfaces and toward more task-specific interfaces. Consequently, there is more to be gained from building platforms on which specialized user interfaces can easily be fashioned.

The advantages to a single, widely-used interface include lower cost of design, development, and maintenance. The resulting uniformity is also beneficial to users, who need learn only one interface and are often better equipped to handle novel applications or parts of applications. Apple users, for example, have benefited significantly from the uniformity promulgated by the enforcement of a standard set of interface guidelines [2].

But there are limits to how generic an interface can be before it suffers from not meeting context-specific requirements. For Apple, placing pull-down menus at the upper left of the monitor worked well for the original 8” Mac and works fine on handheld devices, but it is awkward for multiple monitor configurations and will be unacceptable for wall-sized displays. A simple search capability might be perfect in a word processor, but is insufficient for a database application, and yet another interface will be needed for Web searches.

Despite the difficulties involved in designing a widely-applicable user interface, many examples nonetheless exist: The Windows and Mac operating systems, IE and Netscape browsers, Microsoft Word, AOL Instant Messaging, Real Audio, and so forth. On the other hand, two trends are working to make such successes more difficult:

1. Software is attempting to do much more for people. The closer it engages with their activities, the more it will suffer from a lack of optimization for individual work (and play) contexts.
2. The secret is out: Computer users know that software is very flexible. Therefore, expectations are increasing. People are less willing to adapt their practices to use software when they realize that it could work better.

In light of these factors, there is a growing demand for more flexible software platforms on which domain experts can easily build their own task-specific user interfaces. This trend will have significant impact on both researchers and practitioners who wish to design useful software systems with effective user interfaces. We illustrate the trend through our sometimes surprising experiences developing, deploying, and refining a multimedia annotation system.

We built the first version of the system in early 1998. It was intended to be a widely-usable, ‘generic’ platform enabling people to speak or type comments that are linked to specific points or ranges in a video or audio presentation. Annotations are stored independently and thus can comment on any multimedia file on an intranet or on the Web. Annotations can be private or shared with a specified group; and they can be responded to or emailed, with email recipients able to play the associated segment of multimedia. We envisioned it being used to support discussions around archived video, including taped classroom lectures, digitized films, videotaped usability sessions, and so forth.

We conducted a series of laboratory and field studies using the system we developed, modifying the software and redesigning its user interface each time. The lab studies identified problems and enabled us to make generally useful changes. Each field study, on the other hand, identified new design requirements that were specific to the context in which it was deployed, and without which the acceptability of the system was in question. As this research has progressed, it called into question how useful a "generic" interface will be and forced us to enhance the original annotation system with features that have transformed it into a general-purpose multimedia annotation platform.

Following a brief discussion of related work in the next section, we describe our initial multimedia system in more detail. We then describe a series of lab studies and field deployments, illustrating them with different interfaces that resulted. Finally, we discuss the implications of these results for designers of platforms and applications.

## RELATED WORK

The intent of this paper is to describe the evolution of interfaces to a multimedia annotation system, culminating in a suggestion that the relationship between widely used applications and extensible or tailorable systems is changing. There are a few literatures of some relevance.

Because a multimedia annotation system is used as an example in this paper, the fact that it builds on other systems such as the Classroom 2000 project [1] is only indirectly relevant. Those interested in a more detailed description of multimedia annotations and reviews of the relevant literature can find them in the several cited papers that provide details of studies summarized in this paper [3][4][14][15].

Discussion in the HCI literature has largely centered on enabling "end-users" to tailor their environments [8][9][10][18]. But several researchers have found that people do extremely little customization of their applications and systems [16][17]. As a result, providing the ability to tailor the interface increases the interface complexity as well as the development and maintenance cost, but does nothing to remove the necessity for providing the best possible default design.

In this paper we do *not* propose people build highly customizable interfaces aimed at the end user. Instead, we propose that designers shift their thinking toward more generic platforms on which domain experts (but not necessarily software developers) can fashion task-specific user interfaces.

Other research has addressed the evolution of human-computer interfaces [6][11][12][13]. The effects of designing for increasingly specialized activities in a context of growing expectations fits into the progressions described in these papers without being an issue upon which they focus.

And finally, extensible or tailorable systems has been a major topic in the software engineering literature [19][20]. While the goals of extensibility in software engineering are somewhat different from our goals in designing a multimedia annotation system, there is nonetheless much to be learned from software engineering in the area of designing more generally useful platforms.

## MULTIMEDIA ANNOTATION SYSTEM

In this section we examine our initial system design goals, and we describe the architecture and original user interface features of the system. As preface, we present a scenario to illustrate the use of our multimedia annotation system as we originally envisaged it.

### Scenario

A student logs in to watch a lecture in the evening from her home computer. Through her web browser she receives the audio and video of the lecturer, the associated slides that flip in synchrony with the video, and notes associated with the slides. In addition to typical VCR-like navigation features for the lecture video, there is a table of contents of slide titles, and with a click she can "seek" or jump the presentation to the appropriate slide and audio-video point.

The student also sees questions and comments entered by classmates who watched the lecture before her, as well as responses from other students, teaching assistants, and the lecturer. These questions are linked to the lecture content. As she watches a lecture, questions asked during that portion of the lecture are automatically highlighted or "tracked." The content of a question appears in a preview window; if one piques her interest she can jump the presentation to it. As she is watching, she sees a question that nobody has answered. She types a response, which is automatically registered with the system and displayed with the question. The person who posed the question is notified of the reply by email.

Later, she has a question. She selects the "ask question" button, then types a subject header and her question. Afraid that the question may sound uninformed, she makes it anonymous. In addition, she enters the email address of a friend, who may be able to answer it before the TA gets to it. When she saves the question, it is added to a pre-existing shared "discussion" collection and is automatically emailed to the TA alias and to her friend. A TA browsing through his email sees the question arrive and opens the message. The email includes the text of the question along with a URL pointer to the point in the lecture where the question was asked. It also contains enough meta information for a reply to be added to the annotation database, making it visible to students who later watch the lecture.

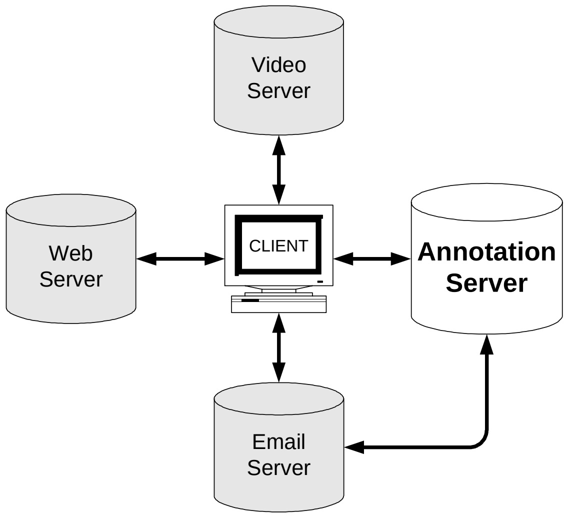


*Figure 1. The Annotation Server fits into a standard multimedia network architecture.*

The student can similarly record personal notes, also linked to the lecture. These are added into a different collection, with permissions set by the student.

## Initial System Design Goals

This scenario helps illustrate some of our initial design goals. In particular, we wanted:

- A general-purpose user interface to support the kind of activity presented in the scenario across a wide variety of web pages containing embedded video (college course and corporate training web pages, news websites like CNN, and usability study pages, to name a few).
- Fine-grained organization and access control structures to support structured sharing among users. We wanted to be able to group annotations into sets and control who could add annotations to the set and who could see annotations in the set. With this simple mechanism, we could create a shared discussion set for a college class, a personal notebook set for each user, a table of contents set for each video file, and so on.
- Close integration with email so that annotations can be sent out via email and replies can be cast as annotations by the annotation server. Email is widely used and well suited for asynchronous collaboration, and with close integration a single conversation can span both mediums.
- Anchoring and display of annotations "in-context" of multimedia content just like notes in the margin of a book, so that we could tie annotations to particular points or ranges along a media timeline.
- Annotations stored external to the annotated content (e.g., the audio-video file) in a separate store. This is critical as it allows third parties to add annotations without having write access to the content. Students should not, for example, be able to modify the original lecture.

At the outset, we believed we could meet all of these goals with a single, well-designed user interface, thus among our non-goals was any kind of user interface specialization ability.

## Original System and User Interface

Given these goals, the system we built was designed to support annotation of multimedia content that appears anywhere on the web. When a user accesses a web page containing video, the browser contacts the web server to get the HTML page and the video server to get the video content. Annotations associated with the video on the web page can be retrieved by the client from the annotation server.

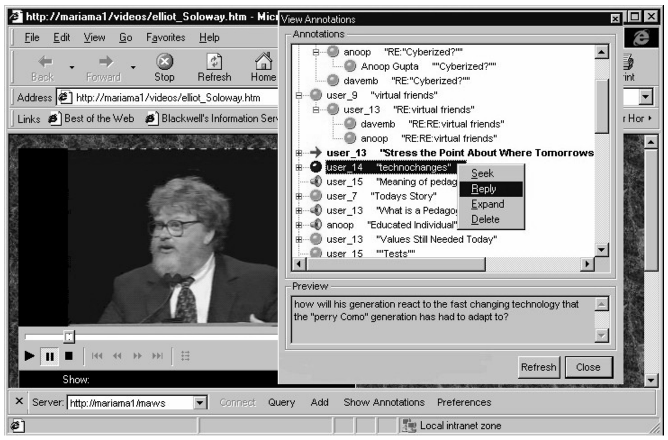
Figure 1 shows the interaction of these networked components. The annotation server communicates with the annotation client via HTTP. Annotations are keyed on the URL of the media with which they are associated. The annotation server communicates with email servers via SMTP, and can send and receive annotations in email.

The video appears in a browser window in Figure 2 and is controlled with a standard media player. Slides appear to the right, synchronized with the video. Annotations made previously appear in a separate window, which in the figure is above the slide frame. Indented annotations are replies. The red arrow marks the annotation linked to the spot closest to the current position of the video, and the blue annotation has been selected. The preview window shows text of the selected annotation, and if none is selected it will show the text of the nearest annotation. Various controls appear at the bottom of the display and in a menu summoned with a mouse click.

When replying to an annotation or adding a new one, a viewer has a choice of making a text or voice annotation. Figure 3 shows the respective dialogue boxes.

## LABORATORY STUDIES AND GENERAL INTERFACE IMPROVEMENTS

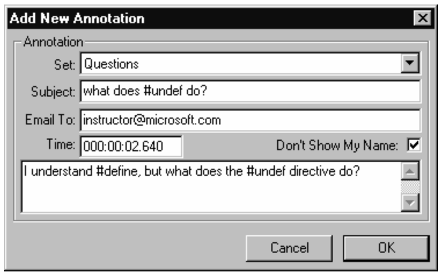
We conducted laboratory studies of the use of the system, detailed in [3]. We examined the use of text and voice


http://mariama1/videos/elliot_Soloway.htm - Mic
File Edit View Go Favorites Help
Back Forward Stop Refresh Home
Address http://mariama1/videos/elliot_Soloway.htm
Links Best of the Web Blackwell's Information Ser
View Annotations
Annotations
Seek
Reply
Expand
Delete
Preview
Refresh Close
Show:
Server: http://mariama1/maws
Connect Query Add Show Annotations Preferences
Local intranet zone

anoop "RE:"Cyberized?""
Anoop Gupta ""Cyberized?""
davemb "RE:"Cyberized?""
user_9 "virtual friends"
user_13 "RE:virtual friends"
davemb "RE:RE:virtual friends"
anoop "RE:RE:virtual friends"
**user_13 "Stress the Point About Where Tomorrows**
user_14 "technochanges"
user_15 "Meaning of pedag
user_7 "Todays Story"
user_13 "What is a Pedago
anoop "Educated Individual"
user_13 "Values Still Needed Today"
user_15 ""Tests""

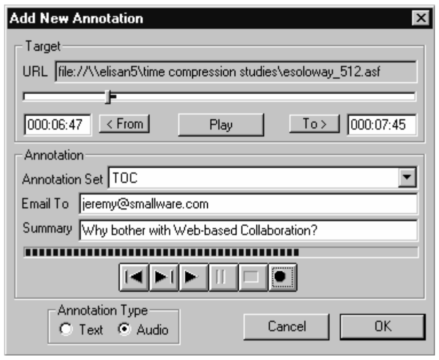
how will his generation react to the fast changing technology that the "perry Como" generation has had to adapt to?

*Figure 2. The first interface. An annotation window appears over a browser window in which a video plays.*



Set: Questions
Subject: what does #undef do?
Email To: instructor@microsoft.com
Time: 000:00:02.640
Don't Show My Name: ✓
I understand #define, but what does the #undef directive do?

*A: text annotation.*



URL file://\\elisan5\time compression studies\esoloway_512.asf
000:06:47
000:07:45
Annotation Set TOC
Email To jeremy@smallware.com
Summary Why bother with Web-based Collaboration?
Annotation Type: Text [✓ Audio]

*B: voice annotation.*

*Figure 3: Adding a new annotation.*

annotations, contrasted paper and pencil annotation-taking with the use of the system, analyzed the effects of reading others' annotations on responding and adding new annotations, and got feedback on many aspects of the interface. These studies led to substantial interface changes as well as some evolution of the features.

As shown in Figure 4, the interface was now embeddable in a standard web page, and could easily be modified to fit in with the "look and feel" of the rest of the page. At the bottom of the annotation frame in the lower left were tabs for selecting one of three annotation sets: Contents (perhaps a list of slide titles), Questions (for viewing prior public annotations), and Notes (for viewing personal notes one has taken). At the top of this frame were buttons for adding to the public discussion or personal notes.

Regarding functions, the lab study revealed an unanticipated lack of interest in voice annotations, so we added the ability to configure which annotation media types (text, audio, or web urls) were available to users. Voice annotations were found to be less useful for subsequent viewers since they cannot be previewed as the video rolls. More significantly, though, they cannot be edited and polished the way text can.

People chose to pause the video when adding text annotations, so we made it possible to configure the annotation client to pause the video automatically when an annotation is being added. Curiously, pausing a video to make notes on it more than doubled study participants' viewing time, however all participants reported preferring it to taking notes while the video was playing.

In general, the design of the multimedia annotation software evolved to accommodate more flexible construction of task-specific user interfaces. More details can be found in [4].

Following this iterative design process we had a robust prototype and considered deployment sites. We settled on two domains. One is the education context covered in the scenario we presented earlier. The other is to support analysis of and dissemination of results from usability studies, which are routinely videotaped.

## FIELD STUDY IN 'C' LANGUAGE COURSES

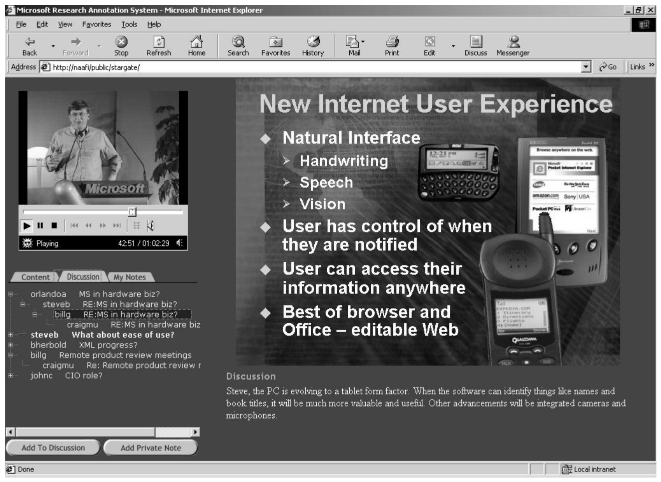
To conduct this study, we observed and videotaped a C programming course taught by our internal education group and attended by employees. We then used the digitized video and slides to conduct two on-demand versions of the course. Students signed up for the courses in the usual way, aware that these offerings would involve an experimental system. They met face to face at the beginning and end of the course and used our multimedia annotation interface to view the lectures and interact in the interim. Study details are described in [4].

Early in our observations we realized that programming language classes make particularly heavy use of online demos that are not picked up adequately by a single video camera focused on the instructor. This motivated a system modification: demos were captured after lectures were taped. Links to the demos were added as annotations that would execute appropriately as a video was viewed. This modification can be seen in Figure 5 on the next page.

### Results

Students in the two on-demand series of classes were generally very positive, citing the convenience. Instructors had fewer time demands but missed the direct contact. The class interaction was at a level close to that of the live class. Their comments pointed out some general and specific aspects of using the system for the C programming class.

At the general level, the students benefited from clarification questions asked in the live class, which they


Microsoft Research Annotation System - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss Messenger
Address
Go Links
Microsoft
Playing 42:51 / 01:02:29
Content Discussion My Notes
Add To Discussion
Add Private Note
Done
Local intranet

http://naafi/public/stargate/

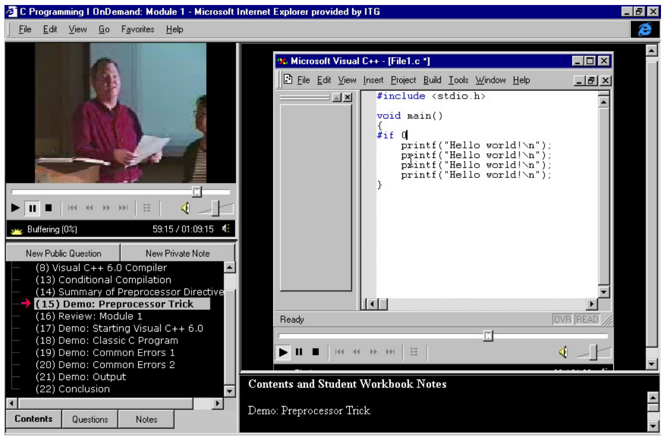
New Internet User Experience
- Natural Interface
  - Handwriting
  - Speech
  - Vision
- User has control of when they are notified
- User can access their information anywhere
- Best of browser and Office – editable Web

orlandoa MS in hardware biz?
steveb RE:MS in hardware biz?
billg RE:MS in hardware biz?
craigmu RE:MS in hardware biz
**steveb What about ease of use?**
bherbold XML progress?
billg Remote product review meetings
craigmu Re: Remote product review r
johnc CIO role?

Discussion

Steve, the PC is evolving to a tablet form factor. When the software can identify things like names and book titles, it will be much more valuable and useful. Other advancements will be integrated cameras and microphones.

*Figure 4. Browser-based interface following lab studies.*


C Programming I OnDemand: Module 1 - Microsoft Internet Explorer provided by ITG
File Edit View Go Favorites Help
Microsoft Visual C++ - [File1.c *]
File Edit View Insert Project Build Tools Window Help
Buffering (0%)
59:15 / 01:09:15
New Public Question
New Private Note
Ready
OVR READ
Contents
Questions
Notes

```
#include <stdio.h>

void main()
{
#if 0
    printf("Hello world!\n");
    printf("Hello world!\n");
    printf("Hello world!\n");
    printf("Hello world!\n");
}
```

(8) Visual C++ 6.0 Compiler
(13) Conditional Compilation
(14) Summary of Preprocessor Directive
(15) Demo: Preprocessor Trick
(16) Review: Module 1
(17) Demo: Starting Visual C++ 6.0
(18) Demo: Classic C Program
(19) Demo: Common Errors 1
(20) Demo: Common Errors 2
(21) Demo: Output
(22) Conclusion

Contents and Student Workbook Notes

Demo: Preprocessor Trick

*Figure 5: C class interface: "Questions" replace "Discussion," among other changes.*

saw online. A number noted that they asked fewer questions than they might have because their questions were already answered, either on the video or with the system. Good questions and the replies could of course be left in place for subsequent classes. One student complained about a detail in the interface, saying "I have questions about C, I don't want to 'Discuss' C." Our choice of the term "Discussion" for the public annotation set might be a good one for seminar-style classes, but not for this one. We subsequently changed this (Figure 5).

More seriously, the flexibility of asynchronous viewing led some students to procrastinate, to others' detriment as well their own, since last-minute viewing is not conducive to creating and sharing comments with the class (e.g. via annotations on the lecture videos). This led us to extend the system to include features that are modeled on approaches to discourage procrastination in some live classes: group exercises and quizzes.

### Follow-up laboratory studies

We used our system's built-in annotation set mechanism to create annotation sets for group projects. Annotations in a group set were shared by the few students assigned to work together if they could not meet face to face. The group's product was reported using the class-wide annotation set. In this case, 'Questions' was no longer an appropriate label for the shared class-wide annotation set, and 'Discussion' was restored.

The system and interface was then used in a laboratory study to gauge the effectiveness of the group project approach [14]. It was found to be successful, and the study also generated a strong demand for a feature not included: The ability to easily copy an annotation from one set to another.

We then extended the interface to include quizzes linked to the video via annotations. Questions appeared at designated points in the video and were potentially useful for self-assessment, grading, or monitoring progress. After responding students could be given a link to the appropriate spot in the lecture to review the question topic. We conducted a laboratory study to explore student reactions and such issues as whether people prefer a question to stop the video or scroll by in the preview window. We found that preferences vary [15].

In conclusion, the field study led to the discovery of a range of interface issues and the identification of additional task-specific features: means for incorporating demos for certain kinds of classes, interface terminology dependencies for different classes, support for group projects and assessment tools of different kinds. Class content, instructor style, and student style created different user interface demands.

## MULTIMEDIA ANNOTATION USE BY USABILITY ENGINEERS

We explored the use of the annotation system with usability engineers (UEs) supporting several product groups. After taping participants in studies, UEs typically review and take notes on the videos, laboriously identify and excerpt segments illustrating key points, and disseminate observations in meetings (where they show the video highlights) and documents (where they do not). We expected that the annotation system would allow them to annotate digitized videos as they review them, providing others with links to relevant portions. Viewers could choose to view material before or after a chosen highlight if need be (which they could not do when the highlights were excerpted).

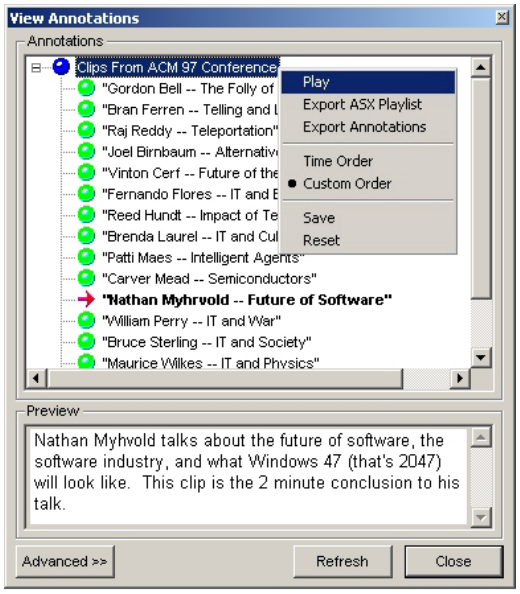
We quickly discovered that this activity represents a conceptual shift from lecture viewing. The shift could be described as going from a timeline-centric point of view to an annotation-centric perspective. The assumption with a lecture is that viewers are on the whole watching from beginning to end, although they can use annotations to jump from point to point. Everything is organized around the video timeline.

Usability engineers, once they have reviewed the tapes and annotated segments of interest, need to collect the segments for presentation together. For example, a usability engineer may annotate three different regions (in different video files) showing examples of users misunderstanding the same menu label. They then need to play the video segments to which their annotations correspond one after another. Thus, rather than watching a single video and its associated annotations, they need to watch a set of annotations and the video segments they annotate.

This led to the development of the playlist feature. A playlist is a sequence of video annotations, possibly from different target videos, which can be played sequentially: when one segment ends, the next will be played.

Our first interface handled playlists in a straightforward manner: right-clicking brings up a menu item that provides access to playlists (Figure 6 on the next page).

This interface quickly proved mismatched to the UE's task, however. Results are communicated in face-to-face review meetings, where this playlist feature would be a fine supplement to a slide presentation. However, comments


View Annotations

Annotations

- Clips From ACM 97 Conference
  - "Gordon Bell -- The Folly of
  - "Bran Ferren -- Telling and L
  - "Raj Reddy -- Teleportation"
  - "Joel Birnbaum -- Alternativ
  - "Vinton Cerf -- Future of the
  - "Fernando Flores -- IT and E
  - "Reed Hundt -- Impact of Te
  - "Brenda Laurel -- IT and Cul
  - "Patti Maes -- Intelligent Agents"
  - "Carver Mead -- Semiconductors"
  - **"Nathan Myhrvold -- Future of Software"**
  - "William Perry -- IT and War"
  - "Bruce Sterling -- IT and Society"
  - "Maurice Wilkes -- IT and Physics"

Preview

Nathan Myhvold talks about the future of software, the software industry, and what Windows 47 (that's 2047) will look like. This clip is the 2 minute conclusion to his talk.

*Figure 6. Playlists: A major conceptual change.*

from team members are collected verbally in such meetings, and the key feature of supporting asynchronous discussion is not useful. UEs circulate their findings via email also, so we thought that perhaps this is where our annotation system could be of potential of use.

We found, however, that UEs were not willing to adapt to use the annotation system -- or felt their teams were not willing -- even when we managed the process of digitizing the videos. They wanted to have the multimedia annotation functionality embedded in the documents or slide presentations that they sent around in email.

This led to the development of a prototype interface that did just that. Figure 7 is an example of a video annotation interface embedded directly in a Word document that has been saved as HTML. The example shown is not from a usability report, but it shows a new arrangement of features including no slide window and a larger preview window.

## SHAKESPEARE COURSE USE AND INTERFACE

It was clear to us at this point in our research that, contrary to our initial conception, the classroom and usability requirements for multimedia annotation differ quite substantially. We focused on the classroom environment for the next experiment. Peter Donaldson, an MIT professor of Drama, was interested in using our annotation system in a Shakespeare class that is centered on comparative examination of performances of plays, many of which are digitally available.

It became clear that to be acceptable in the film class, the interface required modification. Figure 8 on the next page shows several major feature changes. To compare performances or aspects of performances, a second video window is added. Buttons beneath the video windows provide much finer-grained control of playback than previous interfaces, such as single-stepping forward or backward by frame.

One might consider this to be "gold-plating' the interface, but without these and other features, the system would not have been accepted. It might have been accepted ten years ago, but due to the software's flexibility and users' savvy in this case, there was significant demand to create a specialized interface.

The system was used in class projects during the fall semester, 2000. "In a couple of cases [the annotation system] got students in touch with the films in ways that don't happen with conventional essays," wrote Donaldson [7].

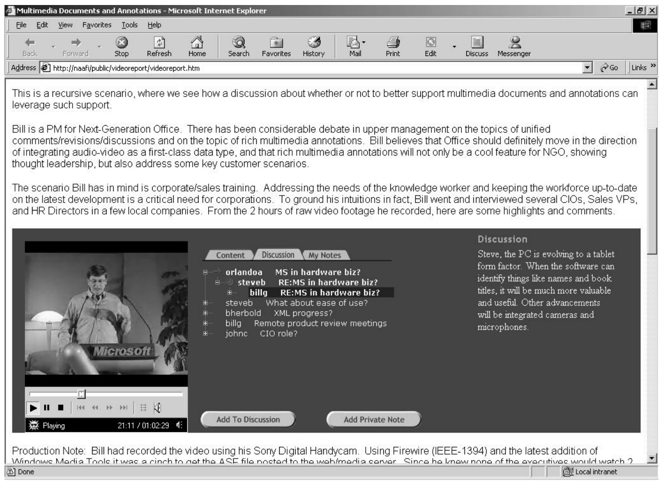
Although this outcome is very exciting, the experience has also proved to be a source of concern. It required considerable effort to develop the interface for the film class, and that interface is not likely to be useful for other classes. It may be useful for other *film* classes, but even that is not a foregone conclusion, since instructors' teaching styles and class format differ significantly.

## PLATFORM REQUIREMENTS

Our initial hope that a single multimedia system interface would capture enough to find broad applicability has so far not been borne out. Wherever we have looked we have found new and often orthogonal, perhaps incompatible, interface requirements.

Are we involved in a process for defining a range of features that can eventually be brought into a single package that many users can adapt to their purposes, or should we focus on building a platform or toolkit that others can use to design specific interfaces that will vary considerably based on domain and approach?

So far, all indications point to the latter. There are clearly commonalities shared among the different task domains we have studied, however there are enough differences among them to require distinct interface features for each context. A single, general-purpose multimedia annotation interface that incorporates all the features runs the risk of being too complex and too task-*non*-specific to be useful in any context. The most prudent and fruitful direction to head in


Multimedia Documents and Annotations - Microsoft Internet Explorer

File Edit View Favorites Tools Help
Back Forward Stop Refresh Home Search Favorites History Mail Print Edit Discuss Messenger
Address
Go
Links

http://naafi/public/videoreport/videoreport.htm

This is a recursive scenario, where we see how a discussion about whether or not to better support multimedia documents and annotations can leverage such support.

Bill is a PM for Next-Generation Office. There has been considerable debate in upper management on the topics of unified comments/revisions/discussions and on the topic of rich multimedia annotations. Bill believes that Office should definitely move in the direction of integrating audio-video as a first-class data type, and that rich multimedia annotations will not only be a cool feature for NGO, showing thought leadership, but also address some key customer scenarios.

The scenario Bill has in mind is corporate/sales training. Addressing the needs of the knowledge worker and keeping the workforce up-to-date on the latest development is a critical need for corporations. To ground his intuitions in fact, Bill went and interviewed several CIOs, Sales VPs, and HR Directors in a few local companies. From the 2 hours of raw video footage he recorded, here are some highlights and comments.

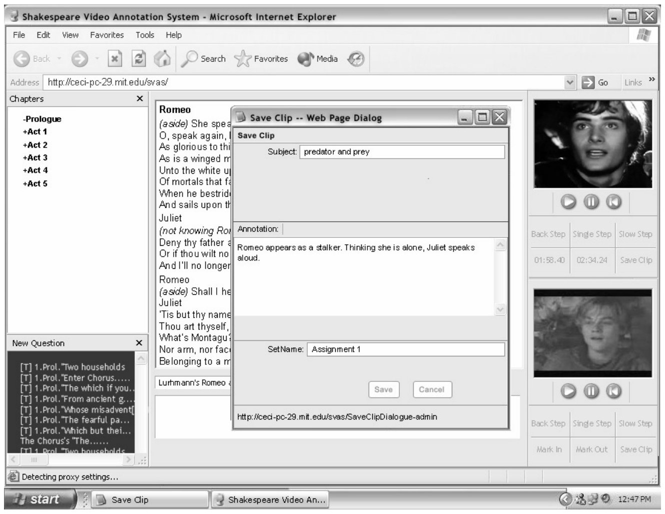
- **orlandoa MS in hardware biz?**
  - **steveb RE:MS in hardware biz?**
    - **billg RE:MS in hardware biz?**
- steveb What about ease of use?
- bherbold XML progress?
- billg Remote product review meetings
- johnc CIO role?

Discussion

Steve, the PC is evolving to a tablet form factor. When the software can identify things like names and book titles, it will be much more valuable and useful. Other advancements will be integrated cameras and microphones.

Production Note: Bill had recorded the video using his Sony Digital Handycam. Using Firewire (IEEE-1394) and the latest addition of Windows Media Tools it was a cinch to get the ASF file posted to the web/media server. Since he knew none of the executives would watch 2

*Figure 7. Annotation system embedded in a document.*


Shakespeare Video Annotation System - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://ceci-pc-29.mit.edu/svas/
Go Links
Chapters
-Prologue
+Act 1
+Act 2
+Act 3
+Act 4
+Act 5
New Question
Back Step Single Step Slow Step
01:58.40 02:34.24 Save Clip
Mark In Mark Out Save Clip
Save Cancel
Detecting proxy settings...
start
Save Clip
Shakespeare Video An...
12:47 PM

Save Clip -- Web Page Dialog

Save Clip

Subject: predator and prey

Annotation: Romeo appears as a stalker. Thinking she is alone, Juliet speaks aloud.

SetName: Assignment 1

http://ceci-pc-29.mit.edu/svas/SaveClipDialogue-admin

Romeo
(aside) She spea
O, speak again, I
As glorious to thi
As is a winged m
Unto the white u
Of mortals that fa
When he bestrid
And sails upon th
Juliet
(not knowing Ro
Deny thy father a
Or if thou wilt no
And I'll no longer
Romeo
(aside) Shall I he
Juliet
'Tis but thy name
Thou art thyself,
What's Montagu?
Nor arm, nor fac
Belonging to a m

Lurhmann's Romeo

[T] 1.Prol."Two households
[T] 1.Prol."Enter Chorus.....
[T] 1.Prol."The which if you..
[T] 1.Prol."From ancient g....
[T] 1.Prol."Whose misadvent[
[T] 1.Prol."The fearful pa...
[T] 1.Prol."Which but thei...
The Chorus's "The......
[T] 1.Prol."Two households

*Figure 8. The Shakespeare interface: Understanding drama by comparing performances.*

has been toward a generic annotation platform, on top of which task-specific user interfaces can easily be fashioned with a modicum of programming skill.

There are general lessons to be learned from our experience. First, we have distilled several general requirements for a more generic software platform for the support of multimedia annotations:

- *Thorough support for common activities*. The most common annotation functions -- such as creating, saving, retrieving, and deleting annotations -- should be the easiest to incorporate into an interface.
- *Extensibility and customizability* at both the interface and platform levels. For instance, designers should be able to extend an annotation's schema to accommodate task-specific features like voting, logging the number of times the annotation has been read, assigning an annotation "type" ("comment," "question," etc), or controlling annotation status ("open issue," "resolved," etc).
- *Storage flexibility*. Designers should be able to store annotations in a variety of configurations. For personal annotations, it may be important to store annotations in the video or audio file itself for portability purposes. For shared annotations on read-only media, annotations may be stored in a separate database. Storing annotations in one facility should not preclude transferring them to another.
- *Universal annotation support*. Ultimately, a general-purpose annotation platform should support annotating any media type with any other media type. Many of the problems encountered with annotations on video and audio apply to annotations on text, images, and complex composite presentations.
- *Interoperability* among task-specific interfaces. Annotations made in one interface based on the platform should be transferable to another interface based on the platform with minimal effort.

These requirements have continued to inform our work, and we are currently developing a much more powerful and flexible annotation platform combined with a more flexible interface toolkit.

Secondly, and more importantly, the laboratory studies and field deployments we have conducted over the past several years have specific implications for the designers of multimedia systems, and potentially for two other groups: third parties who would tailor such platforms for specific domains, and the users of resulting applications. We discuss these implications in the next section.

## GENERAL DISCUSSION

One way to look at our experiences is that we are exploring the requirements for a relatively new kind of multimedia system. Annotated multimedia is a domain with some research and product precedents, but it is not yet widely embraced. Field studies have enriched our understanding of what such systems should support, laboratory studies have refined the interaction design and interface features.

This is accurate as far as it goes, but there is more to the picture. It is unclear that the general-purpose application that we initially envisioned is realizable. Field study approaches such as contextual design [5] are generally undertaken with the goal of developing a widely-used product; usability studies are undertaken to refine the interaction design. Based on our experience, in designing systems such as this one, we will do better to think less in terms of delivering a generic system and more in terms of providing a toolkit or application development environment. This was not what we set out to do.

Interfaces based on a more generic annotation platform could be developed by third parties or could be constructed by technical support groups, working with individual instructors or other users. In this sense it might resemble Lotus Notes, which is less a generic application than an application development environment that requires the involvement of professionals to create databases and views.

These studies suggest a general trend to be considered by designers of applications targeted at a wide range of people. It may be, as we have found, that an application that seems basic and general enough for wide use without modification will encounter resistance, and context-specific requirements will emerge. If designers do not anticipate this, they may not build in the flexibility that will permit shifting to a toolkit approach.

This trend is arguably due to two factors. New applications are striving to support more complex, specialized activities, which are more subject to requirements based on domain or stylistic differences. At the same time, the people being supported are increasingly aware that software is highly flexible and are less inclined to adapt, more inclined to be annoyed by having to work around an interface.

Software specialization is not new. To support vertical markets, people build novel interfaces to applications such as Microsoft Word and Excel. The difference is that generic off-the-shelf Word and Excel are useful applications to many people just by themselves. Customized versions came later. A generic multimedia annotation system may not be appreciated by many, just as there is not really a generic Lotus Notes application.

The process of analyzing general platform requirements from task-specific contexts and then driving them into a platform is crucial because it makes quickly developing new task-specific interfaces easier. As users become more savvy and start to demand more customization, providing more generic platforms that can be easily specialized will become easier and cheaper than providing one-off task-specialized interfaces.

## CONCLUSION

We have described experiences with a series of prototype multimedia annotation systems over several years, and our growing recognition of the difficulty of creating an interface that will be widely useful, despite favorable responses to our specialized interfaces. The technology will prove useful, but the path to realizing the uses may be through development of a platform that third parties can build upon. This is a common enough approach, but it did not seem appropriate for *this* seemingly simple application.

To support specialized activities carried out by people increasingly aware of the potential of software, and to do so while we are pushed to tighten the feedback loop between user and designer, requires a different approach. From the outset we must think carefully about potential third-party partners in development, rather than assume we can deliver a turn-key system. We must plan a mix of laboratory studies and field studies to determine who will be responsible for the different elements of the design, and how the partnership could work.

## ACKNOWLEDGMENTS

We would like to thank Scott LeeTiernan, Francis Li, Elizabeth Sanocki, Peter Donaldson, Belinda Yung, Randy Hinrichs, David Aster, and others for their contributions.

## REFERENCES

1. Abowd, G., Atkeson, C.G., Feinstein, A., Hmelo, C., Kooper, R., Long, S., Sawhney, N., and Tani, M. Teaching and Learning as Multimedia Authoring: The Classroom 2000 Project, Proceedings of Multimedia '96 (Boston, MA, Nov 1996), ACM Press, 187-198.
2. Apple Computer, 1992. Macintosh Human Interface Guidelines. Addison-Wesley.
3. Bargeron, D., Gupta, A., Grudin, J. & Sanocki, E., 1999. Annotations for streaming video on the Web: system design and usage studies. *Proc. WWW8,* 61-75.
4. Bargeron, D., Gupta, A., Grudin, J., Sanocki, E. & Li, F., 2001. Asynchronous collaboration around multimedia and its application to on-demand training. *Proc. HICSS-34*, CD-ROM, 10 pages.
5. Beyer, H. & Holtzblatt, K., 1998. *Contextual design.* Morgan Kaufmann.
6. Bøgh Andersen, P., 2001. Elastic systems. *Proc. Interact 2001*, 367-374.
7. Donaldson, P., personal communication, January 2001.
8. Dourish, P., 1995. Developing a reflective model of collaborative systems. *ACM Transactions on Computer-Human Interaction, 2,* 1, 40-63.
9. Eisenberg, M. & Fischer, G., 1994. Programmable design environments: Integrating end-user programming with domain-oriented assistance. *Proc. CHI'94*, 431-437.
10. Fischer, G. & Girgensohn, A., 1990. End-user modifiability in design environments. *Proc. CHI'90,* 183-191.
11. Gentner, D. R. and Grudin, J., 1996. Human interface design models: Lessons for computer human interfaces. *IEEE Computer, 29*, 6, 28-35.
12. Grudin, J., 1990. The computer reaches out: The historical continuity of interface design. *Proc. CHI'90,* 261-268.
13. Grudin, J. and Norman, D.A., 1991. Language evolution and human-computer interaction. *Proc. 13th Annual Conference of the Cognitive Science Society,* 611-616.
14. LeeTiernan, S. and Grudin, J., 2001. Fostering engagement in asynchronous learning through collaborative multimedia annotation. *Proc. INTERACT 2001, 472-479.*
15. LeeTiernan, S. and Grudin, J., unpublished manuscript.
16. Mackay, W., 1990. Users and customizable software: A co-adaptive phenomenon. Ph.D. thesis, Sloan School of Management, MIT.
17. McClintock, M., personal communication, 2000.
18. Mørch, A., 1997. Three levels of end-user tailoring: Customization, integration, and extension. In *Computers and Design in Context.* M. Kyng & L. Mathiassen (Eds.), pp. 51-76. MIT Press.
19. Reiss, S.P., 1990. Connecting tools using message passing in the field environment. *IEEE Software*, July, 57-66.
20. Teitelman, W. & Masinter, L., 1981. The Interlisp programming environment. *Computer, 14,* 4, 25-34.