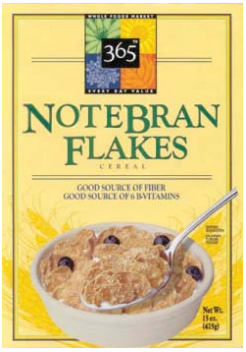



COMPUTER SCIENCE 160 · USER INTERFACES

# NoteBran

## FINAL REPORT

DUE 6 DECEMBER 2004 · GROUP 6


| | |
|---|---|
| **Steven Chan** | Discussion Panel Redesign, Report Compilation and Editing |
| **Tingting Huang** | Login/Menu Redesign, Report Compilation |
| **Vlad Kaplun** | Toolbar/Review Redesign, Project Integration, Report |
| **Alexey Kudashev** | Report |
| **Jennifer Tsang** | Slide/Open File Redesign, Project Integration, Report |

# TABLE OF CONTENTS

# OVERVIEW

## TARGET USER GROUP

Our project is targeted towards university (1) instructors who prepare and give lectures in a large classroom setting, whose lecturing materials involves extensive use of visual diagrams and multimedia representations, and (2) students who take notes on a regular basis, participate in class discussion and group interaction, and students who have difficulty attending the lectures occasionally.

According to the information collected from our contextual inquiries and heuristic evaluations on our low-fi prototype, instructors are already familiar with Microsoft PowerPoint as the de-facto presentation authoring tool. But there is a relative paucity of electronic note-taking tools are available to students. Due to the complexity of interface implementation, time constraints, and our limited budget[1], we avoided trying to replace PowerPoint's functionality and instead place the emphasis on our design on one user group: the students. Most of the usability issues for both user groups — instructors and students — overlap, so incorporating functionality for instructors merely involves building upon our existing user interface.

## PROBLEM

Throughout the learning process, knowledge is transferred from instructors to students via lecturing. Based upon our interviews, instructors spend an excessive amount of time preparing for lectures. During lectures, they use visual diagrams and handwritten text to supplement their speech. On the receiving end, students must pay attention to the lecture contents while writing down as many notes as possible for future reviewing. In addition, all course-related materials are scattered about various media: students juggle paper handouts, handwritten notes, newsgroup postings, and online documents. This eclectic jumble of information sources makes information retrieval time-consuming, and renders the teaching and learning processes less efficient.

## SOLUTION OVERVIEW

NoteBran, a Tablet PC-based application, aims to solve the problem by integrating a variety of different mediums (webcast, newsgroup, Black Board, black lightening notes, and course websites) and presentation techniques (video clips, images, chalkboard, PowerPoint, transparency, etc.) into one central access point. The design also employs learning principles by encouraging collaborative discussion and providing feedback to students. The seamless interface will allow smooth transfer of learning materials from the instructors to the students, produce more interaction among students, and satisfy students' note-taking and organizational needs.

[1] Zero U.S. dollars.

## TASKS

### TASK 1: REVIEW SLIDES AND WEBCAST AFTER LECTURE (EASY)

From our contextual inquiry, we found that students prefer the webcast to catch up with missing materials if they are late to the lecture or miss the lecture completely. Even students who attend lectures regularly still watch webcasts to clarify difficult concepts and to get a better understanding of presented material. This task is chosen to satisfy students' needs and to make learning process more convenient and remote. The students perform the task by accessing archived webcasts stored in a central database.

### TASK 2: TAKING NOTES DURING LECTURE (MODERATE)

Results from our contextual inquiry showed that students find it difficult to digest lecture materials while taking extensive notes during lectures. We wanted to reduce note-taking to a minimum, so that students can concentrate more on the lecture. The students can take notes on the lecture slides, which are uploaded automatically at the beginning of the lecture, using any annotation tools. From pilot usability tests, we noticed that students have limited exposure to pen input devices. An adjustment period might be required to switch from using regular computers, paper, and pens to tablet PC's.

### TASK 3: ASKING A QUESTION DURING LECTURE (HARD)

Due to the size of the class and limited lecture time, not every student can get his/her question answered by the instructor, or engage in group discussion with each other. But the principles of feedback, reflection, and revision is important, which further encourages metacognition and more effective learning. Thus, we decided to support participation in discussion during lecture, hoping that by building a community of students and instructors, students can receive early feedback, reflect on the concept they are learning, and make revisions to their knowledge if necessary. The students will be able to participate in discussions by an exchange of feedback through a digital discussion board.

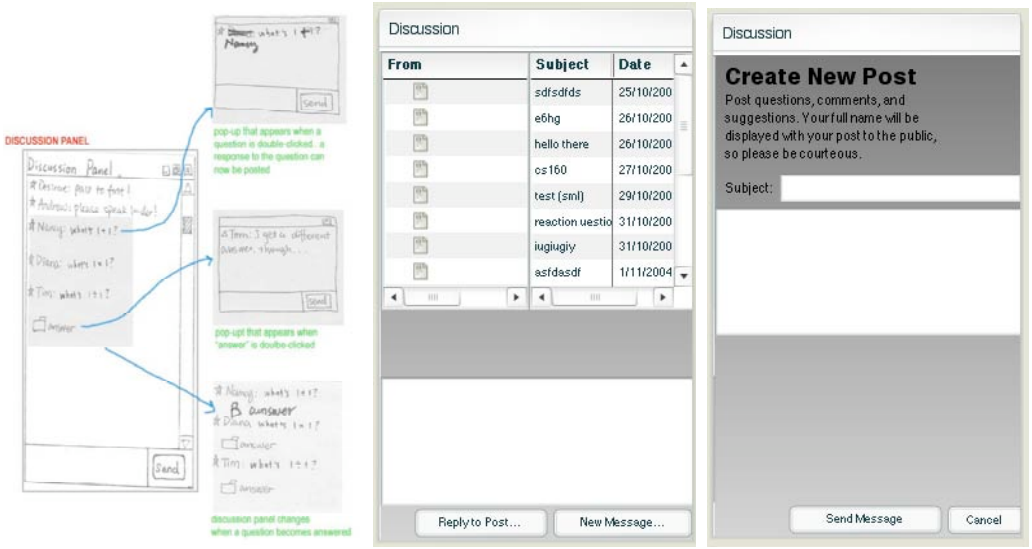



Figure 1: Evolution of the Discussion panel, from low-fi prototype to the initial interactive prototype.

Our final Discussion Panel (please see figure 1) evolved from an initial Interaction sidebar. In our low-fi prototype, it started out modeled after a standard chat interface. A series of messages appeared in the window with an area below for the user to type or write text to send to everybody. The instructor could answer a post by double-clicking the question, which produces a second window pop-up with the question content and area to send the answer. After the low-fi usability tests, we changed the Discussion panel into a more refined message hierarchy akin to newsgroups, rather than an instant messaging interface. In the new system, threads can be expanded and compressed without having new windows pop up. *From our Heuristic Evaluations, we also found the need to implement visibility of system status, since the Discussion Panel took a considerable amount of time to load and to post messages.* In retrospect, we initially had little focus on the Discussion Panel. However, through the observations of the user tests as well as our own reconsideration of the problem/solution, the Discussion Panel became a more integral and significant feature of our design because it promoted communication between the students and instructor.

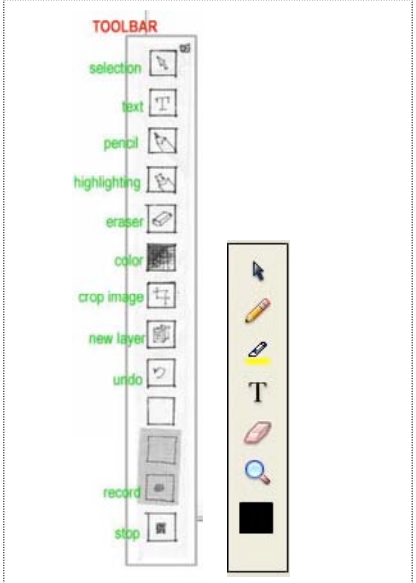



Fig. 2: Evolution of the Toolbar.

Throughout most of our iterations, the **toolbar interface** (please see figure 2) was consistently modeled after that of Adobe Photoshop. For our initial sketches and low-fi interface, the bottom of the toolbar included start/stop buttons to control recording lectures for webcast. Since the participants of our first user testing found it confusing to place the video control buttons with the drawing tools, we placed these

buttons alongside the slide thumbnails in our hi-fi prototype (please see figure 3).

Fig. 3: Location of play/stop buttons, beside the thumbnails.

We had some semblance of **layers** in our initial user interface, but at the time, we didn't lay out a detailed design for how the user would add and navigate through different layers (please see figure 4). For our low-fi prototype, each layer had a tabbed title that the user would click to bring the layer to the front. Right-clicking on the top layer brought up a context menu that contained options to hide, show, project, add, and delete that layer. However, we concluded from our user tests that not all users understood the concept of layers, so we abstracted the idea and implemented a default "Notes" layer with an eye icon that would indicate hiding and showing a layer.

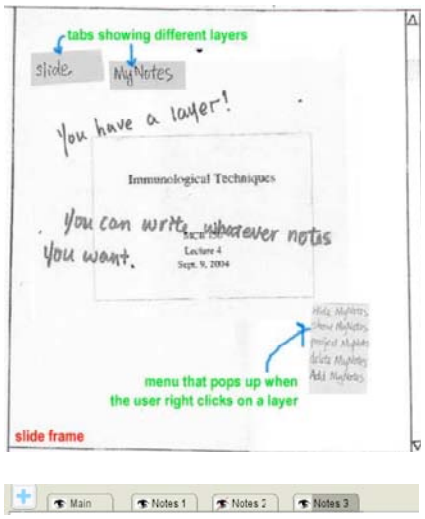



Fig. 4: Layers in our low-fi vs. the interactive prototype.

In our initial sketches and low-fi prototype, one of the prominent features was the **Slide Library** (please see figure 5). This window started out as an upper library containing slide files and a lower library with media files such as images, audio, and video. Three columns organized each part of the library — course, semester, and date. We intended the library to be a convenient means of organizing files, particularly for instructors to create slides (dragging in images and other media). However, students in our low-fi prototype testing found it unnecessary. And when we decided to shift our focus away from the instructor user group, we found even less of a need for the Slide Library and decided to remove it from the interface.

From our low-fi usability tests with the student and instructor user groups, we found that many of our participants complained about having too many irrelevant features. Students commented that it was confusing to have the Slide Library when it wasn't necessary for their specified tasks. As a result, we decided to create separate **modal-like presets** for students and instructors that would toggle the visibility of certain authoring panels (please see figure 6, on next page). Upon entering the program, a window greeted the user asking him/her to specify the preset. However, after deciding to focus on only the student user group and removing the Library Panel, we found that there wasn't much of a distinction between the different modes. We removed the preset modes from our final iteration and instead implemented a toggling functionality in the "Window" option of the

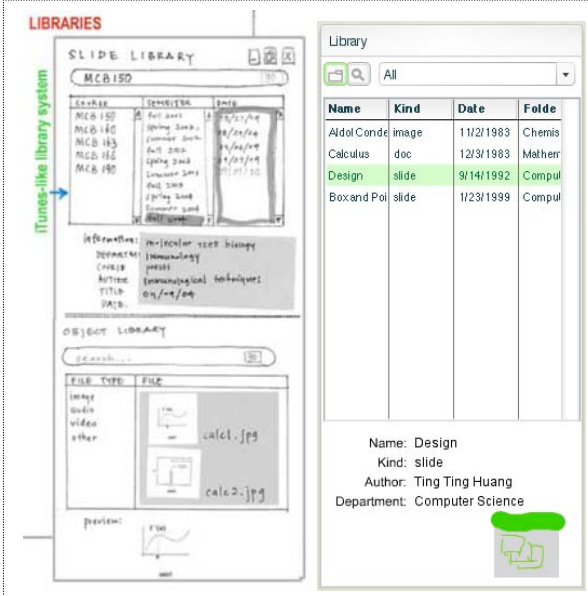



Fig. 5: The Slide Library, deprecated.

main menu.

## WHICH EVALUATION TECHNIQUE WAS MOST VALUABLE?

Each evaluation technique played a role in the process of improvement of our program. The general tendency was that in the earlier stages of project development, evaluations were the most successful in **generating ideas**, whereas in the final stages of producing the actual product, it was most successful in **pinpointing our mistakes**.

We feel that the most valuable technique in the final stage was **user testing** of our interactive prototype. By this time, most of our project was complete so we could receive the constructive criticism to polish our product and correct any UI oversight. Users were harsh because technological limitations of our development platform, Flash, prevented us from fully delivering on the ideas generated in the initial stage.

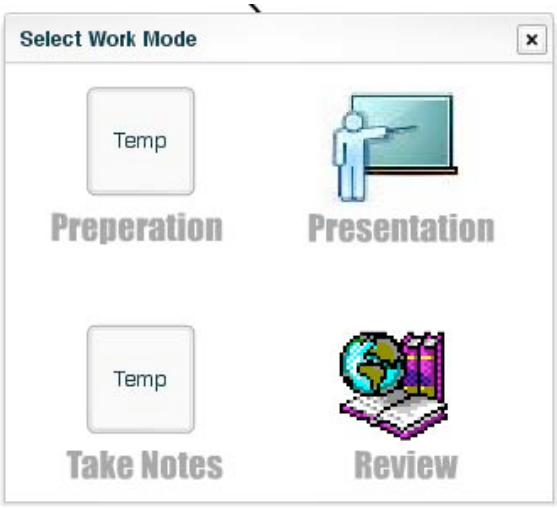
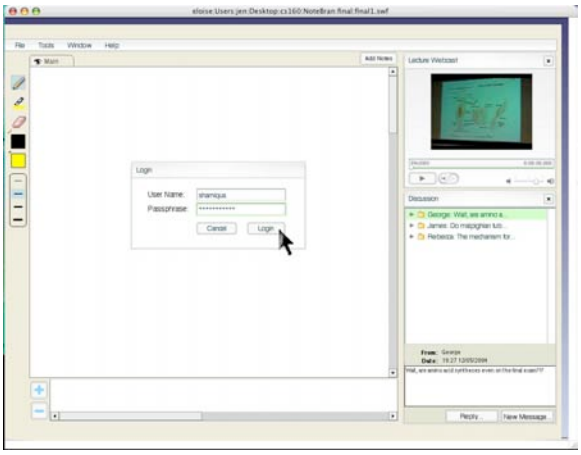
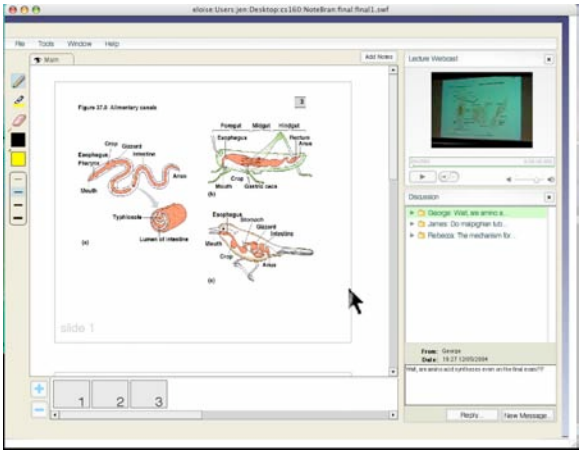



Fig. 6: Work modes. This was deprecated in the final product, in favor of a user interface integrating all of the elements.

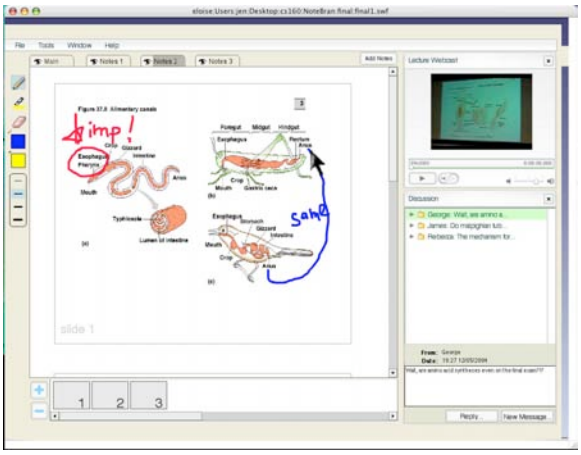
## SCENARIO 1: TAKING NOTES DURING LECTURE



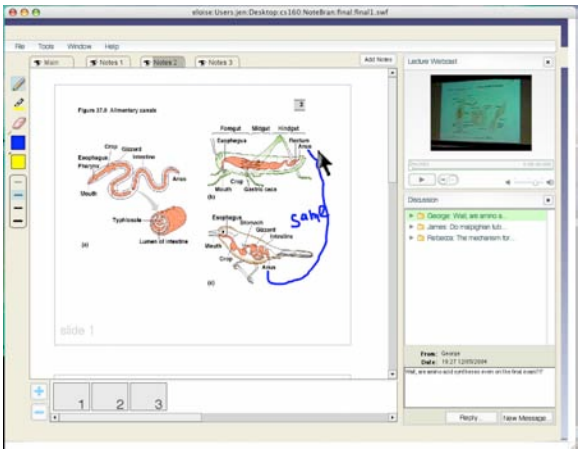
1. While sitting in the lecture room, the professor's lecture slides automatically appear on the student's tablet PC after the student logs in successfully.



2. The student then selects any tools from the toolbar for note taking



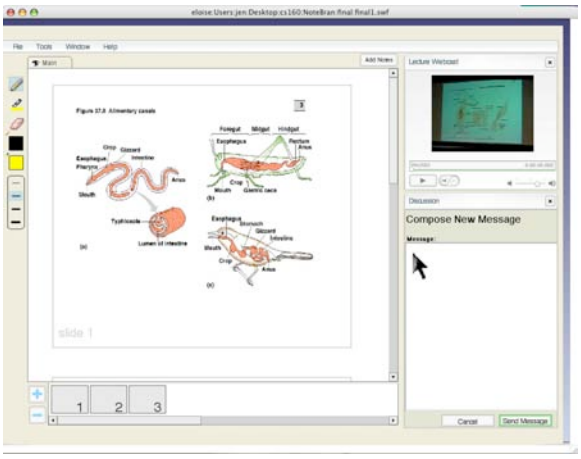
3. The student adds notes for additional note space.



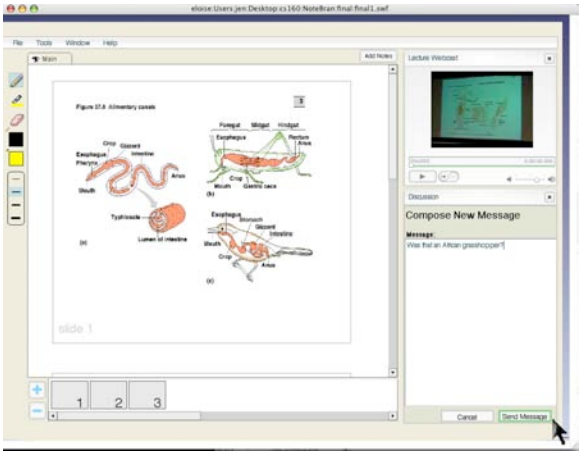
4. The student can hide notes for clarity(screenshot of the invisible eye, and disappearance of the notes in the main canvas)

# SCENARIO 2: PARTICIPATING IN DISCUSSION DURING LECTURE

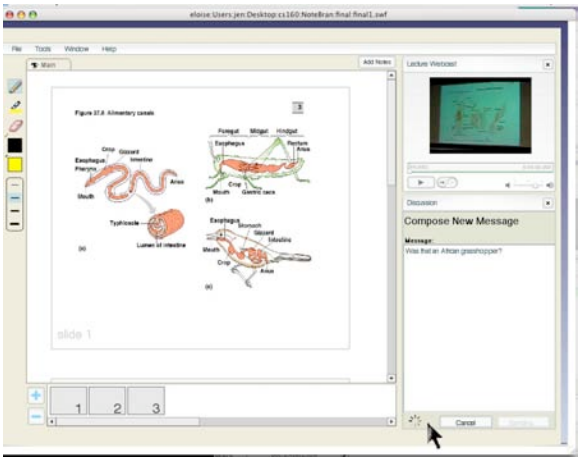
### 1) POST A NEW MESSAGE



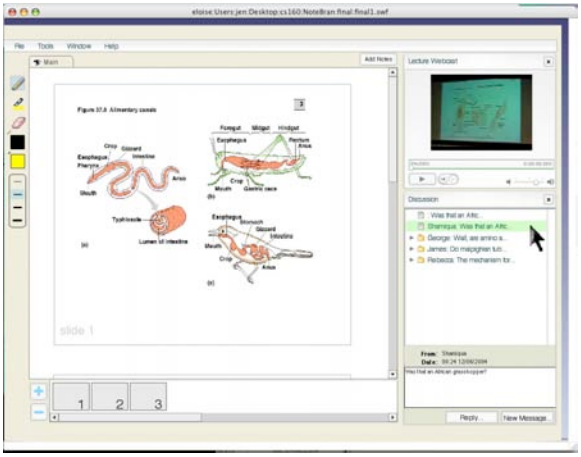

1. The student posts a new message by clicking "New Message…" in discussion panel.




2. The discussion panel shifts to Compose New Message. After typing in the message contents, the student clicks "Send Message."




It sends the message. Afterwards,

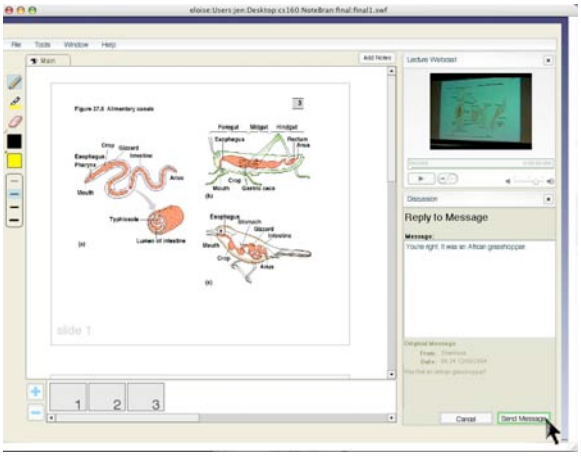


3. The new message appears in the discussion panel, highlighted and selected.

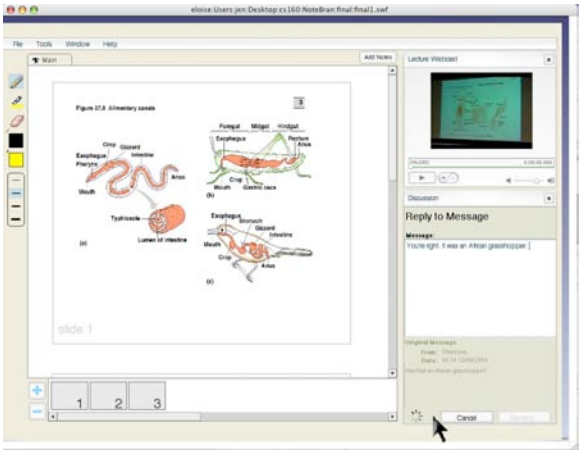
## 2) REPLY TO AN EXISTING MESSAGE

1. The student selects a message to reply to from the list of messages.

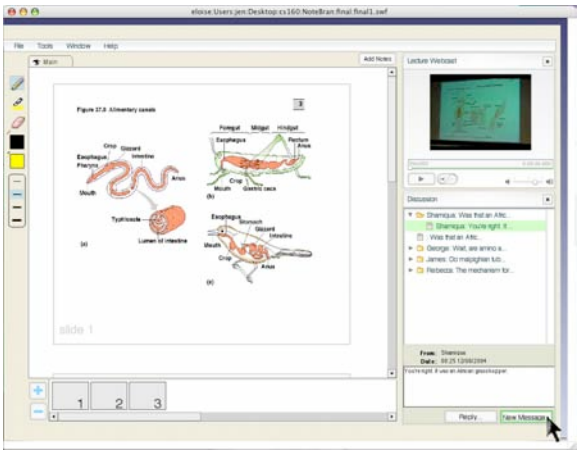
2. The student clicks the “Reply…” button.



3. The discussion panel changes to “Reply to Message.” After typing in the message contents, the student clicks “Send Message.”

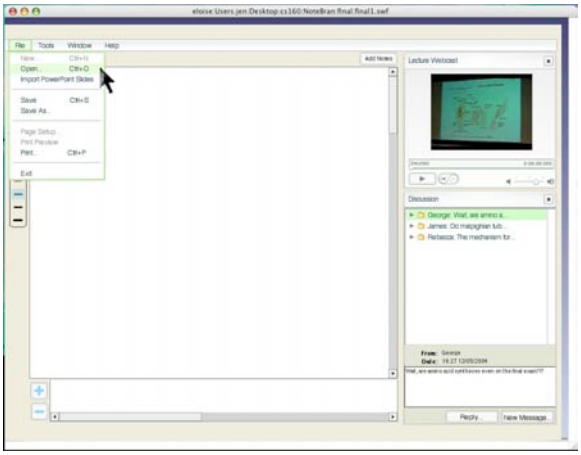


It sends the message. Afterwards,

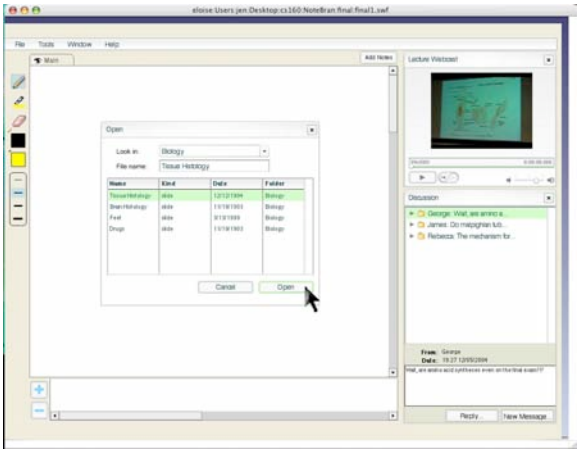


4. The reply appears in the discussion panel, highlighted and selected.

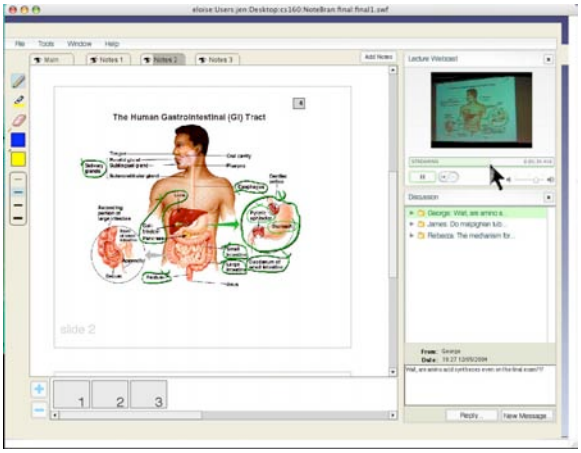
## SCENARIO 3: REVIEW LECTURE SLIDES AND WEBCAST



1. The student opens the lecture slides with webcast by clicking File→Open.



2. The student can search the lecture slides with webcast by looking through folders, typing the filename, and clicking on the lecture list.




3. After selecting the lecture, the student manipulates the webcast with standard video controls in the Lecture Webcast.

## THE FINAL DESIGN

A login window (figure 7) will prompt the user for username and password. This is a verification of the user's status for accessing course materials and discussion posting.

Figure 7: Log-in panel

The menu bar along the top of the interface contains basic functions. Upon clicking, "File," "Tools," "Windows," and "About" each drop down a menu specific to the category.

### THE FILE MENU

The users can search and open specific file with "File→Open", save changes to files with "File→Save" or "File→Save As…", and print file contents with "File→Print". When the user closes the application by selecting "File→Exit" or clicking "☒" on the title bar, a window will pop up to prompt the user to save the changes if he/she hasn't done so. This prevents errors, so users won't lose any important changes to the files by accident.

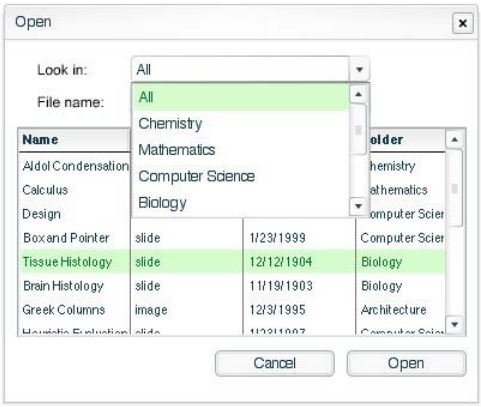


Figure 8: Open dialog box

### THE WINDOW MENU

A general observation from our low-fi usability testing was that users didn't want to see the features that were not relevant to the specific tasks. The menu options in this window — Lecture Webcast, Discussion Panel, and Slide Thumbnails — offer the users the option to hide the corresponding window. This grants the user freedom and control. These options also promote aesthetic and minimal design, so the user can choose to cut down on clutter.

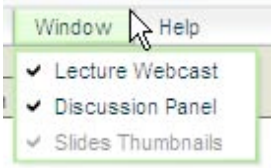


Figure 9: Window menu

### THE HELP MENU

Following domain and task modeling, we also included links to task-specific help. Students can obtain detailed procedures on how to perform three tasks. By selecting NoteBran Support and About NoteBran, users can obtain additional assistance and information about NoteBran.

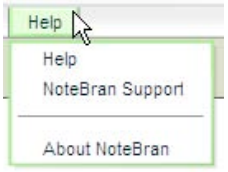


Figure 10: Help menu

### LAYERS FUNCTIONALITY

Layers are shown through tabs above the main slide area. Each tab contains the name of the layer and a graphical eye button that toggles between visible and hidden. An "Add Notes" button to the right of the slide area allows the user to add layers onto the slides. As the button is pressed, a tab corresponding to the new notes appears; up to six layers can be created.

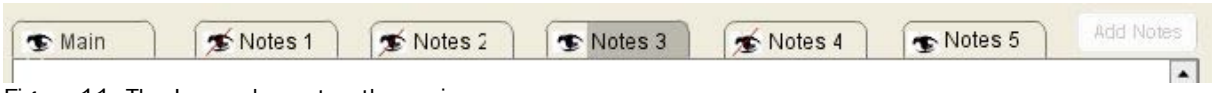


Figure 11: The Layers bar, atop the main canvas

When all the possible layers have been added, the “Add Notes” button is grayed out and can no longer be pressed.

The intent of the layers function is to *allow users to create separate transparencies of information*. The student may have one layer for the instructor’s notes and one layer for his/her own notes. It may also be useful for students to share notes with classmates. Students can then have an organized system of maintaining separate layers of information while keeping a sequential order and correlation between all the notes.

### SLIDE THUMBNAILS AND NAVIGATION

The main slide area is set in a vertical scroll panel. Below this is an area for the slide thumbnail timeline. Each thumbnail is a button that adds an additional means of navigating the slides. By clicking any one of the thumbnails, the user is brought to that slide in the main canvas area. Alongside the slide thumbnails are two “+” and “–” buttons that allow the user to add or delete slide pages from the file. As slide pages are added, a thumbnail is added to the timeline and a slide page is added to the main slide area.

The slide navigation is designed to allow multiple methods of completing the same task, providing flexibility to the user. For example, the user can either scroll up and down the main slide area or click the thumbnails below to immediately jump to a slide.

### DISCUSSION PANEL

The final design of the Discussion panel borrows elements from existing newsgroup and email clients. A "Loading" screen with an animated busy icon (akin to that of Mac OS X) first appears to inform the user that it's loading all of the messages. We opted to forego the progress bar because the loading time was quite short.

The user is then presented with a hierarchical list of messages and a single message details pane. If the user clicks on any message, the complete message, with details of its origin and creation time, is displayed in the details pane. The user can also collapse and expand message hierarchies to hide topics, giving the user some measure of freedom as to which messages were displayed.

New messages can be created simply by clicking on the "New Message..." button. Or, if the user wants to create a message in response to an existing message, he or she can click on the "Reply..." button. Both of these buttons take the user to a message composition screen within the Discussion panel, with a single text editing box. The user can type or handwrite text (with the Microsoft Tablet PC handwriting recognizer) into the text editing box, and click "Send Message" when he or she is done. The "Send Message" button dims and an animated busy icon appears to inform the user that their request is being processed. At any point in this window, the user can click "Cancel" to go back to the discussion panel.

Figure 12:
Discussion panel

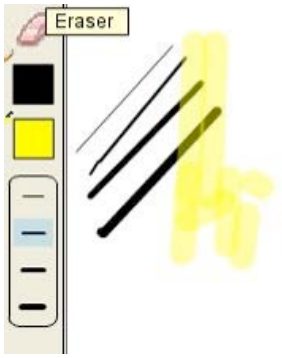



Figure 13: Toolbars, with tooltips and options, were larger than normal 16x16 icons to make pen and mouse targeting easier

### TOOLBAR

The toolbar features the available drawing tools, namely, pencil, highlighter, and eraser. It has two color selectors for the pencil and highlighter. Since the tools are represented only as symbols (icons), each has a tool tip that appears when the user moves their cursor over that tool. The tool tip states the name of the tool, so that the user can recognize these symbols. Additionally, a stroke width selector corresponding to the currently selected tool appears on the tool bar.

### WEBCAST

The Webcast interface is much like any video player. It has play, pause, and stop buttons as well as a scrubber to quickly move to any part of the video clip. A webcast corresponding to a particular lecture is intended to load when the lecture file is opened so there is no need for separate file controls within the webcast window. The position of the Webcast window was changed after discovering in our pilot study that users preferred the window in the upper portion of the screen.


Figure 14: Lecture webcast would be synchronized with the user's notes

## FEATURES LEFT UN-IMPLEMENTED

Due to time constraints, standard editing functions such as undo, redo, cut, copy, paste, etc., were eliminated. Keyboard shortcuts for accelerating program usage, animated professor's blackboard writing, drawing, picture editing tools(Tools screenshot), and text input were also left out, since those features merely complement our existing features but are not necessary for completing the tasks. The complexity of implementing resizable windows using Flash makes it near-impossible and impractical for us to attempt this basic feature. This is a major drawback with our user interface.

The delete slide function is not implemented. As a result, when the user clicks the "–" button at the bottom of the page, a window pops up indicating to the user that the function is not yet implemented. Only four complete slide pages can appear in the main slide area; slide thumbnails, however, will continue to appear upon each press of the "+" button.

Since we shifted our focus towards the student user group and away from the instructor user group, it became unnecessary to fully implement these functions. Our student tasks did not involve creating slides, so it was not necessary for them to add and delete slide, as well as have a large number of slides in the file.

The "File→Open" and "File→Save" options are limited in functionality. For the student tasks, it was only necessary to provide the file-opening option for two specified files. The other filenames appear only to create some difficulty for our test subjects while they performed their tasks.

The discussion panel supports most of the features we were aiming for, but we could not polish it in time for the final release. *Error catching* is a problem; we did not account for things like broken network connections and synchronizing errors. We also did not have time to confine Discussions to a particular lecture, nor did we have time to implement status icons for each message indicating which messages were already read. One user tried drawing in the screen, and at that point, we realized that users still needed to

use pen input for discussing things; after all, they *were* using a Tablet PC. However, we also could not implement a whole whiteboard system in Discussion.

## TOOLS USED

Although we used a mix of tools, the good majority of development was done with Macromedia Flash MX 2004. Flash was very suitable for rapid user interface development. No one on our team had knowledge of other GUI libraries, so Flash was the most appropriate choice for a tool because of its (1) short learning curve and (2) cross-platform nature. It allowed us to complete the majority of the interface in a tight two-week time span, and gave us the flexibility to make subsequent revisions with each iteration of testing. And, because a lot of commercial documentation exists for it both online and in print, we initially preferred Flash over other open-source UI libraries (such as wxWidgets, Tk, and qt).

Its visual nature allows us to simply drag and drop widgets into position without having to code it in. Flash also provided us with a comprehensive suite of ready-made "UI Components." Its extensibility also allowed us to use user-created components from online Flash communities.

However, Flash is plagued with performance problems. Its display routines are not optimized, and the components are nowhere near as responsive as native operating system components. Java's Swing libraries run like speed demons in comparison. The platform also seems inconsistent; user interface components, such as the webcast scrubbing tool, do not function at all when used within windows.

Interface problems abound in the Flash IDE. It tries to cater to a variety of audiences, but there is no mistake that Flash's roots are in animation. Its UI Components architecture feel tacked-on, and Actionscript's libraries are nowhere near as rich as that of existing scripting languages (e.g. Python and Perl) and programming languages (e.g. Java). As a result, much of the code feels like a hack. Searching and replacing in a string, for example, requires us to perform FOR and WHILE loops to search through the characters of a string. Macromedia could have implemented a simple function like this in their internal String class, so that it runs faster.

And the program is not appropriate for complex application development. We could not turn to alternative Flash IDE's because none exist, forcing us to use Flash's own (rather problematic) debugger and trace functions. It was often hard to detect problems because Flash *never* outputs non-severe errors. If we invoke functions to nonexistent objects, for example, Flash hums along quite merrily, leaving us scratching our heads.

Finally, Flash MX 2004 is not as extensible as it seems at first glance. Components made for versions of Flash MX, say, could not be used without some hacking. And tweaking this code involved hunting down bits of Actionscript, which can be stored in a variety of places: in external files, in external libraries, in frames, and in objects. Creating *just* the look of the user interface was very easy in Flash, but creating the behavior that drives the interface seemed cumbersome and difficult.

The Discussion interface communicates messages to a server running PHP with XML tools and a MySQL database back-end. This allowed us to realistically simulate the Discussion panel. These tools proved to be fairly flexible for our needs since we had total control over the networking code's behavior, but perhaps this is because many of us were more familiar with PHP than with Flash.

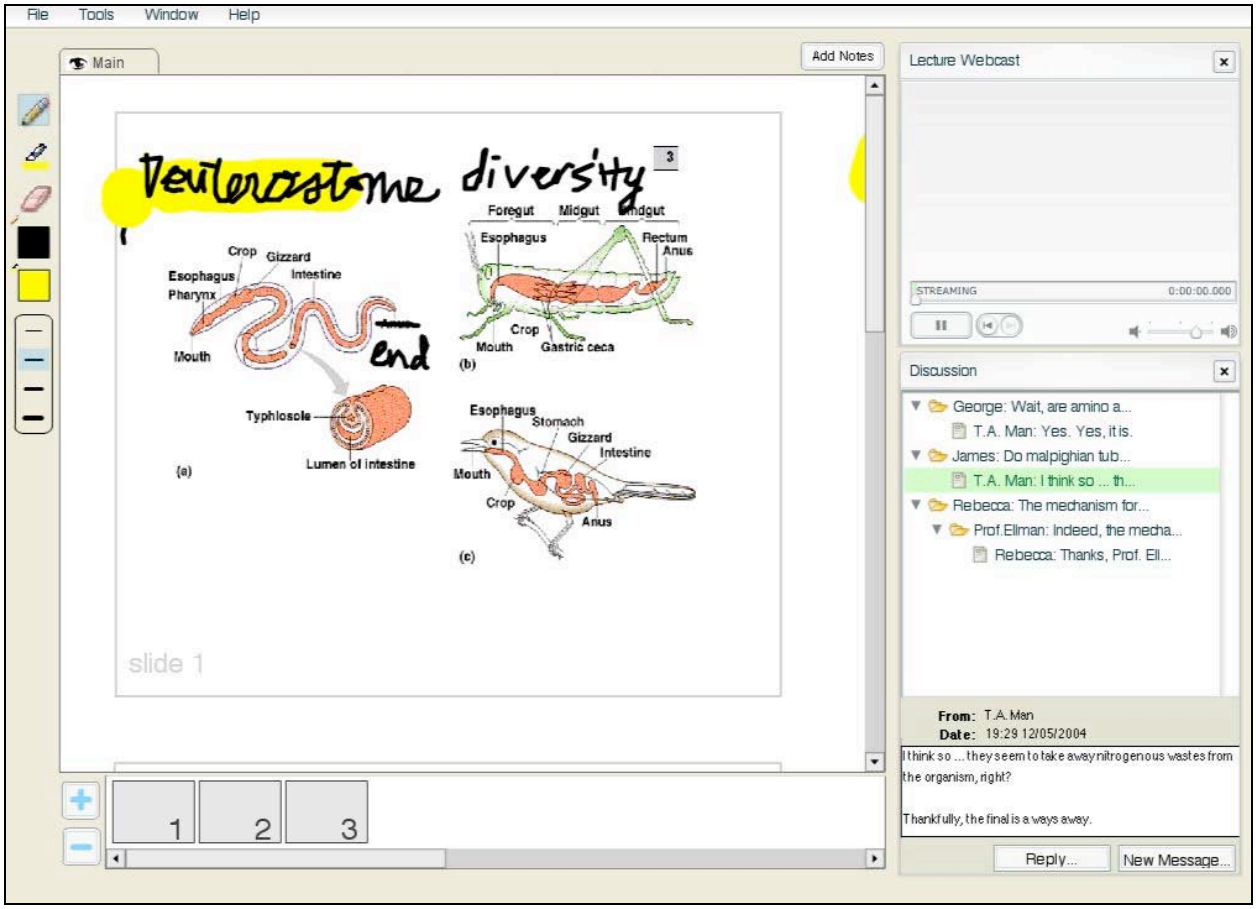



Figure 15: The overall user interface.