Understanding and Promoting Interaction in the Classroom through Computer-Mediated Communication in the Classroom Presenter System

Steven A. Wolfman

A dissertation submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

University of Washington

2004

Program Authorized to Offer Degree: Computer Science & Engineering

University of Washington

Abstract

Understanding and Promoting Interaction in the Classroom through Computer-Mediated Communication in the Classroom Presenter System

by Steven A. Wolfman

Chair of Supervisory Committee:

Professor Richard Anderson
Computer Science & Engineering

Students' active participation in the learning process is a critical factor in its success. Unfortunately, the predominant, lecture style of university education rarely generates this active participation. This dissertation describes a series of three Tablet PC-based educational technologies and their use to study and promote interaction in the university classroom. All three tools take advantage of the critical role presentation slides play in mediating learning in many modern university classrooms. The development and study of all three tools follows the design experiment methodology, a cyclical process of (1) classroom observation to understand challenges to interaction, (2) design of interventions to address these challenges, and (3) study of those interventions in real classrooms, leading to a new cycle. In particular, this dissertation describes the Classroom Presenter system for Tablet PC-based presentation, the Classroom Feedback System for student feedback in the context of lecture slides, and the Structured Interaction Presentation System for unified design of interactive and static presentation material.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

Longstanding commitments and significant contributions demand that I begin by thanking my parents, Jay and Shelley Wolfman. Thanks to you for life, guidance, support, and friendship. My brother Ian Wolfman has been as close to the ideal older brother as anyone could be. I looked up to him as a child, and I look up to him as a man. Thank you, Ian.

I would never have succeeded as I have without the support of my wife, Rachel Pottinger. Rachel has supported me emotionally, technically, and physically. Thank you, my love.

My advisor, Richard Anderson, made this dissertation possible by legitimizing and promoting computer science education and educational technology research as *computer science* research. His courageous and insightful shift in research direction made my shift from artificial intelligence to educational technology possible. His advice, collaboration, and guidance since then have been instrumental to my work.

My courage to make a similar shift in research direction, however, came from Tammy VanDeGrift's lead. She was the first of our group to strike out with Richard, and I am proud and grateful to say that I followed her lead.

The University of Washington Computer Science & Engineering Education & Educational Technology Group ("ET" for short) deserves not just thanks but credit. Richard Anderson, Ruth Anderson, Harris Chan, Kate Deibel, Crystal Hoyer, Ian Li, Vivek Nagaraja, Rod Prieto, Craig Prince, Jonathan Su, Tammy VanDeGrift, Fred Videon, Ken Yasuhara, and Shengli Zhou, among others, directly contributed to the development and research on Classroom Presenter and its extensions. Although Beth Simon is not directly affiliated with the University of Washington, she has nonetheless been a valuable ET Group member and a Presenter developer and user. I am proud to call this group my colleagues and friends.

My thesis committee — Richard Anderson, Bill Griswold, David Notkin, Dan Weld, and Zelda Zabinsky — provided critical guidance in bringing this work to fruition.

Dan Weld and Steve Hanks shepherded my early efforts at research and validated my interest in teaching. As my first-year advisor, Steve set me on a successful path. As my advisor through quals and generals, Dan shaped my approach to research and guided me to my early successes.

Throughout my years of graduate school, Mike Ernst has been an informal second advisor to me. Mike taught me not only how to be an effective researcher and author but how to be a better person.

Although Denise Salisbury also frequently assisted me with my research, her great contribution to my career has been to remind me whenever I needed it how to let work go and enjoy life.

Phil Bell and Reed Stevens taught me how to do educational research, Although the failures and weaknesses of my methods are entirely mine, the successes and strengths owe much to Phil and Reed's guidance.

The Microsoft Research Learning Sciences & Technology Group provided financial support, technical guidance, code, and valuable discussion. Portions of this work were performed while my advisor was on sabbatical with the LST Group and while I was an intern there.

Financial support for this work also came from the ARCS Foundation, Intel, and the National Science Foundation.

My work would not have been possible without the participation of hundreds of students, staff, teaching assistants, and faculty. To them belong the many innovations and insights in this work that emerged from practice, and their patience and enthusiasm carried this work along.

Frankye Jones, Lindsay Michimoto, and Steph Shull have guided me adeptly through the shoals of graduate school. I am not saying (in print) that they have bent any rules for me, but I am thanking them for it!

Patrick Allen, Shannon Gilmore, Melody Kadenko, and Alicen Smith have smoothed the way for my work. Only recently have Shannon and Mel started to teach me some of what they have been doing for me all along, and I am terrified to imagine that I might not have such able support in the future. Similarly, Scott Rose and UW CSE technical support as a whole have kept my hardware and software running.

Eric Brittain and Doug Zongker each inspired portions of my work with theirs. Doug also helped me explore the vast literature on presentation systems.

As Tammy led me to ET research, Brad Chamberlain led me to teaching. Though he stubbornly denies credit, his tremendous example as instructor for data structures inspired me to take a chance on teaching and also inspired many of my methods.

Wayne Jacobson and the rest of CIDR helped mature my teaching and provided invaluable references to pedagogical literature.

The many students and teaching assistants who participated in my classes, and the others with whom I discussed thorny pedagogical issues also made my teaching a success. Martin Dickey, my co-instructor for CSE142, deserves a special thanks for his collaboration on the course and for his tolerance and encouragement of my crazy, newbie teaching ideas.

As the ET Group supported me in my recent research, the AI Group supported me in my early research, especially Corin Anderson and Tessa Lau, good friends and colleagues.

Owen Astrachan, Alan Biermann, John Board, Carla Ellis, Susan Rodger, and Gary Ybarra of Duke's Computer Science and Electrical Engineering Departments were mentors, friends and role models to me during my undergraduate education and since.

Three teachers, in particular, from my K-12 years helped set the course of my life.

Debbie Brunson taught me to love learning. Jeff Funkhouser taught me to love teaching. Sheila Kolb showed me how to stand as my own person.

Finishing this dissertation was a monumental effort that I could not have accomplished without frequent and persistent distraction. In the final months of work, three groups provided particular support for this aspect of my “work”. The UW CSE bridge group was always ready to eat a lunch hour (or two). Neil Spring supported me with discussion of work but even more so with discussion of LaTeX and titanic grammar smackdowns. My college friends’ play by e-mail group also kept my head from going too far into the sand.

Many other mentors guided me during my research. I cannot name them all; so, I will choose just one to stand for all. Neal Lesh was a tremendous boss and friend during my internship at MERL. Neal led me in that short time to discover methods and ideas that would inform all of my future work.

Many other graduate and undergraduate students have helped with my work. Again, I cannot name them all; so, I will choose just one to stand for all. Vibha Sazawal’s feedback and ideas have been invaluable in improving my work, her friendship helped make my graduate school years a wonderful experience, and she sets an example of extraordinary dedication and lofty principles that I can only attempt to emulate.

Many staff members from UW, MSR, MERL, and elsewhere have helped with my work. You already guessed that I cannot name them all and will choose just one to stand for all. Crystal Eney has helped me many times in her official capacities with the department, but she has also done far more than her duty to improve this department. Besides helping me with my work, I have also had the pleasure to be peripherally involved in her efforts to bring women to computer science. Crystal has been a role model and a friend.

Many faculty and researchers at the UW and elsewhere have helped with my work. Continuing the trend, I will choose just one here to stand for all. Dan Grossman only arrived at the UW recently, but his reflections on research and academic life have already aided my work and my job search immeasurably. Dan embodies the best of academia: intelligent, thoughtful, and always ready to discuss any issue.

Inevitably, I have missed people and groups who contributed to my work and my life. I can only beg their apology here, claiming time pressures and sheer blockheadedness. Thank you, all of you, for making me who I am and helping to bring this work to fruition.

Finally, this dissertation is based, in part, on substantial prior work by the members of the University of Washington Computer Science & Engineering Education and Educational Technology Group [3, 4, 5, 6, 7, 8, 10, 11, 12, 36, 92, 95, 96, 119]. In particular, large portions of two papers [4, 11] and smaller segments of several others [3, 5, 8, 10, 95] are reproduced in Chapters 1, 3, and 4. This entire document should be considered to draw from these critical papers. Therefore, they are not independently cited except to reference content *not* included in this document. Other works by the Education and Educational Technology Group are cited *in situ* where appropriate.

DEDICATION

To my wife, Rachel Pottinger. For the three most important things.

To my parents, Shelley and Jay Wolfman. For borrowing and carrying.

To my advisor, Richard Anderson. For opening the door.

Chapter 1

INTRODUCTION

Our tools affect our teaching,[1] not only increasing our efficiency at teaching tasks but also opening up qualitatively different modes of interaction with our students. Indeed, what classroom would be complete without a projector or a (black-, white-, or green-) board? Who would broach denotational semantics without writing space or trombone slide positions without a trombone?

Tufte carries this argument much further and blames one particular tool, Microsoft PowerPoint, for nothing less than the Stalinization of presentations [104]. To reiterate, he blames a presentation *tool* for a broad cultural shift in presentation style. Although Tufte's claims are (perhaps intentionally) hyperbolic, PowerPoint was always intended to fundamentally change the presentation process. Before PowerPoint — or more accurately, before its predecessor, coincidentally named "Presenter" — the creator of a presentation specified content but left layout to professionals with the time and expertise to prepare slides. Bob Gaskins introduced PowerPoint to "[allow] the content-originator to control the presentation" [81].

These lofty goals and vicious slanders resonate in the classroom as well. Students laud PowerPoint's use in class for rendering abstract concepts visible and providing a tangible record of the class, and instructors praise the system for freeing them to design the visual artifacts they envision, fulfilling Gaskins's purpose. Yet, students revile "the instructor reading PowerPoint slides", and instructors chafe under PowerPoint's yoke, prodded into

[1] I assume that all readers have experienced teaching, if not in the classroom then in the board room, the dining room, the hallway, or elsewhere.

an undesired “demonstration model” of presentation [53].

PowerPoint the tool together with the PowerPoint user community give rise to a culture of presentation that sometimes succeeds and sometimes fails at improving education. PowerPoint is far from alone in this role. Wall-to-wall whiteboards provide awesome resolution and potential for parallel use — *i.e.*, a half-dozen students writing up their solutions on the board at the same time. Yet, writing on the board can be a daunting experience for a student! March to the front of the class, stand with your back to your peers and your teacher, and present your ideas for public criticism. Overhead projectors, student desks, three-ring binders, VCRs, textbooks, and other technologies all carry their own advantages and disadvantages and thus alter the space of effective teaching in their own ways.

In this space of tools, the computer is uniquely suited to the needs of the classroom. Software’s extraordinary malleability facilitates iterative design of creative technologies. Computer hardware supports a plethora of valuable communication technologies like wireless networking and pen-based input. Still, as the travails of PowerPoint show, designing successful classroom technologies is far from simple.

Our research goal, then, is to design new presentation tools for the classroom to enhance interaction among students and between students and instructor. By maintaining the advantages of existing technologies, assuming the advantages of older technologies, and forging new affordances, we aim to create classroom presentation tools that will support and encourage student interaction. There are two sides to this approach: improving presentation tools to give more flexibility in delivering the lecture, and introducing mechanisms to support interaction between student and instructor devices.

Chapter 3 addresses the former, describing the Classroom Presenter system that we developed for delivering a lecture from the Tablet PC and for displaying the instructional materials on multiple machines. The system provides an adoption path that leads from passive presentation toward more flexible and interactive presentations. Extensive use in the classroom, backed up by careful observations and frequent feedback from instructors and students, identified key strengths and weaknesses of the system, and this chapter describes this user-centered design process from design principles through design and evaluation. The distributed architecture of Presenter also makes it an entry point into interacting with student devices.

Later, Chapters 4 and 5 describe two extensions to the presentation system for supporting interaction between students and instructors. The Classroom Feedback System (CFS) enables simple student feedback, giving students a simple and unobtrusive new channel of communication with the instructor. As with Presenter in Chapter 3, Chapter 4 describes the iterative design process that led to CFS. We identified key challenges to feedback in large classes, matched these challenges to design principles for the feedback system, and then refined and evaluated this design in classes. A key contribution of CFS is the identification of successful patterns of interaction for computer-mediated communication, including one that would be infeasible without computer support (Section 4.4.4).

Chapter 5 describes the Structured Interaction Presentation System (SIP), which supports the design and use of rich, interactive exercises embedded into presentation slides. SIP responds to problems instructors had handling real-time, student-initiated feedback in CFS by supporting carefully pre-planned interactive exercises. SIP also allows instructors the flexibility to tailor their exercises closely to the circumstances of their classes. Unlike Presenter and CFS, SIP is a fledgling system that has only been used in one experimental class. Chapter 5 describes the SIP system, the design principles at its core, and this experimental class. Appendix A presents several example SIP presentations that demonstrate SIP's potential to create effective interactive exercises, including the entire presentation from the experimental class with data from that class.

Before diving into the heart of the work, Chapter 2 introduces two key principles that drove the design of the systems described in this dissertation. After the three core chapters (Chapters 3–5), the remaining chapters present related work (Chapter 6) and conclusions (Chapter 7).

Chapter 2

PRESENTER DESIGN PRINCIPLES

The work in this dissertation follows Ann Brown's "design experiment" methodology [24]. A design experiment engineers a learning environment by introducing an intervention. First, the researcher studies the environment to guide design of the intervention; then, she iteratively engineers the intervention and studies each iteration's impact on the environment. As Brown puts it, the design scientist "attempt[s] to engineer innovative educational environments and simultaneously conduct experimental studies of those innovations." The key contributions of the design experiment are the designed intervention and the principles that informed its design.

In each of Chapters 3–5, I will describe both a designed system and the design principles underpinning that system. However, two central design themes arose early in the research process, resonated across all three systems, and drove their design. Section 2.1 discusses the theme of exploiting the power of projected slides as a mediating artifact in the classroom. Section 2.2 discusses the theme of enabling end users' innovations.

2.1 Slides as Mediating Artifact

In his keynote address at CSCL 2003, Roger Säljö described the power of Vygotsky's "mediating artifacts" to change the nature of learning. Mediating artifacts are objects, practices, or ideas that participate in and shape human cognitive processes. As one example, Säljö described how the process of mathematical calculation has changed with time. Before writing, even the simplest calculations would strain human memory. Written text mediated the process of calculation, offloading memory effort and dramatically enhancing human prowess at calculation. The modern calculator shifted more cognitive effort to the artifact. Now, we offload not only memory but algorithmic knowledge of computation as well [90].

Artifacts can also mediate communication among people. Merchants of Hong Kong's famous jade market use small LCD calculators to negotiate prices with their polyglot clientele. A seller punches in a price, displays it to the buyer, and then perhaps hits "$*.9$" to suggest a 10% discount. The buyer punches in her counteroffer. The artifact mediates communication between the two, making it more efficient, changing its character, and even allowing exchanges that would otherwise be impossible.

Prepared slides — of the form created in applications like SliTEX, Microsoft's PowerPoint, and Apple's Keynote — mediate communication in the classroom. Instructors design instruction using slides. Slides remind of, and sometimes enforce, the order of presentation and content of ideas during class. Students and instructor both refer to content on the slides, *e.g.*, "that figure" or "the second bullet". Students embed their notes in slides (and some evidence suggests this is better for learning than using a blank page [1]). Instructors even reuse slides across terms or share them with other instructors, passing knowledge to themselves and others by embedding it into slides.

Throughout this dissertation, the notion of slides as a powerful, public, and shared mediating artifact is central. Changing this core artifact can have significant impact on classroom practice, from instructional design through classroom teaching and into archival reuse. Imbuing this artifact with certain pedagogical choices makes those choices easier, more likely to occur, and more effective. Altering the mediation of communication and learning can elevate simple interventions into powerful shifts in classroom potential.

2.2 Supporting End User Innovation

The most interesting and exciting innovations in computer-mediated communication systems arise from users' natural patterns of practice. Examples abound in our own and related work. In Classroom Presenter, instructors invented features like "whiteout ink" —

erasing mistakes on prepared slides by drawing over them with the slide background color (Section 3.1) — and practices like the "TV talk show host" style — passing the Tablet PC around like a talk show host's microphone to mediate class discussion and collect student responses (Section 3.4.1, page 61). In the Classroom Feedback System, students and instructors showed that leaving feedback on the slides ahead of discussion could be a successful pattern of interaction (Section 4.4.4). Although the Structured Interaction Presentation System has not yet seen real classroom use, its system of exercises built from simple, interconnected interaction primitives already proved flexible enough to realize the presentation design of an author besides its creator (Appendix Section A.2, Appendix Section A.3, and Section 5.3.3). ActiveClass's designers introduced its question answering feature after students co-opted the question-*asking* feature to answer questions [85]. Therefore, a key design goal throughout the work presented in this dissertation is to nurture these user innovations: by attending to users' practices, encouraging users to "co-opt" these systems, responding to users' suggestions, and building flexibility into systems to support unexpected uses.

Chapter 3

CLASSROOM PRESENTER

Classroom Presenter — hereafter, just Presenter — is a presentation system that supports digital ink over prepared slides. In the most common deployment scenario, an instructor prepares slides using standard slide design software such as PowerPoint and exports these to Presenter's slide format (Figure 3.1). The instructor then uses Presenter, running on a Tablet PC pen-based computer, to annotate on and navigate through the slides in class. A second computer, which need not be a Tablet PC, drives projection of the slides and annotations to a public display. The two computers communicate and synchronize via a wireless network connection. (Figure 3.2 summarizes the flow of information among clients.)

The Presenter System itself is a contribution to the improvement of classroom education. Furthermore, the insights and experiences generated by its design process contribute to the body of knowledge about adoption of technology in the classroom, the problems and strengths of pen-based technology, and the key characteristics of successful presentation technology.

The organization of this chapter primarily follows the design process that led to Presenter; however, the iterative nature of the design process makes a linear narrative — from design through implementation to evaluation — a challenging task. Therefore, Section 3.1 begins by giving a brief introduction to Presenter's interface. Using this grounding, Section 3.2 lays out the key goals and decisions that informed Presenter's design, along with the experiences that motivated, validated, and in some cases contradicted those choices. Then, Section 3.3 returns to system description, describing Presenter in detail. Next, Section 3.4 addresses Presenter's use in classes, briefly presenting statistics demonstrating Presenter's aggregate impact and then relating in detail patterns of system use that reveal interesting

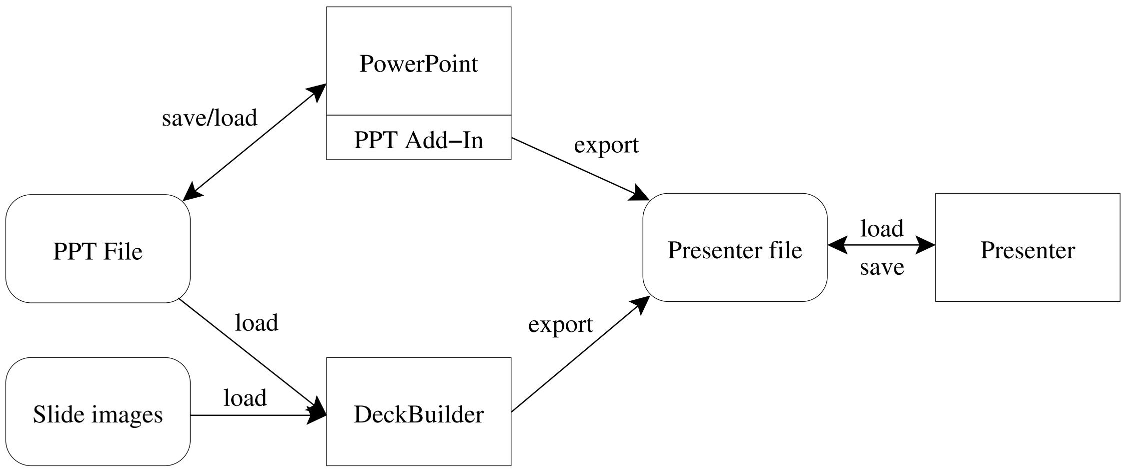


Figure 3.1: Block diagram of creation of Presenter files. The author starts by creating her presentation either as a PowerPoint file or in another slideware application, exporting the slides as images. If she is using Presenter's PowerPoint add-in, she exports the slides directly from PowerPoint to Presenter's file format. With the stand-alone DeckBuilder application, she loads the presentation source and then exports it to Presenter's file format. (The add-in and DeckBuilder are both described in Section 3.3.3.)

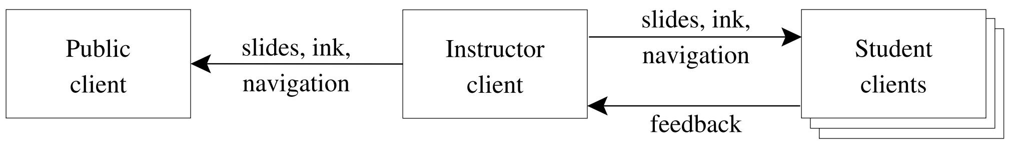


Figure 3.2: Block diagram of data flow among Presenter clients during execution of a presentation. The instructor's client sends slides, ink annotations, and navigation commands to the public display. (Distance courses might also have multiple public clients.) The Classroom Feedback System (Chapter 4) also includes student clients, which can send feedback to the instructor client. The Structured Interaction Presentation System (Chapter 5) adds communication with a database, not shown here. (These arrows represent logical flow of data; data is actually sent using multicast (Section 3.2.4).)

and valuable design lessons. Finally, Section 3.5 explains how decisions in Presenter's design process supported or hindered its adoption in the classroom.

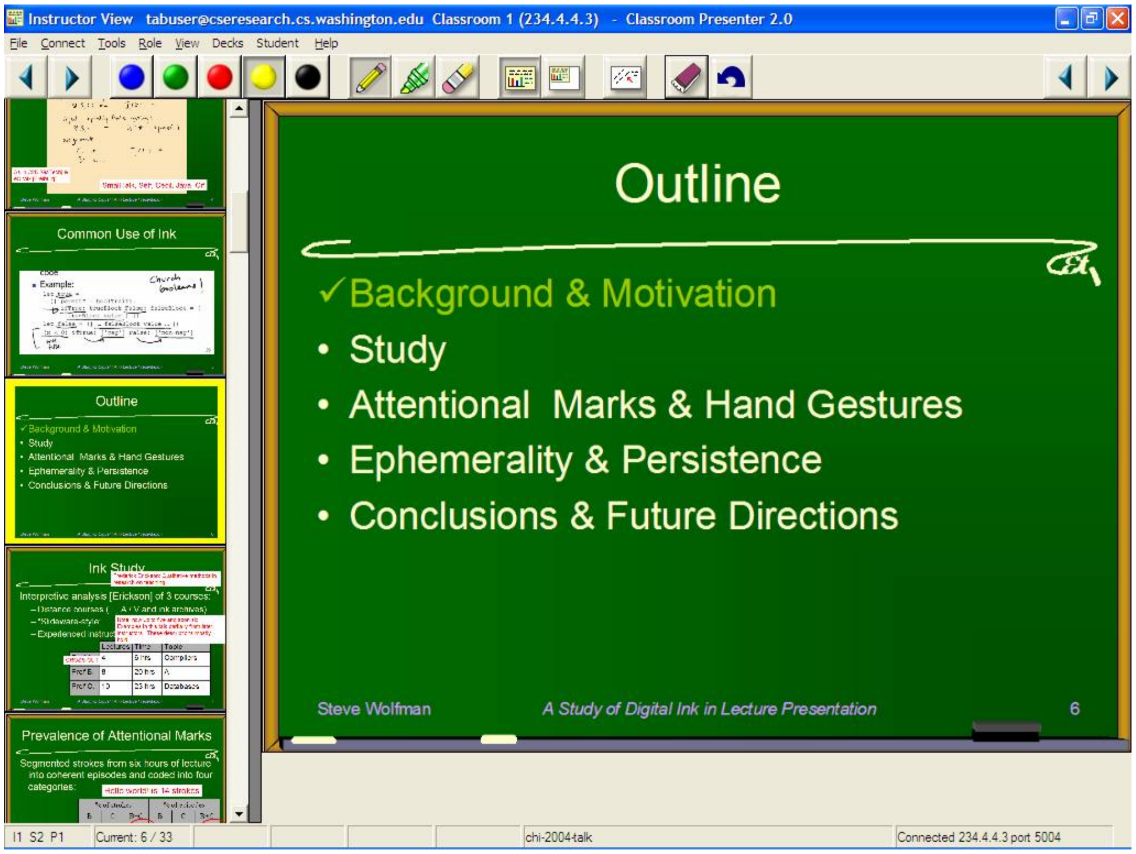
3.1 Presenter Interface

This section provides a brief description of Presenter's primary interface features, the portion of Presenter that is most important to its users. In particular, the focus is on Presenter's instructor interface and public display. Chapter 4 on the Classroom Feedback System and Chapter 5 on the Structured Interaction Presentation System discuss two extensions supporting student participation. However, the extended support for student involvement in the current version of Presenter [96] is outside the scope of this dissertation. The basic Presenter use scenario presented at the start of this chapter will suffice to explain Presenter's interface features. The full list of use scenarios is described in Section 3.3.2.

Figure 3.3 shows the instructor's view of Presenter. There are five primary elements of the instructor's display: the current slide, the "filmstrip" along the left, the large row of control buttons near the top, the small row of menu options above the control buttons, and the informational displays along the bottom.

The current slide takes as much screen space as possible on the right side of the screen. Increasing the size of the current slide improves legibility and increases the effective resolution of ink capture (digitizing) across the slide surface. Insufficient digitizing resolution causes "jaggies", blocky artifacts, in ink rendered on higher resolution displays such as the public projection display. Indeed, a primary motivation for our choice of the Tablet PC platform was its high resolution ink capture compared to, *e.g.*, handheld computers.

Besides differences in resolution, the current slide display usually shows nearly the same view as the public display. Presenter allows some intentional differences. For example, the instructor can include private notes (as ink, text, geometric shapes, or other standard presentation objects) on her display, and the Classroom Feedback System described in Chapter 4 overlays various private displays of student feedback on the instructor's view.

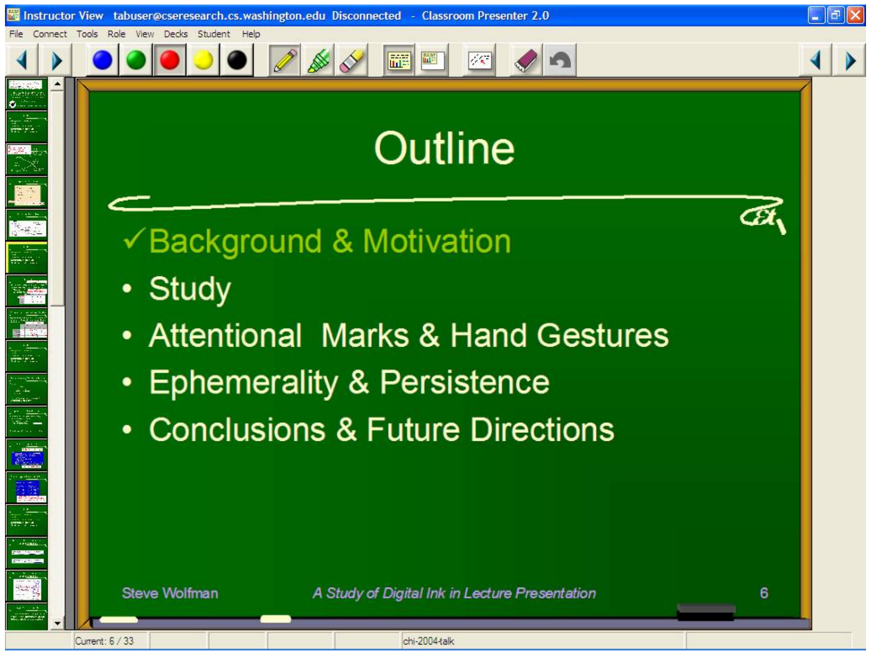

Instructor View tabuser@cseresearch.cs.washington.edu Classroom 1 (234.4.4.3) - Classroom Presenter 2.0
File Connect Tools Role View Decks Student Help
Common Use of Ink
Outline
Ink Study
Prevalence of Attentional Marks
I1 S2 P1
Current: 6 / 33
chi-2004-talk
Connected 234.4.4.3 port 5004

Outline

✓ Background & Motivation
- Study
- Attentional Marks & Hand Gestures
- Ephemerality & Persistence
- Conclusions & Future Directions

Steve Wolfman *A Study of Digital Ink in Lecture Presentation* 6

Figure 3.3: The instructor's view of a Presenter presentation. The large display to the right shows the current slide. Menus and the toolbar span the top of the window. The title bar at the top and status bar at the bottom provide information about the presentation's status. The filmstrip is the scrollable series of slide thumbnail images along the left.


Instructor View tabuser@cseresearch.cs.washington.edu Disconnected - Classroom Presenter 2.0
File Connect Tools Role View Decks Student Help
Current: 6 / 33
chi-2004-talk

Outline

✓Background & Motivation
- Study
- Attentional Marks & Hand Gestures
- Ephemerality & Persistence
- Conclusions & Future Directions

Steve Wolfman *A Study of Digital Ink in Lecture Presentation* 6

Figure 3.4: The instructor's view with the filmstrip resized to be much smaller than in Figure 3.3.

Two other differences are compromises. First, for performance reasons, Presenter bundles multiple ink samples together for network transmission (see Section 3.2.3), causing some "jerkiness" in the flow of ink on the public display. Second, the instructor's cursor — which shows the size, color, and shape of the current pen — is configurable to be shadowed by a cursor on the public display.

The filmstrip is the vertical sequence of slide thumbnails along the left of Figure 3.3. Figure 3.4 shows the filmstrip in use after being resized from Figure 3.3. The filmstrip can also be moved to the bottom or right-hand side of the display, to accommodate different slide aspect ratios or instructor handedness.

The filmstrip plays two roles in Presenter: as an information source that provides instructors with a gestalt of the presentation's flow and as a control allowing rapid navigation throughout the presentation. The filmstrip shows a scrollable window on all the slides in the

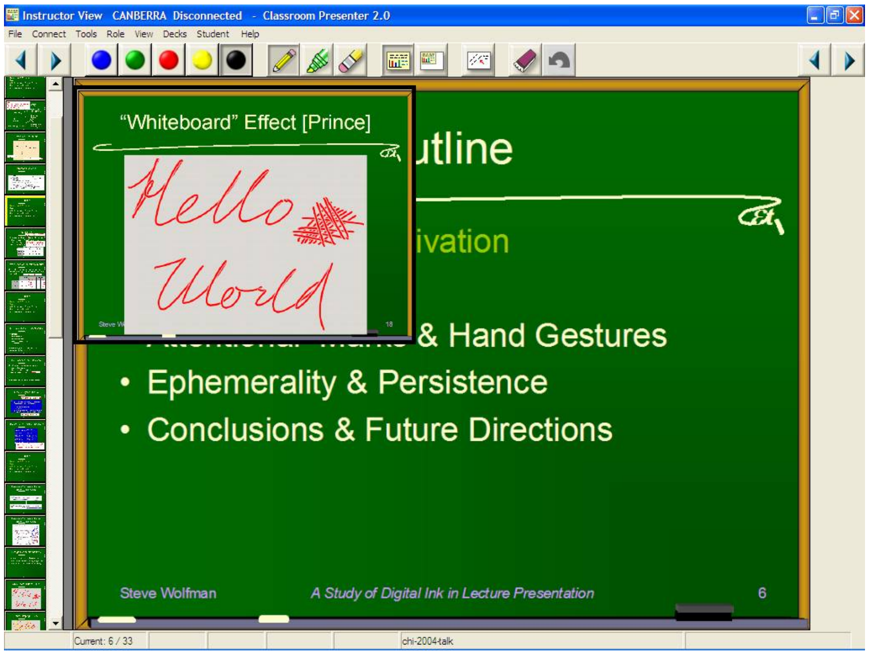

Instructor View CANBERRA Disconnected - Classroom Presenter 2.0
File Connect Tools Role View Decks Student Help
"Whiteboard" Effect [Prince]
Hello World
utline
ivation
& Hand Gestures
• Ephemerality & Persistence
• Conclusions & Future Directions
Steve Wolfman
A Study of Digital Ink in Lecture Presentation
6
Current: 6 / 33
chi-2004-talk

Figure 3.5: The instructor's view of Presenter with a slide preview. The instructor clicked and dragged on the filmstrip, activating the preview overlaid on the current slide's upper left corner. The previewed slide changes as the pen drags into new filmstrip cells. As the pen approaches the top or bottom of the filmstrip, the strip scrolls to expose hidden slides. Lifting the pen kills the preview.

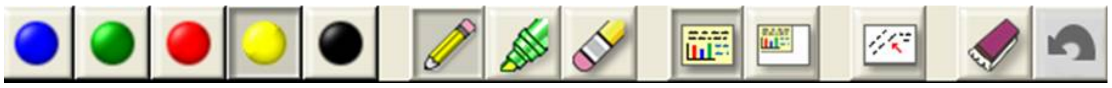
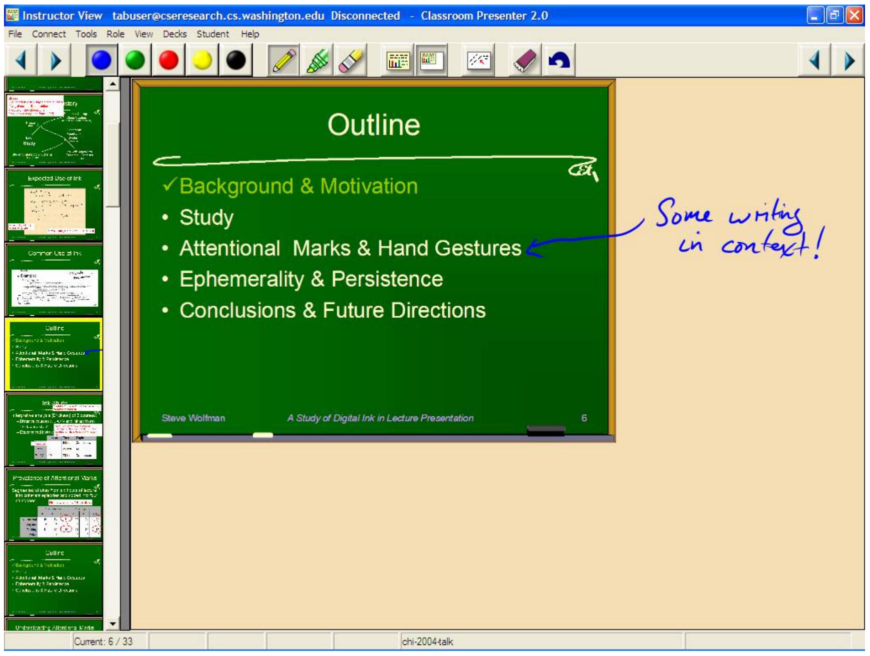


Figure 3.6: Detail of the most important buttons from the toolbar across the top of the instructor's view in Presenter. (See Figure 3.3 for the whole view.)

presentation arrayed in order from top to bottom. The current slide is highlighted in yellow and centered (except when scrolling reaches the start or end of the presentation). Instructors can change the size of the filmstrip using the oversized gray divider bar between the filmstrip and the current slide display. Instructors can get more information and navigate the filmstrip by clicking and dragging. As soon as the instructor starts dragging, the slide under the pen pops up in a larger preview as in Figure 3.5. When the pen approaches the top (bottom) of the filmstrip window, the filmstrip scrolls increasingly rapidly toward the start (end) of the slide deck. Instructors can use the filmstrip to navigate simply by clicking a slide. That slide immediately becomes the current slide and is centered in the filmstrip window. Thus, an instructor can quickly scan for a slide using the click-and-drag gesture. When the slide the instructor seeks shows up in the preview window, she can lift the pen and tap immediately to transition to that slide. Alternatively, instructors can advance (backup) slides rapidly by clicking repeatedly just below (above) the centered current slide, repeatedly bringing a new current slide to the center of the display. Presenter and the filmstrip view also support multiple decks. When multiple decks are loaded, the filmstrip includes tabs for navigating among decks.

Different instructors use the filmstrip in different ways, but two uses are common. First, resizing the filmstrip as large as possible (without seriously degrading the current slide display's resolution) makes it easy for an instructor to glance down and preview the next slide. Second, making the filmstrip as small as possible exposes more of the presentation's slides. The instructor can then use slide previews, by clicking-and-dragging, to quickly scan and navigate the tiny filmstrip thumbnails.


Instructor View tabuser@cseresearch.cs.washington.edu Disconnected - Classroom Presenter 2.0
File Connect Tools Role View Decks Student Help
Current: 6 / 33
chi-2004-talk

Outline

✓Background & Motivation
- Study
- Attentional Marks & Hand Gestures
- Ephemerality & Persistence
- Conclusions & Future Directions

Steve Wolfman A Study of Digital Ink in Lecture Presentation 6

Some writing in context!

Figure 3.7: The instructor’s view of Presenter with the slide “minimized”: scaled down into the upper-left corner to introduce extra annotation space that is still in the context of the slide.

Figure 3.6 shows the most important control buttons, also visible across the top of Figure 3.3. The buttons are in five groups. Group one controls pen color, offering five colors. Group two controls the pen mode: regular ink, highlighter (transparent ink with a large pen tip), and erase mode. Regular ink uses a round tipped pen while the highlighter uses a tall, thin, rectangular tip. The currently selected color in group one is set independently for regular ink and highlighter mode. Regular ink can be blue, green, red, yellow, or black. Highlighter colors are similar pastel shades. Erasures are stroke-based. So, an entire ink stroke — *i.e.*, the ink created by one continuous contact between pen and screen — is erased when touched by the pen-controlled cursor in erase mode. Group three manipulates the slide size. The right button introduces extra annotation space by scaling the current slide down into the upper-left corner of the slide display area, as shown in Figure 3.7. The left button returns to the normal slide display. The single button in group four navigates to and from an auxiliary, extensible deck of blank slides used as white boards. The final group of buttons includes page erase on the left, which erases all the ink on the slide, and undo on the right which undoes any inking but is all too often used to undo accidental slide erases.[1]

Two groups of control buttons were left off of Figure 3.6, the left-handed and right-handed slide navigation buttons at the extreme left and right of the control bar, respectively, in Figure 3.3. The two groups are identical, with a left arrow button to backup the current slide and right arrow button to advance it for step-by-step slide navigation. Instructors often use the filmstrip to accomplish the same task.

The small menubar along the top of the screen in Figure 3.3 accesses the tools in the control bar below along with less commonly used tools including startup and shutdown operations, configuration options, disused older features, and “bleeding edge” tools that have not yet migrated to the main control bar. Detailed description of these tools is outside the scope of this dissertation; however, we describe several of the most important or interesting

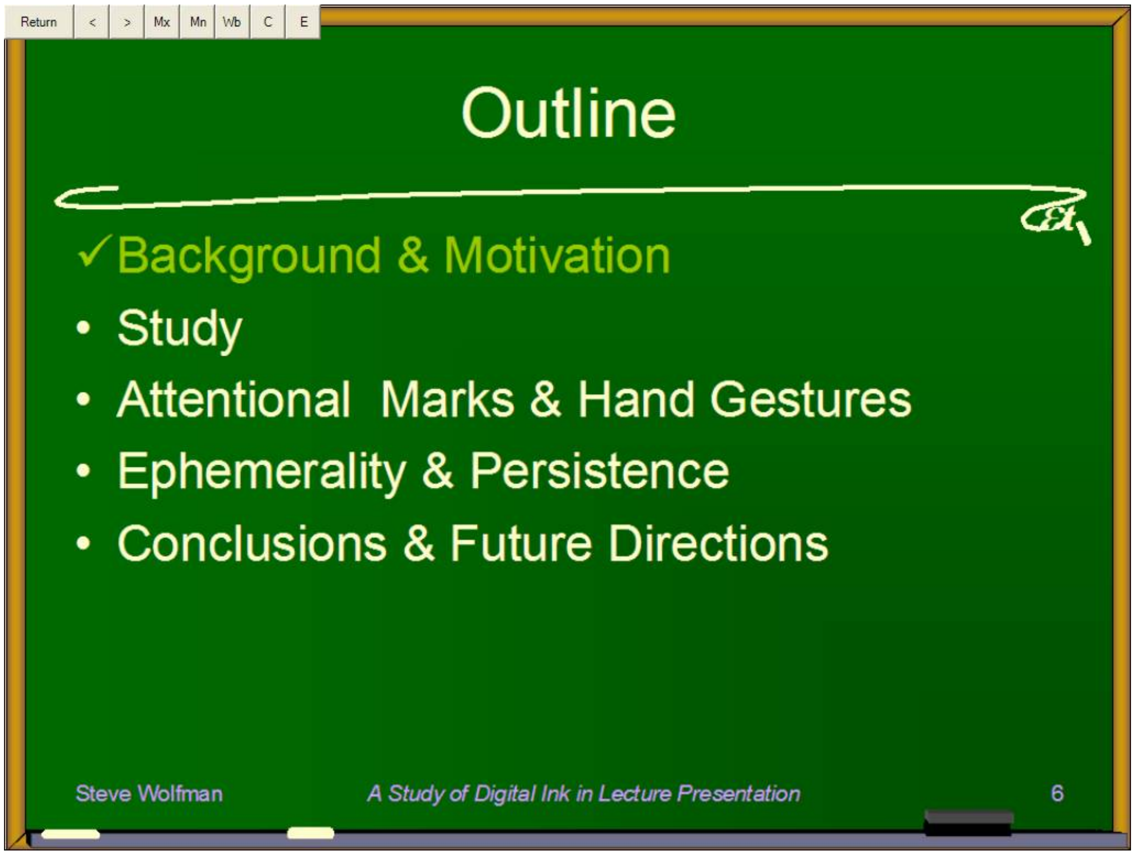
[1]Of course, reversibility of actions is a fundamental user interface tenet [32]. Nonetheless, the undo feature only gained momentum after a child with friends in high places accidentally deleted hours of sketching for a homework assignment.

here. Section 3.3.1 also discusses some options from the `Connect` menu.

The `File` menu includes standard options for opening and saving Presenter files. Saved files include all unerased ink. This menu also includes options to export ink as images or in the Tablet PC API's native ink serialization format.

The `Tools` menu exposes inking controls and general customization options (through properties forms). The options that are most important during presentations are replicated as buttons on the control bar below, *e.g.*, `Undo` and color selection. Of the remaining options, `Whiteout`, the `Tip` sub-menu, and the `Remote Pointer` are of particular interest. `Whiteout` is a new tool for "erasing" slide content to the background slide color that has not yet moved to the main control bar. The whiteout tool samples a slide for the most common color and provides a large, opaque brush in that color. An instructor using Presenter inspired the whiteout tool when he "erased" a mistake from his blue-background slide using the blue pen. With access to the presentation source, the whiteout tool could also acquire a blank "reference" slide and erase to that slide (allowing erasures from a slide with a patterned or gradient background). Eventually, whiteout will be an extra color option on the main control bar. In contrast, the `Tip` sub-menu is an older feature that was moved from the main control bar to the menus because of disuse. The `Tip` sub-menu allows instructors to change the pen tip's shape from round to square. Finally, the `Remote Pointer` option toggles the visibility of a cursor on the remote screen that follows the instructor's pen tip whenever the pen is in range of the digitizer and within the current slide display's bounds. The remote pointer offers an ephemeral alternative to ink for deictic (pointing) gestures. The detailed customizations available in the properties menu are not discussed in this dissertation.

The `Role` menu allows switching among Presenter roles: `Instructor`, `Public` (projected display), and `Student`. Switching roles changes the view presented and the client's role on the network. For example, switching from instructor to public roles removes all the instructor's control and informational displays, maximizes the current slide's size, and ceases advertising the client as an instructor to other clients on the network.



Outline

✓ Background & Motivation
- Study
- Attentional Marks & Hand Gestures
- Ephemerality & Persistence
- Conclusions & Future Directions

Steve Wolfman A Study of Digital Ink in Lecture Presentation 6

Figure 3.8: The instructor's view of Presenter in full-screen mode. The current slide fills the screen except a small control bar in the upper left devoted to frequently used controls and a button to exit full-screen mode. This mode is appropriate for direct projection of the instructor's screen.

The `View` menu customizes the set of visible controls and the set of “layers” (see Section 3.2.4) visible over the current slide display. It also controls switching to full-screen mode, shown in Figure 3.8. As discussed in Section 3.3.2, instructors can use the full-screen display for a simple, single-computer instruction scenario. The public display automatically uses full-screen mode. (In the public role, no return button or other controls are visible, but any pen-tap/mouse-click or the Escape key returns from full-screen mode.)

The last two menus, `Decks` and `Help`, are not discussed in this dissertation.

Finally, the informational displays across the bottom of Figure 3.3 keep clients (instructors and students) apprised of the current state of the presentation: network connection status, number of clients of each role connected, current slide number, *etc.*

3.2 Design Decisions

The Presenter research project is structured as a design experiment [24] — as described in Chapter 1 — with the goal of increasing the interactivity of classroom presentations, especially in large and distance learning classes. Therefore, a primary contribution of the research is the combination of design decisions that led to Presenter’s current implementation and evidence that supports (or contradicts) these design decisions.

Our design for Presenter was informed by studies undertaken prior to and during development of the application. We interviewed instructors and students and conducted field observations. Three key design goals and decisions came out of this process: (1) support for natural and realistic writing in the context of slides, (2) separation of views of the slides and user interface for different user roles (*e.g.*, student, instructor, or public display), and (3) a distributed application design that runs across multiple computers. This section describes these three key themes and then lays out a set of lesser design goals and decisions that nonetheless significantly affected Presenter’s form and the form of its two extensions, described in Chapters 4 and 5.

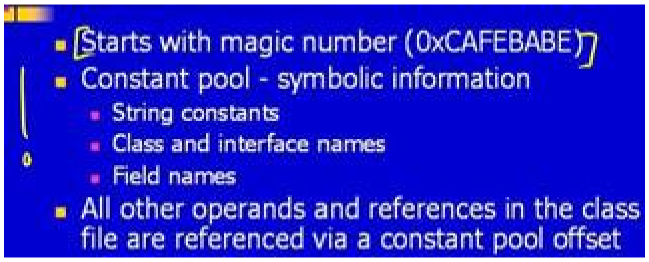
3.2.1 Realistic Ink in the Context of Slides

A key part of the Presenter System is the integration of realistic ink directly in the context of class slides. The Tablet PC, a pen-based computer supporting high-quality, realistic inking, was critical to achieving this goal. Besides using the Tablet PC's ink support, Presenter also synchronizes display of ink on all computers, so that (1) remote displays show ink shortly after the instructor writes it (usually without perceptible delay), and (2) ink "flows" (appears progressively) on the remote display, lending inking a "natural" feel and communicating the direction and speed of writing.

Prior work with electronic whiteboards [6] and early work on Presenter with pen-based computers demonstrated the need for high-quality, realistic ink in presentation. Instructors using the whiteboard were often frustrated by ink marks created unintentionally by their fingers when pointing to the board [6]. Low-resolution ink capture on the pen-based computers used with early versions of Presenter rendered writing illegible and forced instructors to write in large strokes that quickly consumed the available space on the slide.

Instructors and students using the electronic whiteboard also complained about the physical separation between writing space and prepared slides. Writing on the whiteboard often related to context on the slide; however, written notes were not anchored in the slide context to which they referred [6]. Analysis of instructors' use of ink in later versions of Presenter reinforced the importance of slide context, showing that the majority of ink comprised simple attention-directing marks with little meaning outside their slide context [12]. Figure 3.9 shows one such mark (an exclamation point) that conveys importance with its shape and indicates which points are important and the relationship among the points by its placement in the slide context.

Presenter's design responds to the need for high-quality, contextual ink by integrating writing directly over the slides. The Tablet PC supports high-resolution ink capture, smoothing of ink strokes, and varying-width strokes based on pen pressure. Presenter's multicast networking rapidly and scalably communicates incremental stroke information to



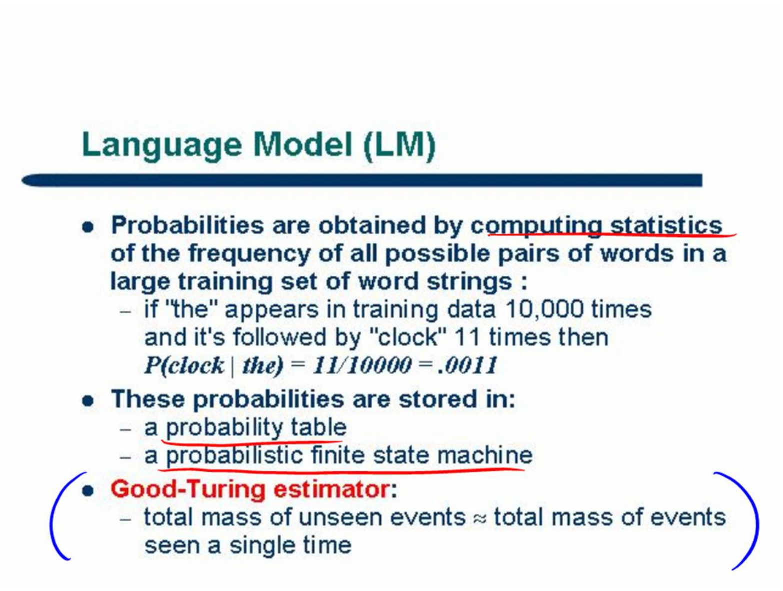
- Starts with magic number (0xCAFEBABE)
- Constant pool - symbolic information
  - String constants
  - Class and interface names
  - Field names
- All other operands and references in the class file are referenced via a constant pool offset

Figure 3.9: A slide from a course on programming languages showing the use of ink to direct attention to points on the slide. The attention-directing mark, an exclamation point, shows the importance of one bullet and groups it with its sub-bullets.

remote computers. Presenter stores ink data for each slide independently (*i.e.*, each slide is a separate "canvas") to unify ink and slide navigation operations. Ink on Presenter slides also persists from one visit to the slide to the next, making ink a reliable medium for important annotations. (Indeed, some Presenter project members now use Presenter exclusively to annotate slides during other members' practice presentations.) Finally, Presenter supports scaling a slide and its ink, creating extra writing area around the slide to support copious and complex annotations.

Putting ink in the context of slides also creates design challenges. Management of annotation space is a critical issue since writing can obscure slide content. Addition of controls for moving and resizing ink is one possible solution. Indeed, the "student submissions" feature of Presenter (not described in this dissertation) provides these controls on student computers to resolve collisions between private student ink and public instructor ink [96]. However, the introduction of such controls would add complexity and mode changes to the instructors' interface when instructors already have difficulty effectively managing other simple mode changes such as color changes, despite a simple, one-click interface.

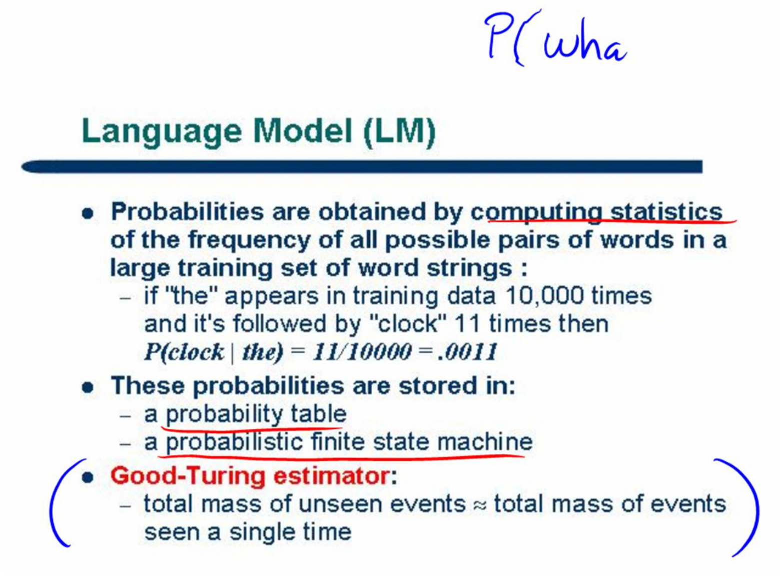
Presenter includes a coarse tool for manipulating ink location that also proved problematic: a virtual "mylar" layer controlled by a scrollbar on the instructor's display. Moving the scrollbar reels an infinite, transparent inking layer up and down across the slide, like



Language Model (LM)

- Probabilities are obtained by computing statistics of the frequency of all possible pairs of words in a large training set of word strings :
  - if "the" appears in training data 10,000 times and it's followed by "clock" 11 times then *P(clock | the) = 11/10000 = .0011*
- These probabilities are stored in:
  - a probability table
  - a probabilistic finite state machine
- Good-Turing estimator:
  - total mass of unseen events ≈ total mass of events seen a single time

Figure 3.10: The start of a sequence of screenshots (ending with Figure 3.14) from an artificial intelligence course. The sequence shows an accidental scroll of ink and a failed correction of the scroll. Note in this figure that the inked underlines and brackets line up with text on the slide.

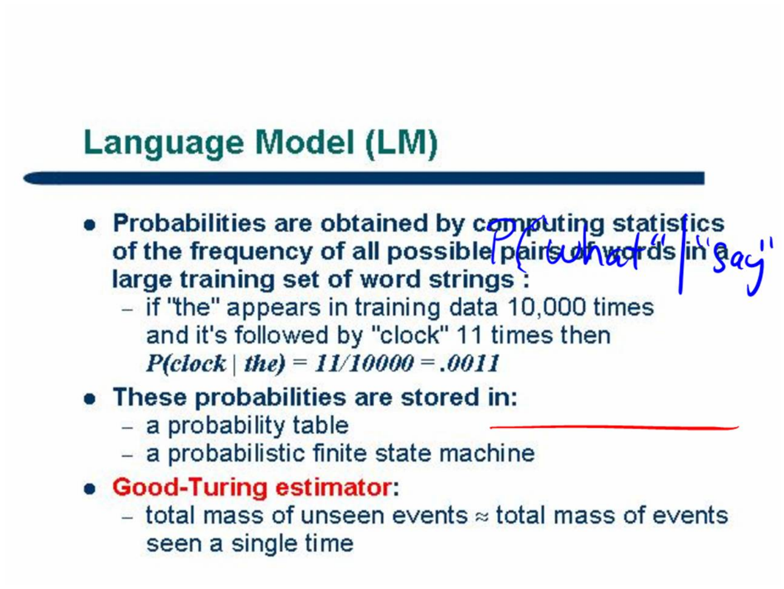


P(wha

Language Model (LM)

- Probabilities are obtained by computing statistics of the frequency of all possible pairs of words in a large training set of word strings :
  - if "the" appears in training data 10,000 times and it's followed by "clock" 11 times then *P(clock | the) = 11/10000 = .0011*
- These probabilities are stored in:
  - a probability table
  - a probabilistic finite state machine
- Good-Turing estimator:
  - total mass of unseen events ≈ total mass of events seen a single time

Figure 3.11: The instructor begins writing text in the open space at the top of the slide.

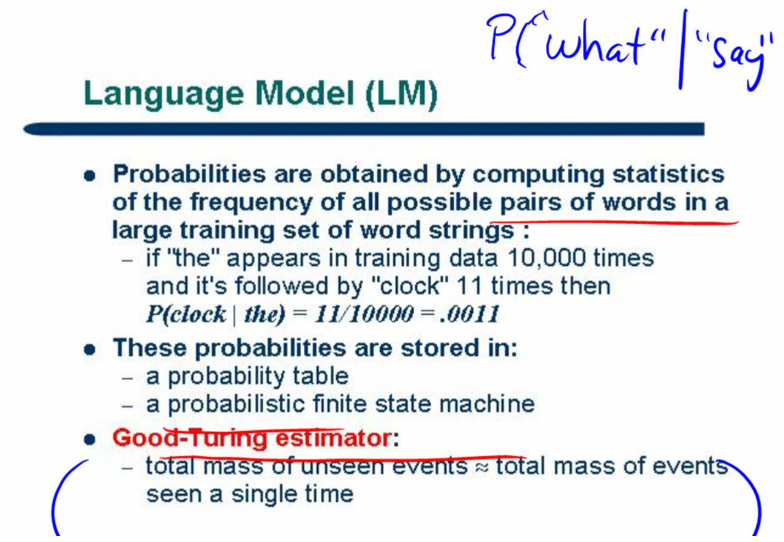


Language Model (LM)

- Probabilities are obtained by computing statistics of the frequency of all possible pairs of words in a large training set of word strings :
  - if "the" appears in training data 10,000 times and it's followed by "clock" 11 times then *P(clock | the) = 11/10000 = .0011*
- These probabilities are stored in:
  - a probability table
  - a probabilistic finite state machine
- Good-Turing estimator:
  - total mass of unseen events ≈ total mass of events seen a single time

P("what" | "say")

Figure 3.12: The text ran off the writing area toward the scrollbar control, causing a scroll when the instructor accidentally clicked the scrollbar above its thumb. (For reference, Figure 3.17 shows the instructor's view with the scrollbar visible to the right of the current slide.)

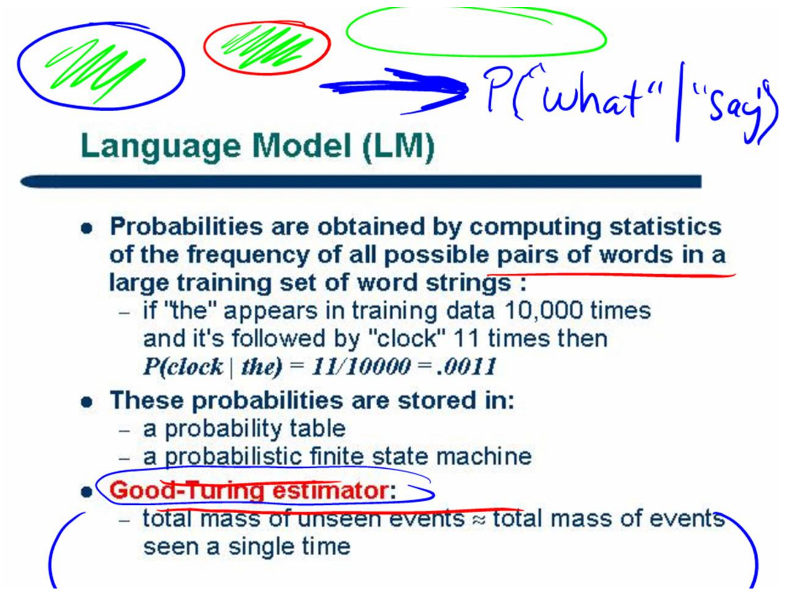


P("what" | "say")

Language Model (LM)

- Probabilities are obtained by computing statistics of the frequency of all possible pairs of words in a large training set of word strings :
  - if "the" appears in training data 10,000 times and it's followed by "clock" 11 times then *P(clock | the) = 11/10000 = .0011*
- These probabilities are stored in:
  - a probability table
  - a probabilistic finite state machine
- Good-Turing estimator:
  - total mass of unseen events ≈ total mass of events seen a single time

Figure 3.13: The instructor partially corrected the scrollbar's position by dragging its thumb. The unanchored text is suitably repositioned in open space. However, the underlines and brackets are out of place; some even unintentionally match new slide text.



Language Model (LM)

- Probabilities are obtained by computing statistics of the frequency of all possible pairs of words in a large training set of word strings :
  - if "the" appears in training data 10,000 times and it's followed by "clock" 11 times then *P(clock | the) = 11/10000 = .0011*
- These probabilities are stored in:
  - a probability table
  - a probabilistic finite state machine
- Good-Turing estimator:
  - total mass of unseen events ≈ total mass of events seen a single time

Figure 3.14: The instructor added more ink near the top that is anchored only to existing unanchored ink (the heavy arrow) as well as new unanchored ink (the three circles). Near the bottom, the instructor has circled slide text, anchoring ink in this new scroll position.

the mylar rolls on an overhead projector. Unfortunately, while some ink can be sensibly scrolled to a new position, other ink (like the attention-directing marks mentioned above) is only meaningful in its slide context, and the scrollbar cannot distinguish these uses of ink. Indeed, few instructors ever made productive use of the scrollbar feature, but several discovered it accidentally, to their dismay.

Figures 3.10–3.14 show the typical, messy result of an accidental scroll, illustrating both problems with the mylar scrollbar and the broader need for management of writing space. The instructor drew attention-directing marks (red underlines) anchored to locations on the slide and some text (at the top) that was unanchored. When he accidentally scrolled and tried to correct the scrollbar's placement, the attention-directing marks were left askew. The unanchored text, though no longer at its original location, is placed "well enough" as it is back within the open area at the top of the slide. The instructor then drew a diagram — the series of three circles in Figure 3.14 — in the cramped open space remaining. These circles appear connected to the text on the right by the heavy arrow in the middle, yet the connection is wholly unintentional. The arrow, when created, was meant only to draw attention to the text. With improved space management, the instructor would be able to express and retain intended connections to the slide — the red underlines — and avoid expressing unintended connections.[2] Creative solutions to this space management problem, such as zoomable user interfaces [51] or intelligent categorization and layout of ink might resolve the problem.

Digital capture and control of ink also offers potential that we have just begun to explore in Presenter [84]. The Tablet PC already smooths and improves writing automatically. Presenter might also manipulate ink colors to improve contrast with the background, a goal of which instructors are surprisingly oblivious [12]. Altered ink renderings — *e.g.*, erasing a narrow strip on each side of an ink stroke to show order of strokes where they overlap — might expose visually useful information that is currently only captured in the internal ink representation. Generalizing, Presenter's digital ink capture offers the opportunity to acquire copious data (such as stroke timing) and enhance ink rendering to better present that data to human users.

3.2.2 Separation of Views

The next key design decision in Presenter was separating the views different user roles have of Presenter's user interface. Like many design decisions in Presenter, separation of views was motivated by classroom use. Initially, Presenter was a distributed application with the model that all clients maintained the same information and display. We first noticed in a distance learning class that the projected version of the slide wasted screen real estate on unused controls. In response to this problem, we altered the public display to show the slide across the full screen — as PowerPoint and other presentation systems generally do. This was a simple first step toward separated displays.

[2]The instructor could have shrunk the slide to create extra annotation space, but he never discovered that feature.

A comment during an interview of another instructor drove home the great potential of separated views. We suggested to him that we could make his instructor view full-screen as well: removing the Windows taskbar and devoting that extra space to slide and control display. To our surprise, the instructor objected to the change. He wanted to keep the taskbar for its clock display! He pointed out that with Presenter he managed classroom time more effectively because the clock was continuously peripherally available.

In exchanges like this one, instructors consistently contributed the most critical embellishments of the instructor's view in Presenter. In another instance, an instructor complained that he frequently "talked over" his next slide: continuing at length on a topic while projecting one slide only to discover that he had unintentionally covered material from upcoming slides. He proposed a set of thumbnail slide images for a gestalt overview of a slide's neighborhood in the talk. That proposal developed into the filmstrip shown in Figure 3.3. We made the filmstrip resizable to accommodate the instructor's myopia.[3] Another instructor, who resized the filmstrip small enough to quickly navigate the entire presentation, was frustrated that she had to choose between legibility of thumbnails and scope of the filmstrip view. She requested a facility for scanning and previewing slides in the filmstrip which became the slide preview illustrated in Figure 3.5.

These innovations have clear value to instructors, yet all of them would be distractions and, worse, wastes of space on the public display. These differing and often conflicting constraints on different users' displays suggest an architecture that frees distributed clients to generate independent, customized views of the common underlying data. To effectively facilitate communication and interaction, this architecture must still recognize the critical role of a shared, mediating artifact (as described in Chapter 2), but the artifact's display and context need not be identical across all participants and computers. This is a simple idea, yet one that has not penetrated the dominant commercial presentation tools or even many otherwise innovative presentation systems (*e.g.*, CounterPoint[51] and Slithy [120]).

[3]Eliding the (already visible) current slide from the filmstrip to make more room for succeeding slides might support this instructor's task even better.

In Presenter's separation of model (data) and view, distributed clients maintain the same underlying representation of presentation state but present different views of that information based on the role (instructor, student, or public) and personal preferences of the user. This design allows customization of the interface to the requirements of each user or class of user. Nonetheless, unifying the underlying data maintains and simplifies network communication and respects the central importance of the mediating artifact by keeping the presentation as a *shared*, not fractured, artifact.

Presenter now implements separation of views on several levels. The underlying slide data carries visibility tags, allowing display of different slide images for different participants, *e.g.*, "instructor notes" that are visible only on the instructor's display. Different participants have different user interface configurations based on their roles, although the user interface can also be individually customized. For example, the instructor's display (but not other roles' displays) shows the filmstrip. However, individual instructors can customize the filmstrip: resizing it, moving it to the right or bottom of the screen, or hiding it.

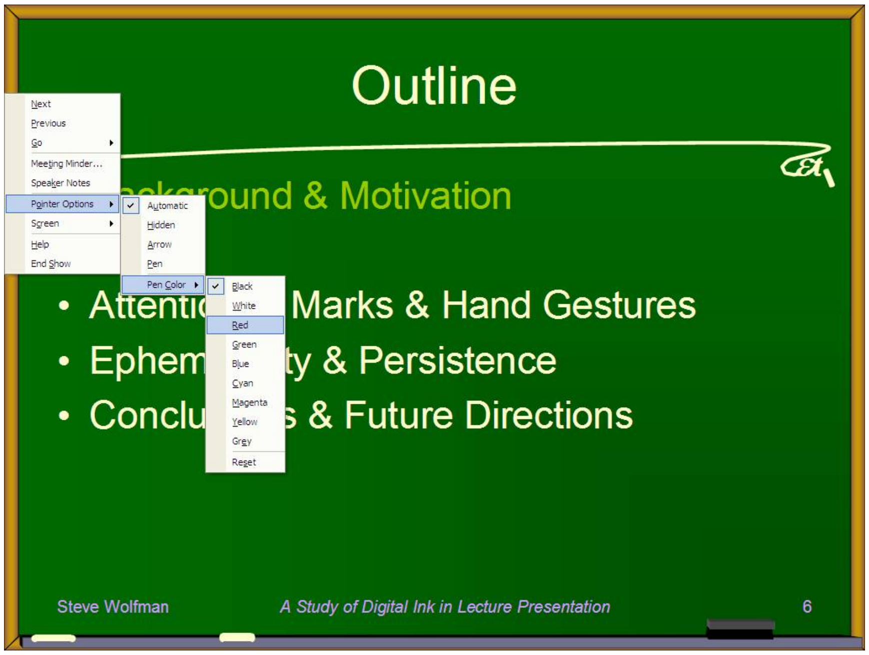
Separated views have proven valuable and resolved some of the tension among conflicting interface demands from different users. However, despite recognition of the "mediating artifact" design principle's centrality, separation of views entails some fragmentation of the once shared artifact, generating new tension between private customizations and the power of the shared, public artifact. The next two subsections describe the key benefits and primary challenges arising from separation of views.

Benefits of separation of views

Separation of views in Presenter has created several interface and architectural benefits. (1) Each role's view is tailored to contain only the controls and informational displays necessary for that role, devoting remaining space to the largest possible view of the current slide. (2) Instructors can tailor slide content to different views, *e.g.*, leaving private notes to themselves on the slides. (3) All Presenter users can manage their own windows and displays external to the Presenter application. (4) Finally, the separated view architecture naturally supports a distributed application architecture and extensions for new types of users such as students or teaching assistants.

In its current form, Presenter exposes only the controls and informational displays used by each role to that role. The public display shows only the current slide, scaled to fill the entire screen. The instructor's display includes controls, such as those for navigating through the slides; informational displays, such as a display showing the number of network clients connected to the presentation; and menus for more complex control operations, such as the `File > Open` menu option for opening a new Presenter deck. These targeted controls allow instructors to rapidly and effectively control and understand the current state of the presentation. In contrast, consider the standard PowerPoint 2002 (XP) presentation model in which the instructor's view is identical to the public display view (see Figure 3.15). In order to maximize slide size on the public display and avoid clutter, the instructor's controls are hidden. Therefore, even simple tasks require complex, hidden interfaces. For example, to change the pen color to red, the instructor either right-clicks or clicks on the small hidden control menu in the lower left. Then, she navigates to the `Pointer Options` sub-menu, to the `Pen Color` sub-sub-menu, and selects `Red`. Furthermore, all these distracting interface actions are visible on the public display. In Presenter, the instructor simply clicks on the red color button exposed on her private display, as in Figure 3.16.

Separated views of the controls surrounding the slide leads naturally to the notion of separated views of the slide itself. Figures 3.17 and 3.18 show one scenario when an instructor might want a slightly different view of a slide from the public view. In this figure, the public slide shows a question and some data relevant to answering the question. The instructor's view of the slide contains extra notes and sketches outlining the instructor's intended answer to the question. Conceptually, this is similar to the "notes" field in PowerPoint. Indeed, under certain conditions — dual-monitor display and appropriate startup sequence — PowerPoint notes are visible on the local display at the same time as slides



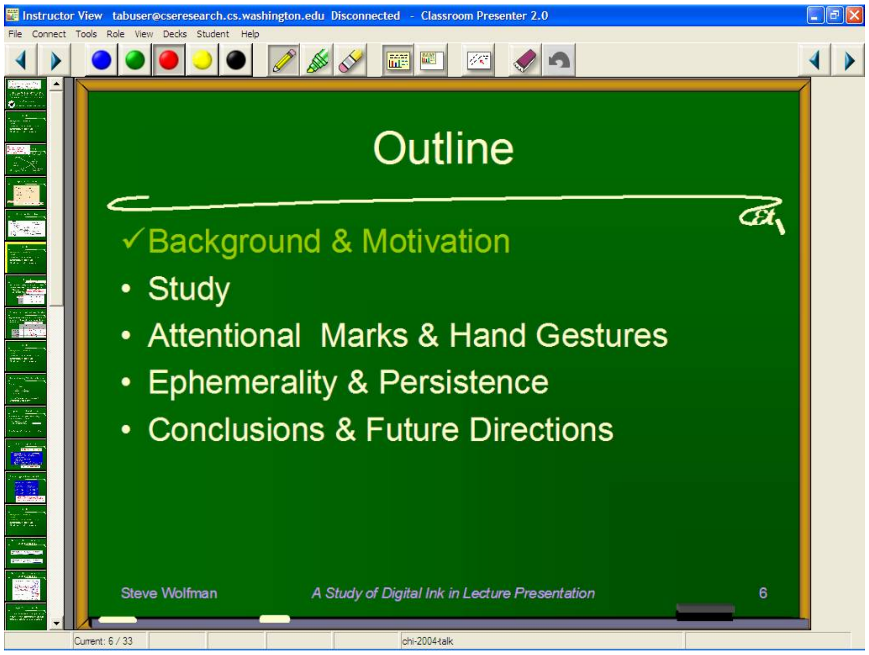
Outline

ckground & Motivation

- Attentio Marks & Hand Gestures
- Ephem ty & Persistence
- Conclu s & Future Directions

Steve Wolfman A Study of Digital Ink in Lecture Presentation 6

Figure 3.15: Changing the pen color to red in PowerPoint. The instructor right-clicks the slide and navigates menus and sub-menus to the appropriate selection.


Instructor View tabuser@cseresearch.cs.washington.edu Disconnected - Classroom Presenter 2.0
File Connect Tools Role View Decks Student Help
Current: 6 / 33
chi-2004-talk

Outline

✓Background & Motivation
- Study
- Attentional Marks & Hand Gestures
- Ephemerality & Persistence
- Conclusions & Future Directions

Steve Wolfman A Study of Digital Ink in Lecture Presentation 6

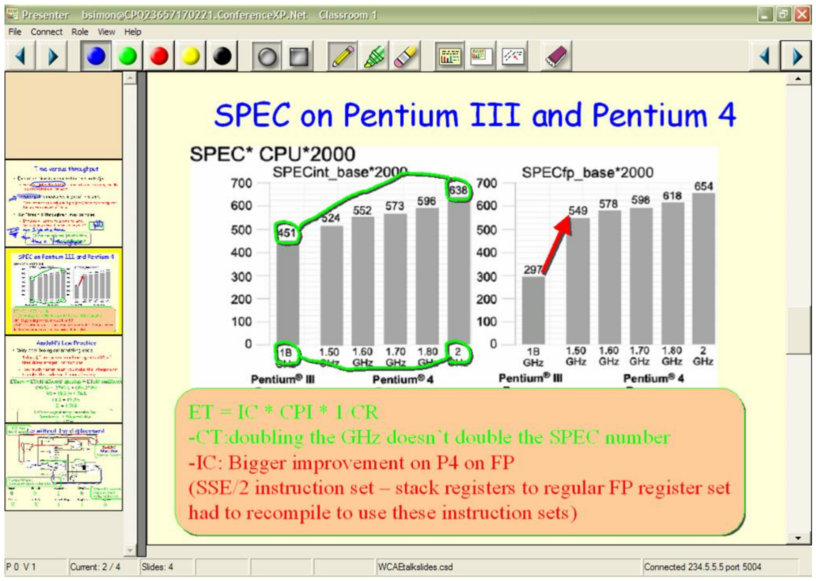
Figure 3.16: Changing the pen color to red in Presenter. The instructor clicks the red color button in the toolbar.

are shown on the projected display. Presenter's notes differ in that they are directly embedded in the slide, allowing instructors to use any standard PowerPoint object in the notes — such as text boxes, circles, or the digital ink objects used in Figure 3.17 — and to leverage slide context for meaning — as with the inked lines and circles in Figure 3.17 connecting numbers on the slide.

We developed an add-in (extension module) to PowerPoint that enables instructors to design different views for different Presenter roles (see Section 3.3.3). Instructors have made extensive use of the add-in to insert private notes and diagrams for in-class use. They have also suggested innovative uses of the design environment itself: creating "archival" notes meant to explain the slides when they are used in future terms, distinguishing between viewable and printed content in order to print student slide copies with "holes" in them, and leaving student-only notes that elaborate on slide content or provide references to textbooks and other resources.

Instructors can also take advantage of Presenter's separation of views to arrange their computers' desktops to display information that is important to them. The public display shows a full-screen slide image regardless of the layout of the instructor's display. Therefore, instructors can keep private resources, such as a system clock or a reference web page, open and available on their displays without interfering with the public view of the slides. However, this separation also leads directly to unintended problems described below.

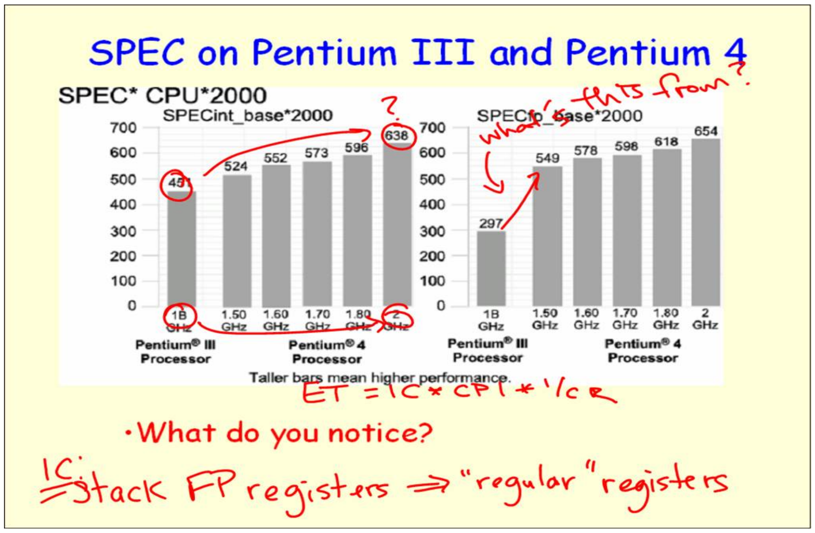
At the architecture level, the separation of views by roles naturally supports Presenter's design as a distributed application and extensions to Presenter to incorporate new types of users, such as teaching assistants. Presenter works as a distributed application by synchronizing an underlying data structure representing the state of the presentation. Each instance of Presenter then generates a customized view of the identical underlying state based on the instance's role (and individual customizations). Separation of views makes rich distinctions at the interface level natural even atop a simple underlying model of the shared data. Presenter's role structure also provides a natural way to introduce new types of users. New classes of Presenter users can be supported by adding new customized views that render


Presenter bsimon@CP023657170221.ConferenceXP.Net Classroom 1
File Connect Role View Help
SPEC* CPU*2000
SPECint_base*2000
SPECfp_base*2000
451 524 552 573 596 638
297 549 578 598 618 654
1B GHz 1.50 GHz 1.60 GHz 1.70 GHz 1.80 GHz 2 GHz
Pentium® III
Pentium® 4
P 0 V 1
Current: 2 / 4
Slides: 4
WCAEtalkslides.csd
Connected 234.5.5.5 port 5004

SPEC on Pentium III and Pentium 4

ET = IC * CPI * 1/CR
-CT:doubling the GHz doesn`t double the SPEC number
-IC: Bigger improvement on P4 on FP
(SSE/2 instruction set – stack registers to regular FP register set had to recompile to use these instruction sets)

Figure 3.17: An instructor's view of a slide from an architecture course with "instructor notes", slide objects visible only to the instructor. The two inked lines and circles on the left graph, the red arrow on the right graph, and the large, shaded text box at the bottom are all instructor notes. Compare this slide to Figure 3.18 below, which shows the student view after the instructor annotated the slide with ink.



SPEC on Pentium III and Pentium 4

ET = IC * CPI * 1/CR

• What do you notice?

IC: Stack FP registers ⇒ "regular" registers

Figure 3.18: The public view of the slide from Figure 3.17. The "instructor notes" are invisible here; however, as part of her discussion of the slide, the instructor wrote in much of the information from these notes.

underlying data in appropriate ways for those users.

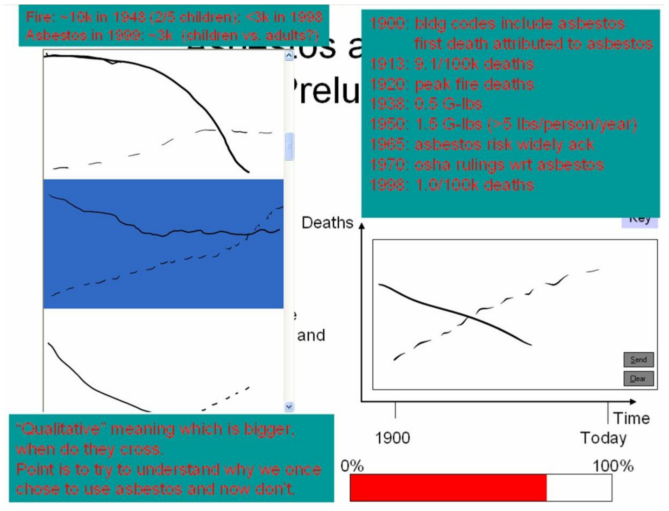
To summarize, role-based separation of views enables Presenter's users to avail themselves of the information and control they need for their individual tasks (*e.g.*, the filmstrip display for situating an instructor in her presentation's flow and navigating through the slides) and to envision new roles that reflect unexpected tasks (*e.g.*, a teaching assistant role or a role supporting "archival" notes that pass information from instructor to instructor and term to term).

Challenges of separation of views

Separation of views also introduced challenging problems into Presenter's design. In one sense, the idea of separated views violates the fundamental design tenet (from Chapter 2) to rely on the power of slides as a *shared* mediating artifact. Each difference Presenter introduces between the public view of slides and the instructor's view chips away at the integrity of the slides as a mediating artifact.

At the most basic level, the instructor's Presenter display is in a different place than the display students see. In contrast, with chalkboards, whiteboards, and SMARTboards [97], instructors and students share the same display. Standard computer-based slide presentation suffers a slightly less severe problem: the instructor and public displays are physically separate, but because they usually present the same information, instructors can focus primarily on the display students share. Overhead projectors provide an even closer connection, transmitting even instructor's actions on the "private display" (the overhead's projection table) to the public display through shadows.

Audiences and instructors using Presenter have noticed this disconnect in the mediating artifact. One student pointed out the effects of an instructor's shifts in gaze focus: looking at the class creates a connection (even without direct eye contact), looking at the screen engenders a feeling of joint focus, and looking down at the tablet encourages the student to look down at her lap. Instructors who appreciate integrated instructor notes on slides have also complained that they have to write *around* these notes in class, reducing the available

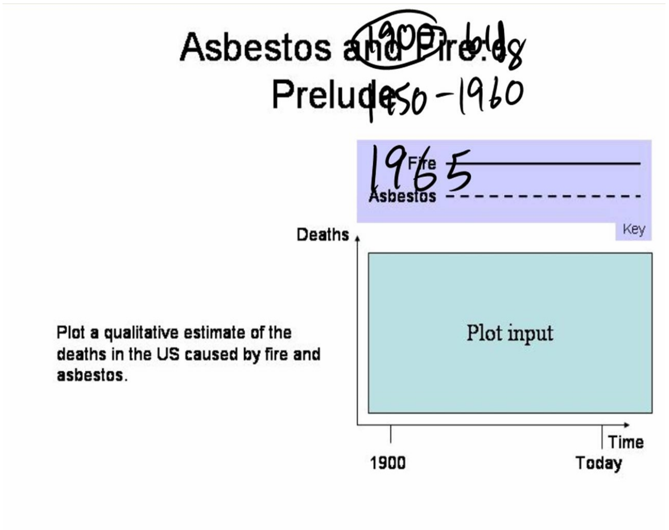


Fire: ~10k in 1948 (2/5 children); <3k in 1998
Asbestos in 1999: ~3k (children vs. adults?)

1900: bldg codes include asbestos
first death attributed to asbestos
1913: 9.1/100k deaths
1920: peak fire deaths
1938: 0.5 G-lbs
1950: 1.5 G-lbs (>5 lbs/person/year)
1965: asbestos risk widely ack
1970: osha rulings wrt asbestos
1998: 1.0/100k deaths

"Qualitative" meaning which is bigger, when do they cross.
Point is to try to understand why we once chose to use asbestos and now don't.

Figure 3.19: The first of two slides from the Structured Interaction Presentation System experimental class (described in Chapter 5) that illustrate problems with overwriting instructor notes. The darkly shaded textboxes, the graphs on the left, and the progress bar on the bottom are instructor notes. The instructor placed the note in the upper right over slide content that he did not need to see during class. Figure 3.20 shows how this caused a problem.



Asbestos and Fire:
Prelude

Plot a qualitative estimate of the deaths in the US caused by fire and asbestos.

Figure 3.20: The public view of the slide from Figure 3.19 with the ink drawn during class. The instructor inked over the text box in the upper right of Figure 3.19, assuming incorrectly that he had left blank space beneath it. As a result, the ink obscures slide text and both are difficult to read.

writing space and creating patterns of ink that are incomprehensible to the students (who cannot see the obstacles being avoided!). Of course, instructors *can* write over instructor notes as they write anywhere else on the slide, but doing so feels unnatural and may have unfortunate results. Figures 3.19 and 3.20 show a problematic case from the Structured Interaction Presentation System experimental class described in Chapter 5. The instructor overwrote a note intending to write in blank space, but instead his ink overlapped the slide title. A less obvious problem is the potential for unintentional reference to non-shared artifacts, *e.g.*, accidental spoken reference to hidden notes or even use of vocabulary from hidden notes. In either case, the separation of views subverts the artifact into dividing the instructor and student experiences rather than unifying them.

Still, these problems are far from apocalyptic or even endemic to Presenter. Instructors' notes tend to be concise reminder text that they speak or act on in class or diagrams they retrace with ink. In either case, these hidden notes become public eventually. Furthermore, instructors introduce schisms between public and private material by other means than Presenter. Many instructors bring written or printed notes to class. *All* instructors have different knowledge, experience, and perspectives from their students. Finally, *any* hidden or changing content on the slides (such as PowerPoint animations) could cause the inking problems shown in Figures 3.19 and 3.20.

The key design lesson for Presenter, then, is to remain cognizant of the tradeoff between "purity" of the shared, mediating artifact and integrated support for separated views. As with the overall design of Presenter, this means attending not only to what is possible in the system but to what is enabled, supported, and encouraged. For example, the design environment for Presenter currently places new objects as instructor objects when in the instructor view. Objects are only placed as "unrestricted" (visible in all views) if the display is set to the unrestricted view when they are placed. The design environment might instead place all objects with unrestricted visibility so that instructors must make a conscious choice when they wish to hide objects and fracture the mediating artifact.

At a more technical level, separation of views also complicates the sharing of content originating outside of Presenter. Computer science instructors, in particular, often want to share displays of outside applications such as compilers, program output, or web browsers. Sharing these applications is trivial when the public display simply reproduces the private display. However, when the public display differs from the private display, sharing these applications can be more difficult. This is true even when using standard PowerPoint but displaying the presentation on a second video feed and the PowerPoint source on the instructor's private display (a scenario favored by some instructors who use PowerPoint's notes facility). By default, most applications shown and controlled on the instructor's private display will not be visible on the public display (and vice versa) in this configuration.

Similarly, Presenter can make it difficult to share outside applications with the class. If an instructor drives the projector directly from her computer but using a second video feed (see the single-computer, dual-display described in Section 3.3.2), she must show and manipulate applications on the public display with no corresponding private display, a challenging feat made especially difficult by the pen interface! If the instructor controls the presentation from one computer networked to another that drives the public display (see the two-computer networked configuration in Section 3.3.2), the instructor can easily show other applications by running them on the computer driving the public display. However, (1) this requires switching computers to use the applications, (2) instructors cannot use digital ink to annotate the applications, and (3) only the local public display (and not any student or remote displays) will show the applications.

Underlying these implementation problems is a deeper design problem: distinguishing between public and private information. Presenter's separation of views is one flexible but imperfect mechanism for making this distinction: only information relevant to a given role is displayed for that role. The "replicated display" model described above is an alternate mechanism in which *all* information is shared. This is a less flexible solution that loses several of the benefits of separated views described in the previous section. However, it does resolve the challenge of sharing a variety of applications in class. A scheme that resolves this challenge and still maintains the advantages of separated displays must define a more flexible policy for distinguishing between public and private information (and more flexible mechanisms for specifying whether and which information is private).

3.2.3 Distributed Application

Presenter is distributed so that each client shares underlying slide, ink, and navigation data while presenting a view and controls tailored to its users. Presenter clients keep their data synchronized using networked (wired or wireless) communication. As a distributed application Presenter is a natural fit for distance learning deployments (which, in fact, were among its initial use scenarios). Another advantage of the distributed model is untethering the instructor from the data projector so that he has the freedom to move about the classroom.

Making Presenter a distributed application has clear disadvantages as well. Transmission over the network induces degradations in ink synchronization that are particularly problematic for gestural communication [54], exactly the sort of communication that dominates ink use in Presenter [12]. Although the Tablet PC itself is tuned for rapid capture of ink data and simultaneous rendering of that data to the screen, it does not support rapid transmission and rendering across a network. To keep network traffic in check, Presenter sends multiple ink samples as a single bundle rather than sending each sample as it is captured. The result is a periodic lag in rendering of ink on remote displays visible as a "jerkiness" in the flow of ink. With carefully tuned networking Presenter might maintain, on average, the same sample rate as for local capture. However, the distributed application would still suffer from network latency and jitter effects [54].

Scaling a distributed architecture to multiple computers also requires tradeoffs between delay, network traffic, and transmission reliability. Presenter uses multicast networking to support scaling to large numbers of distributed clients, but the multicast protocol is unreliable, causing occasional loss of transmitted information. Future versions of Presenter will explore reliable multicast (*i.e.*, Pragmatic General Multicast [98]) and selecting among alternative network protocols depending on the client topology, such as using reliable, point-to-point TCP for two-client topologies. Nonetheless, any scheme must sacrifice short delay, low traffic, or reliability to achieve gains in the others.

3.2.4 Other Design Goals and Decisions

Besides realistic ink, separation of views, and distributed architecture, several lesser design goals and design decisions also informed Presenter's development. This section describes these choices and their impacts on Presenter's design.

Low cognitive load

One design goal was maintaining a low cognitive load on Presenter's users (both instructors and students). Detailed discussion of this design goal is outside the scope of this dissertation. However, the core idea was to provide all Presenter controls needed during normal classroom use as large, easily accessible controls requiring only a single click to use. The most obvious consequence of this design goal is the set of large control buttons across the top of the instructor's view of Presenter (see Figure 3.6 for a close-up of these controls). A subtler example is color control across the pen and highlighter. Initially, color selection and pen type selection (*i.e.*, pen *versus* highlighter) were independent. That is, selecting red would make both the pen and the highlighter red at their next uses. However, because instructors rarely wish to highlight in the same color as their pen (due to low contrast), this design entailed multiple clicks to switch between pen and highlighter: changing pen type and then changing color. Therefore, the current version of Presenter preserves color choices separately for the pen and highlighter, and the instructor can switch between, *e.g.*, a red pen and a blue highlighter with a single click to change pen type.

Layered slide rendering architecture

Another design goal was to support extensions (including the Classroom Feedback and Structured Interaction Presentation Systems described in Chapters 4 and 5). Presenter's slide rendering architecture directly responds to this goal. The architecture mimics photo editing systems' layered designs. Each layer independently paints itself and responds to or captures interface events. Each slide display composites its set of layers into a single image. Presenter uses different stacks of layers to display and solicit different information for different participants. For example, the Classroom Feedback System introduces several visualization layers that summarize student feedback on a slide's bullet points, *e.g.*, by drawing bar graphs next to each bullet. In general, the layered rendering architecture simplifies support for separation of views by transforming the problem of customizing a slide display into the problem of selecting a set of layers to render.

Presenter's layered design also created several problems. The layered design, and particularly automatic transforms for scrolling and scaling, complicates the use of coordinates that are "absolute" (or at least relative to the window size, screen size, or screen resolution). Yet, such calculations are important, *e.g.*, for rendering fonts. Calculations like this are currently supported through access to the cumulative transform imposed on a given layer, through helper functions for common operations (like computing the largest font at which a rendered string fits within a specified box), and through exposing the underlying user interface control used to render the composited layers.

Each additional layer also imposes significant rendering cost in Presenter's current architecture, although Presenter limits that extra cost by combining and caching some layers and limiting layer rerendering to changed regions. Presenter caches layers that change rarely in practice and combines neighboring groups of these static layers by stacking them and rendering them into a single cached image. Furthermore, even dynamic layers can be combined with underlying static layers if they disallow "erasing" existing paint from a canvas (allowing underlying paint to show through) since changes to the dynamic layer will only obscure, not reveal, content in the lower layers.

So far, we have created the following layers for Presenter and its extensions:

- slide display layer: renders the underlying slide image

- student and instructor ink capture layers: captures and (on the local computer) renders a Presenter user's ink; to exploit optimized Tablet PC routines for capture and immediate rendering, these layers fall outside the standard layering architecture

- student and instructor ink display areas: renders ink from one Presenter user on another Presenter user's display

- interactive widget layer: renders the interactive interface elements that instructors compose to create exercises in the Structured Interaction Presentation System; to simplify the initial design of these elements, this is currently a custom layer

- highlight layer: accepts right-clicks to toggle display of box highlights around slide geometry elements (such as bullet points or diagrams)

- geometry layer: renders all detected geometry on a slide for debugging and for association of Classroom Feedback System feedback menus with slide regions

- remote pointer layer: displays an icon that follows the instructor's cursor on remote displays

- comment control layer: accepts user input to create and withdraw feedback items in the Classroom Feedback System

- feedback summarization layers: a set of related layers that give visual summaries of student feedback in the Classroom Feedback System, including blobs of color, histograms, and text

Embedding meta-data into PowerPoint

One simple but powerful design decision was to use self-contained native PowerPoint documents as the primary source form for Presenter presentations. Although Presenter actually loads its own custom format, this format is normally used only as a temporary, intermediate representation while the authoritative source is the native PowerPoint file. (The intermediate representation also allows instructors to use other source formats, although this capacity is currently limited to importing slides as images.) The Presenter System includes a PowerPoint add-in (see Section 3.3.3) that unobtrusively embeds all Presenter meta-data directly into PowerPoint source files' data structures through native extension mechanisms. Although Presenter's meta-data can only be manipulated using the Presenter add-in, the resulting PowerPoint source files are still viewable, editable, and usable *without* the Presenter add-in. One alternative — or really an alternate class of decisions — would be to place meta-data in an associated file, as with CounterPoint [51]. Another alternative would be to embed meta-data in the PowerPoint data structures in such a way that the add-in was required to use the resulting data.

The critical reasons for choosing to embed Presenter meta-data directly in the native source file were: to maintain partial compatibility with unaugmented PowerPoint installations and to ease management of Presenter source files. Instructors can use unaugmented PowerPoint to open, view, edit, or present from Presenter-enhanced source files. All of their presentation objects visible in each view (see Section 3.3.3) will remain in the presentation although those outside the last view used in the add-in software will be invisible. Instructors can therefore share their presentations with students who have not adopted Presenter and use the presentations on computers that have not been Presenter-enabled. Both of these were important design concerns.

A "separate file" model for meta-data would also have supported compatibility; however, instructors would need to understand the two-file model to manage their Presenter source. An instructor copying her Presenter source presentation from one computer to another would need to know to copy both the PowerPoint file and the auxiliary presenter file. Furthermore, edits to one of the files might corrupt its connection to the other file. With embedded meta-data, instructors can manage their presenter source files as single, standard files. Easy management of these files also simplifies sharing of presentations among instructors.

The primary disadvantages of this choice are code complexity and performance. Because PowerPoint's native extension mechanisms were not designed to support Presenter's needs, Presenter's add-in uses a custom-built API atop the extension mechanisms. This API masks the extension mechanism's structural limitations (*e.g.*, all extension data takes the form of association lists from strings to strings) and code quirks (*e.g.*, inconsistent treatment of letter case and blank entries). Both the API and PowerPoint's implementation of extensions induce performance overhead. However, neither of these disadvantages is a compelling reason to switch strategies.

Multicast networking

The use of multicast protocols for networking in Presenter was a difficult design decision that entailed a variety of tradeoffs. Presenter's primary communication layer is a multicast protocol called RTP [91] implemented as part of the Microsoft ConferenceXP project [72]. Each network client subscribes to one or more multicast addresses. Data sent to a multicast address is broadcast to all clients subscribed to that channel. A "time to live" parameter on data packets limits these multicast addresses to virtual "cells", like cellular networks' spatially distributed cells that use the same frequency band.

The choice of multicast was a tradeoff. Multicast makes sense for most instructor network data. This data comprises ink and navigation commands that all other participants (students and public displays) are likely to need. Multicast is theoretically a scalable solution for transporting this information to all clients, a substantial win for large classes in which every student runs a Presenter client.

However, multicast protocols (besides recent research protocols [98]) are not reliable. That is, they do not guarantee delivery of individual packets. Reliability is unnecessary for the video and audio data often sent using RTP because lost data results only in transient and graceful degradations in quality of service. However, unlike video and audio data, Presenter's navigation and ink data is inherently persistent, and navigation data is inherently discrete. Therefore, losing this data causes significant degradation in performance manifested as lost slide transitions or dropped ink strokes, and this degradation potentially persists indefinitely.

Presenter addresses this problem for small, slowly changing state data by periodically broadcasting the data from the instructor's computer. For example, Presenter periodically broadcasts the current slide number. If a slide change packet drops between the instructor and a student, the student's client will stay on the old slide. However, the next periodic broadcast received on the student side will contain the new slide number, and the student's client will transition to the new slide. Unfortunately, ink data is too rapidly changing and too large to use this periodic rebroadcast strategy. Therefore, dropped strokes are a significant concern in the multicast environment. Broadcasting very large pieces of data (like the slides themselves) can also be problematic in a multicast environment, as any such broadcast is likely to be lose one of its many constituent packets even with low packet loss rates.

Another concern with multicast is limited institutional support for multicast protocols. Currently, many networks and routers are configured to discard multicast traffic. As a result, awkward failures of service can arise even for intranet use. For example, Presenter often works between computers that both use the same wireless or the same wired network but fails across wireless/wired boundaries because a router connecting the networks is configured to discard multicast packets. Multicast may be disabled because the network administrator saw no need to enable yet another potentially dangerous service, because of specific policy decisions, or just by default. Regardless, this underlying technology adoption problem is a significant challenge for all applications built on multicast. We have addressed this problem by building alternate connectivity layers supporting TCP/IP, a reliable unicast protocol, beneath Presenter. In the long run, ConferenceXP, which handles Presenter's multicast network, will support unicast tunnels, alleviating the problem.[4]

[4]In a sense, Presenter itself is a step toward securing multicast adoption. As Presenter and other multicast applications create demand for multicast, adoption is likely to follow.

Finally, multicast is not necessarily appropriate in all Presenter use scenarios (*e.g.*, two-computer scenarios, see Section 3.3.2) or for student-to-instructor communications. For the two-computer case, multicast is clearly unnecessary; Presenter can instead use reliable TCP/IP networking. In both of the multicast-based student communication extensions to Presenter — the Classroom Feedback System (described in Chapter 4) and student submissions [96] — student communication is logically private from student to instructor, *i.e.*, not exposed to other students or the public display. Therefore, multicast is a mismatch with the logical network structure. Furthermore, multicast in these cases represents a privacy threat, exposing students' responses unnecessarily to third parties. For now, these extensions just use multicast for student-originated communication, both because wasted resources used by student multicast communications are minimal and because Presenter does not yet take security concerns into account.

Archiving

Another design goal in Presenter was to support low-cost archiving and replay of presentations. Although archiving is an important feature for users, detailed description of archiving in Presenter — including technologies for archiving, motivation for use of archives by various Presenter stakeholders, and use patterns of archives — is outside the scope of this dissertation. Several research projects have already investigated archiving of computer-based presentations from different perspectives [1, 59]. Archiving in Classroom Presenter is also described elsewhere [107].

However, archiving is apropos insofar as it supports effective analysis of Presenter's classroom use and facilitates design iterations. Archives of Presenter use (and of the Classroom Feedback and Structured Interaction Presentation System extensions, described in Chapters 4 and 5) supports analysis by allowing researchers to carefully review classroom use of the tools. Researchers review class data with instructors and students to help them clearly recount classroom experiences and review class data independently to rigorously code observed activities. Archives also allow automatic analysis of tool use. Finally, archives support the design process by allowing simulated testing of new designs. For example, an archive of ink drawn in an old version of Presenter could be re-rendered with a new ink rendering algorithm to investigate the properties of the new algorithm.

Archiving in Presenter is accomplished through the use of two tools.[5] "Snooper" is a simple, custom-built archiving system that joins the multicast group of a presentation, logs all packets transmitted, and later replays them. Snooper also has a variety of more advanced capabilities, such as customizable filters to exclude certain types of packets. The second archiving system, ConferenceXP WebViewer [107], creates and replays integrated archives of audio, video, ink, and navigation commands from Presenter. WebViewer's archives are specifically targeted for students' review of class sessions.

3.3 Presenter System Description

This section elaborates Section 3.1, providing a detailed account of Presenter's network connectivity, use scenarios, and design environment as they stand in the current version.[6] Like Section 3.1, this section focuses on the instructor and public displays.

3.3.1 Network Connectivity Sequence

This section describes the sequence of steps for establishing a network connection and sharing critical data among Presenter clients. (Section 3.2.4 gives details on the particular network architecture used in Presenter.) This section also assumes that the underlying network used by the clients is available to every client and forwards multicast traffic. Addressing problems with these two assumptions is outside the scope of this dissertation.

The standard way to establish a connection among several Presenter clients is to change each client to the role it will perform in the presentation (*i.e.*, instructor, student, or public) and then connect to a common multicast address. In many cases, Presenter's users will

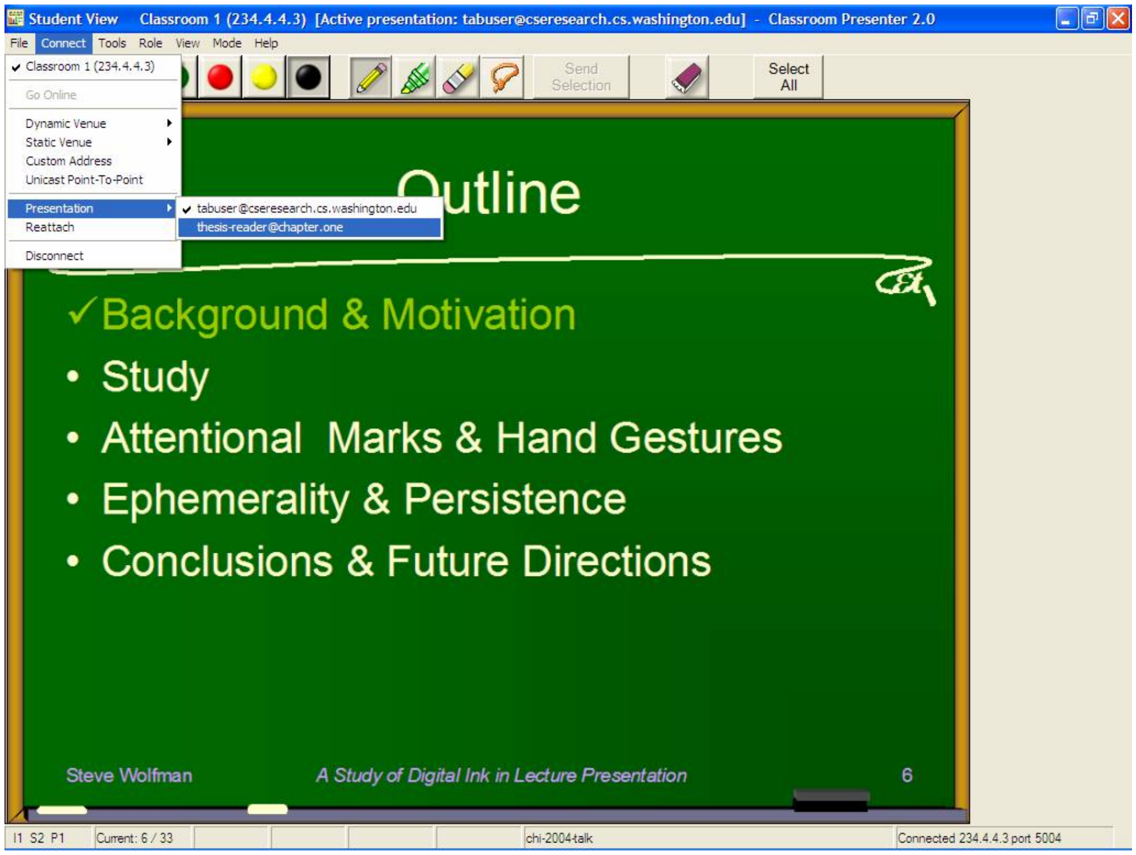
[5]Actually, just saving completed Presenter presentations with their ink is a low-cost, albeit low-fidelity archiving option.

[6]Classroom Presenter 2.0, development version 1.9.8, built 2004 March 22.

be confident that no other presentation can occupy the same multicast address — *e.g.*, if the network used is stand-alone or only routes multicast traffic internally. In such cases, each client can simply connect to the default address that appears as the first option in the Connect menu. Alternatively, Presenter supports both static addresses (addresses that are always available, listed in an XML configuration file) and dynamic addresses discovered from a ConferenceXP server advertising multicast venues [72]. Both of these are listed in sub-menus of the `Connect` menu. Finally, for two-computer communication, one computer can make a (reliable) unicast connection directly to the other computer's IP address. Note that there is no built-in mechanism for negotiating among several clients *which* multicast address to use; instead, users must agree on the multicast address *a priori* or out-of-band (*e.g.*, by voice, instant messaging, or telephone).

Several factors complicate this basic scenario. First, it is possible for several instructors to use the same multicast address at the same time. In this case, each instructor's client advertises its presence, and student or public clients connecting to the address must choose which instructor to attach to for their presentation. Currently, each client chooses arbitrarily, indicates its choice on the title bar, and indicates the presence of multiple instructors in a status display. Clients may select among instructors using the `Connect` menu's `Presentation` sub-menu, which dynamically updates to list the instructors advertising presentations on the current multicast address. Figure 3.21 shows one student view after connection to a venue with multiple instructors. "Active Presentation: wolf" in the title bar indicates that the client chose "wolf" as the instructor. "I 2" on the left of the status bar across the bottom indicates two instructors at the current address. Finally, the student is currently using the `Presentation` sub-menu to select between the two instructors.

The second complicating factor is that switching to the public role puts Presenter into a full-screen view with no menus available, and thus removes connectivity options. This is a known problem in Presenter's interface. Currently, public Presenter clients are connected either by connecting in another role (student or instructor) and switching to the public role or by switching to the public role, hitting Escape to exit the full-screen view, connecting,


Student View Classroom 1 (234.4.4.3) [Active presentation: tabuser@cseresearch.cs.washington.edu] - Classroom Presenter 2.0
File
Connect
Tools
Role
View
Mode
Help
✓ Classroom 1 (234.4.4.3)
Go Online
Dynamic Venue
Static Venue
Custom Address
Unicast Point-To-Point
Presentation
Reattach
Disconnect
✓ tabuser@cseresearch.cs.washington.edu
thesis-reader@chapter.one
Send Selection
Select All
I1 S2 P1
Current: 6 / 33
chi-2004-talk
Connected 234.4.4.3 port 5004

Outline

✓Background & Motivation
- Study
- Attentional Marks & Hand Gestures
- Ephemerality & Persistence
- Conclusions & Future Directions

Steve Wolfman A Study of Digital Ink in Lecture Presentation 6

Figure 3.21: The student's view of Presenter in the "student submissions" extension (described in detail elsewhere [96]). Connection information for the student role is visible in the title bar across the top and the status bar across the bottom. The student is using the `Connect` menu's `Presentation` sub-menu to select among multiple instructors in one multicast venue. (Section 3.2.4 describes multicast networking in Presenter.)

and then reselecting the public role. A better design might automatically display a connection menu on switching to the public role when no connection has yet been established.

Designing the interface for connectivity has proven particularly challenging. Partly, this is due to the proliferation of usage scenarios for networking: single client unconnected, two clients connected by unicast, or multiple clients connected by multicast. The potential for multiple instructors on one multicast address further complicates the design. Finally, failures at the networking level are frequent, ranging from misconfiguration by the user to dropped packets.

The ConferenceXP venue service [72] might seem an ideal solution to the challenge of establishing connectivity, allowing users to select among familiar venues advertised by a known venue service. However, this solution creates several new problems. First, it requires an accessible venue service. This causes little trouble for Presenter topologies connected to the wider network, although it does introduce a new point of failure. In isolated Presenter topologies — *e.g., ad hoc* networking or standalone class-wide networks — some client must run the venue service, and discovering that computer may be complicated by dynamic addressing. Second, using ConferenceXP's venue service demands that users learn a separate interface (ConferenceXP) to use Presenter. The dynamic address menu in Presenter is a compromise that exposes the results of querying a venue service without introducing a new interface and fails gracefully when the venue service is unavailable.

Finally, slide decks also complicate network connectivity for two reasons. First, slide decks can be quite large, and so transmitting them can be slow, especially over unreliable multicast connections to many clients. Second, different Presenter users in a single session may reasonably wish to use slightly different decks. For example, imagine that an instructor begins discussing a deck one day in class but finishes only half of the slides. Between that class and the next, a student annotates his slides with private notes. The instructor also makes changes to the slides, perhaps correcting mistakes in the portion of the deck already discussed and adding new content or slides to the other portion. This student and instructor will wish to use *different* versions of the slide deck in the next lecture: the student for his annotations and the instructor for her changes. For both of these reasons, Presenter supports both “in-band” transmission of slides — in which the instructor broadcasts or students request individual slides or the entire deck — and “out-of-band transmission” — in which slides are loaded separately on the instructor’s and students’ computers (*e.g.*, by downloading from a web page) and matched across computers. When the decks are the same, matching them is straightforward. When the decks differ somewhat, Presenter uses heuristics to select a good match.

3.3.2 Use Scenarios

Presenter is commonly used in several different configurations. The primary configurations are: single-computer, single-display; single-computer, dual-display; two-computer networked; and multi-site networked.

In its single-computer, single-display configuration, the instructor directly projects her own view of the slides to the data projector. This requires video output from the instructor’s computer to the data projector through a video cable (although wireless options exist at higher cost and lower fidelity, *e.g.*, Projector People’s “Wireless Projector Adaptor” [118]). To accommodate this scenario, Presenter supports hiding distracting controls (like the toolbar or the filmstrip) and a full-screen mode in which the critical controls are available at reduced screen size (see Figure 3.8). The primary advantage of this configuration is simplicity: an instructor need only connect her tablet to the projector to begin presenting. The single-display configuration also makes it easy for instructors to display other applications, such as a development environment in a computer science course. Finally, ink rendering is as rapid and fluid on the public display as on the tablet in this configuration. However, this configuration has two primary disadvantages. Separation of views is impossible, eliminating features such as instructor notes and improved navigation. This configuration also generally tethers the instructor’s computer to a projector, reducing her mobility. (Of course, the instructor is still free to roam the classroom, but control of and annotation on the slides is only possible from the tethered tablet.)

The single-computer, dual-display configuration was created as a compromise for instructors who wanted the benefits of separated views but were frustrated by the time required to establish a network connection between two computers (in the configuration described below). In this configuration, a tablet with support for dual-display shows the instructor's view on the tablet itself and, through a video tether to a digital projector, shows the public view on the public display. This configuration allows full separation of views but does not require a network connection. Ink rendering on the public display is therefore reliable and relatively rapid, suffering no network lag. However, ink on the public display must still be rendered in bundles of samples (see Section 3.2.3), causing jerkiness in ink flow. Also, the instructor's computer remains tethered to the projector. Finally, displaying alternate applications on the public display is difficult in this configuration since they must either be configured to replicate themselves on the second display or moved to and manipulated on the second display (nearly impossible with only the pen).

The two-computer networked configuration is a generalization of the two-computer wireless configuration described at the start of this chapter. The instructor controls the presentation using Presenter's instructor view on her tablet. A separate computer (which need not be a tablet) drives the projector with the public view of the presentation. The computers must be connected by some network connection, either wireless, wired, or mixed and either infrastructure-based (*i.e.*, using a router) or *ad hoc* (*i.e.*, directly connecting the computers). The wireless case allows separation of views and full mobility for the instructor. One additional advantage of this configuration is easy display of alternate applications. The instructor simply runs and displays these applications on the tethered computer and swaps between them and Presenter's public display. However, transmission of ink in this configuration may be unreliable (leading occasionally to strokes missing from the public display), and public ink rendering suffers delay from network lag. Finally, the wireless configuration is inherently more difficult to secure than a single-computer configuration. The wired configuration is identical to the wireless configuration except for the reimposition of a tether, and improvements in reliability, lag, and security.

Finally, instructors can run between multiple sites in any of these configurations. Each site must establish a network (multicast) connection with the presenting site. Increased network delay between sites will lead to increased lag in ink rendering and slide navigation. Also, sharing alternate applications remotely requires separate software support.

3.3.3 Constructing Decks: DeckBuilder and the PowerPoint Add-In

Presenter slide decks are essentially ordered sequences of slide images. Initially, Presenter used a separate application called DeckBuilder to generate its slide decks. DeckBuilder loads PowerPoint presentations or individual slide images and saves Presenter presentations. Loading slide images preserves potential for compatibility with other presentation sources (*e.g.*, SliTeX and Apple's Keynote). DeckBuilder's basic operation is pictured in Figures 3.22–3.25.

To streamline the deck building process, we developed an add-in — extension module — to PowerPoint. The add-in allows an instructor to export a Presenter deck directly from PowerPoint (see Figure 3.30). The original intent of this add-in was to expedite instructors' workflow at a time-sensitive point in their working process. Several early Presenter users habitually made final adjustments to their slides within minutes before their classes started. At such times, a host of concerns press on the instructors, from technical issues with projection or microphones to student questions. Removing several steps from their workflow by folding DeckBuilder into their presentation design environment was a welcome improvement. In fact, several other, simpler modifications were made to target this same goal, *e.g.*, associating Presenter decks' extensions with Presenter so that double-clicking on the deck immediately opens it in Presenter and introducing a rapidly-selectable "default venue" on the `Connect` menu.

Initially, instructor notes (slide elements visible only on the instructor's view) were created by designing slightly different versions of a slide deck and separately building the decks for Presenter. One Presenter deck would contain the instructor's notes and be loaded on the instructor's computers. The other deck would hold the public view and be loaded on

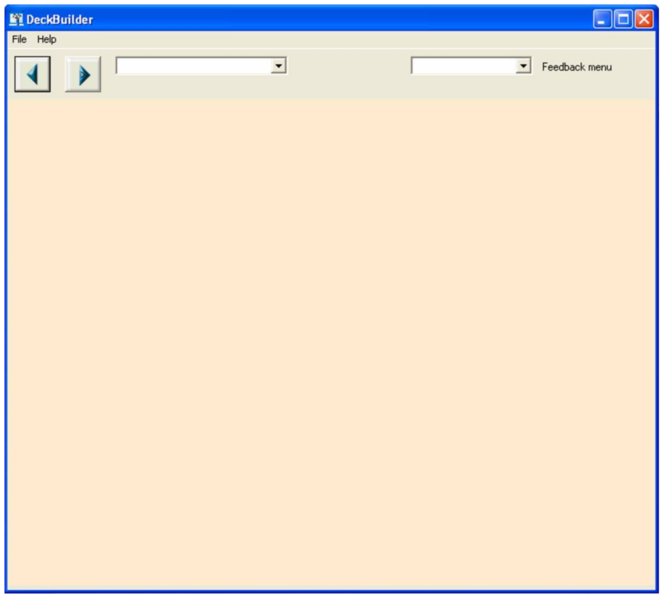


Figure 3.22: The start of a sequence of screenshots (ending with Figure 3.25) showing the DeckBuilder application for building Presenter presentations from PowerPoint source. This figure shows DeckBuilder with no presentation loaded.

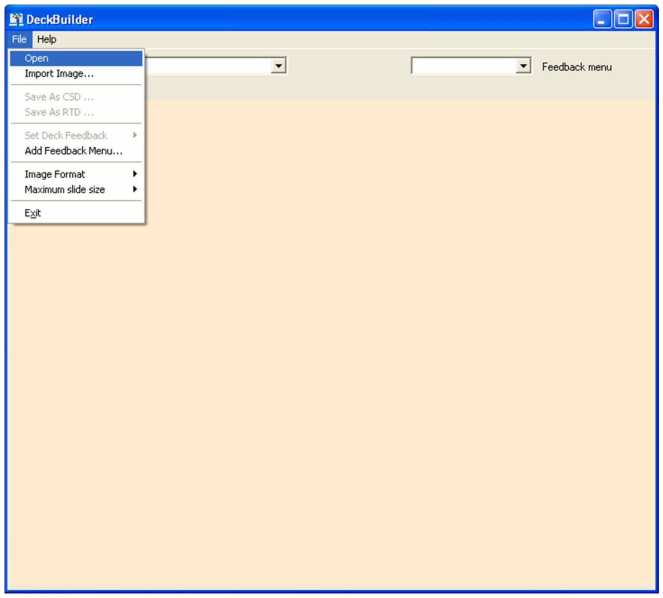
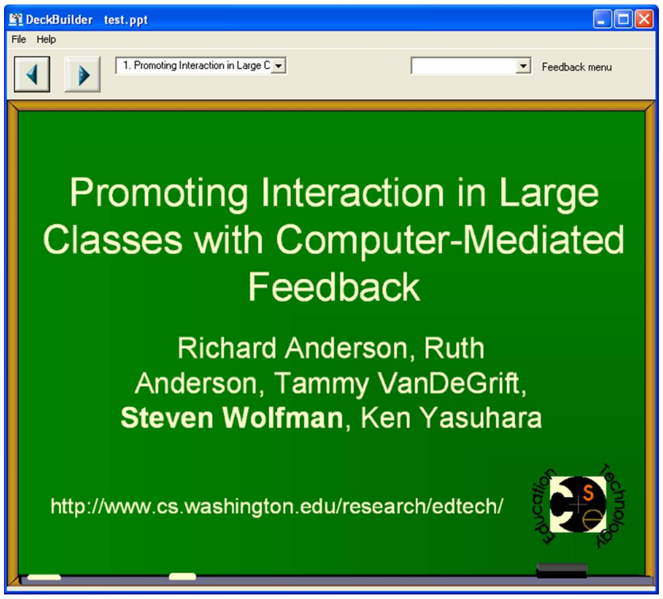


Figure 3.23: The instructor uses the `Open` option. This will call up a dialog for selecting the PowerPoint source file.

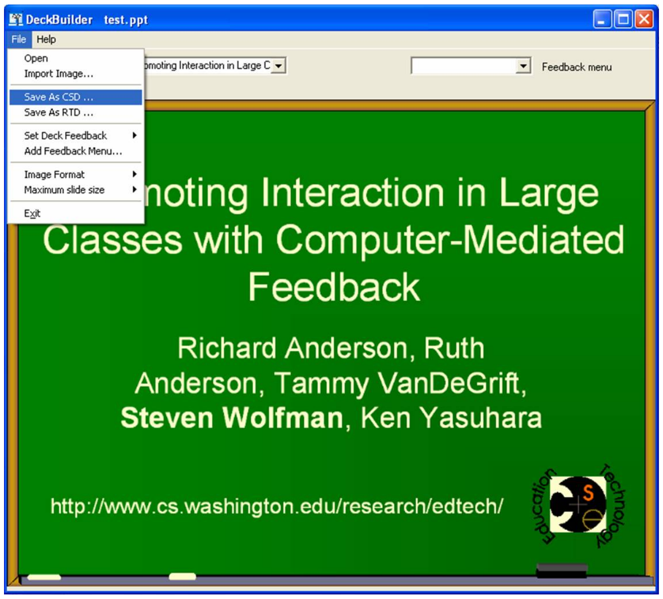


Promoting Interaction in Large Classes with Computer-Mediated Feedback

Richard Anderson, Ruth Anderson, Tammy VanDeGrift, **Steven Wolfman**, Ken Yasuhara

http://www.cs.washington.edu/research/edtech/

Figure 3.24: DeckBuilder with a sample presentation loaded. The left and right arrow buttons move through the presentation slides so the instructor can check their integrity and set options (*e.g.*, a Classroom Feedback System feedback menu, see Chapter 4).



noting Interaction in Large Classes with Computer-Mediated Feedback

Richard Anderson, Ruth Anderson, Tammy VanDeGrift, **Steven Wolfman**, Ken Yasuhara

http://www.cs.washington.edu/research/edtech/

Figure 3.25: The instructor uses the `Save as CSD` option to save the presentation in Presenter's native format. This will start a dialog to select the new file's name.

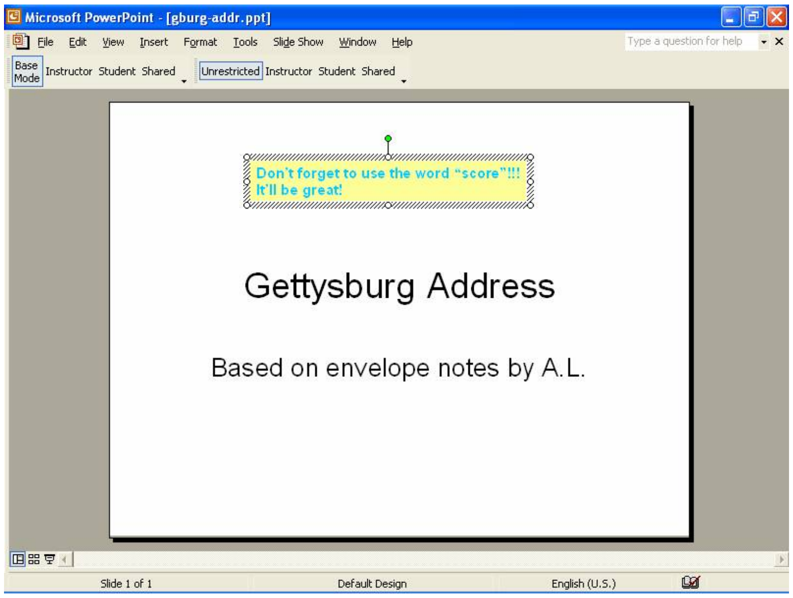
the projection computer. This *ad hoc* approach allowed early experimentation that proved the value of instructor notes at low development cost. Presenter's tolerance for surprising uses — *i.e.*, loading mismatched decks across clients — facilitated this experimentation.

This *ad hoc* instructor notes approach suffered from three clear problems. First, management of paired source presentations (instructor and public views) is unwieldy. Second, the complicated build process imposes an extra burden on the instructor during the already high-pressure period between finishing a deck and executing it in class. Third, the complex file loading process was dangerous. On at least one occasion, an instructor swapped the files for the public and instructor views with embarrassing consequences.

The problems with *ad hoc* instructor notes led us to develop an add-in to PowerPoint that directly supports instructor notes. The add-in addresses each of the problems above: it represents all versions of the presentation in a single presentation source file; it uses the same simple export process for building decks with instructor notes as for building a single view; and, because Presenter's deck format now supports multiple views, Presenter automatically presents the appropriate deck view based on its role (instructor, student, or public). Figures 3.26–3.30 show the primary features of the PowerPoint add-in.

Consider first the `Mode` toolbar, the left-hand toolbar in Figure 3.26. `Base Mode` shows all the normal content of a source presentation. By default, a presentation starts in base mode. Each other mode (`Instructor`, `Student`, and `Shared`) acts like a glass pane that can cover the presentation. That glass pane can have any standard PowerPoint object on it. When the pane is down, those objects are visible. When it is up, they are not. Furthermore, while a pane is down, any newly placed object falls on top of the glass pane and will thus be lifted the next time the pane lifts. Clicking a mode in the toolbar switches to that mode (raising the previous mode's pane and lowering the new mode's pane). Figures 3.28–3.29 show the process of switching between modes.

As long as an instructor wishes to display most content to all Presenter clients, most presentation source objects and most editing can occur in base mode. Edits to the content in base mode are automatically reflected in other modes. Note that managing edits across

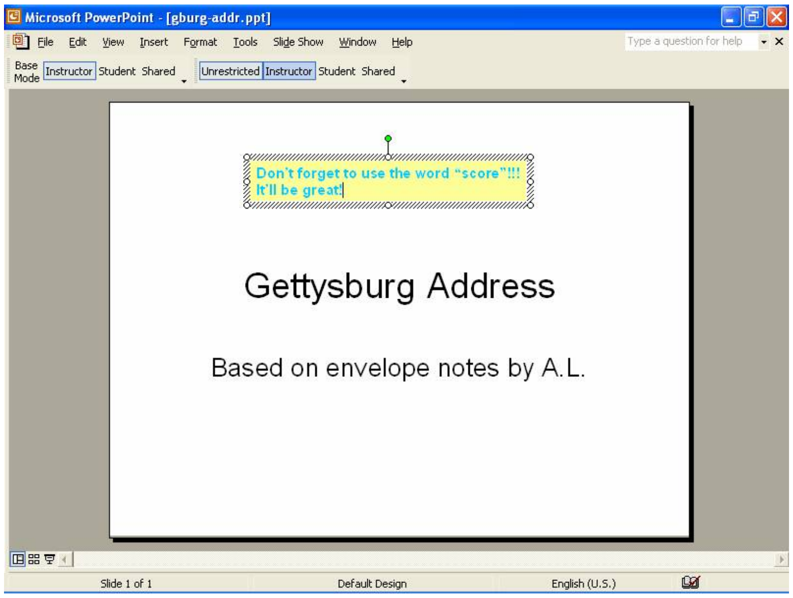

Microsoft PowerPoint - [gburg-addr.ppt]
File Edit View Insert Format Tools Slide Show Window Help
Type a question for help
Base Mode Instructor Student Shared
Unrestricted Instructor Student Shared
Slide 1 of 1
Default Design
English (U.S.)

Don't forget to use the word "score"!!! It'll be great!

Gettysburg Address

Based on envelope notes by A.L.

Figure 3.26: The start of a sequence of screenshots (ending with Figure 3.30) showing the PowerPoint add-in that manages instructor notes and exports Presenter decks. Here, the instructor has started creating an instructor note by selecting the textbox that will become the note. The left-hand toolbar's highlighted `Base Mode` button indicates that only unrestricted objects are visible. The right-hand toolbar's highlighted `Unrestricted` button indicates that the selected textbox is unrestricted.

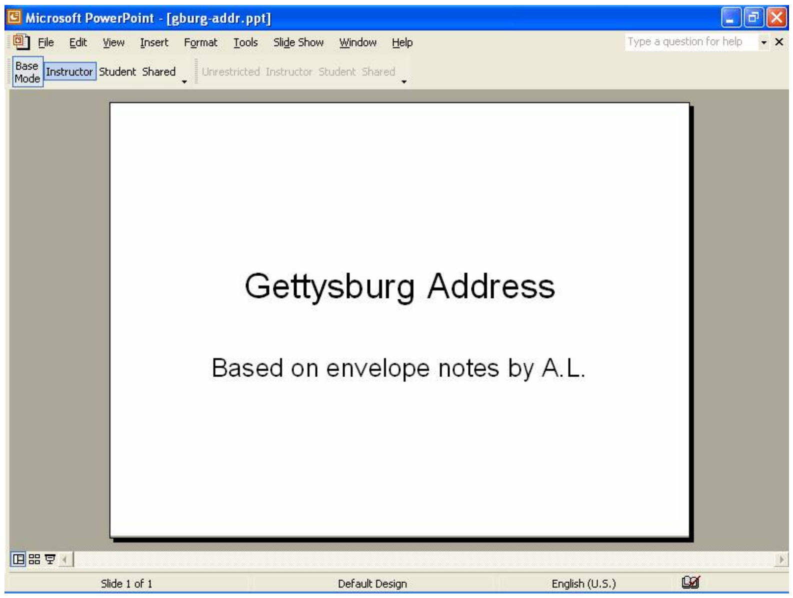

Microsoft PowerPoint - [gburg-addr.ppt]
File Edit View Insert Format Tools Slide Show Window Help
Type a question for help
Base Mode Instructor Student Shared
Unrestricted Instructor Student Shared
Slide 1 of 1
Default Design
English (U.S.)

Don't forget to use the word "score"!!! It'll be great!

Gettysburg Address

Based on envelope notes by A.L.

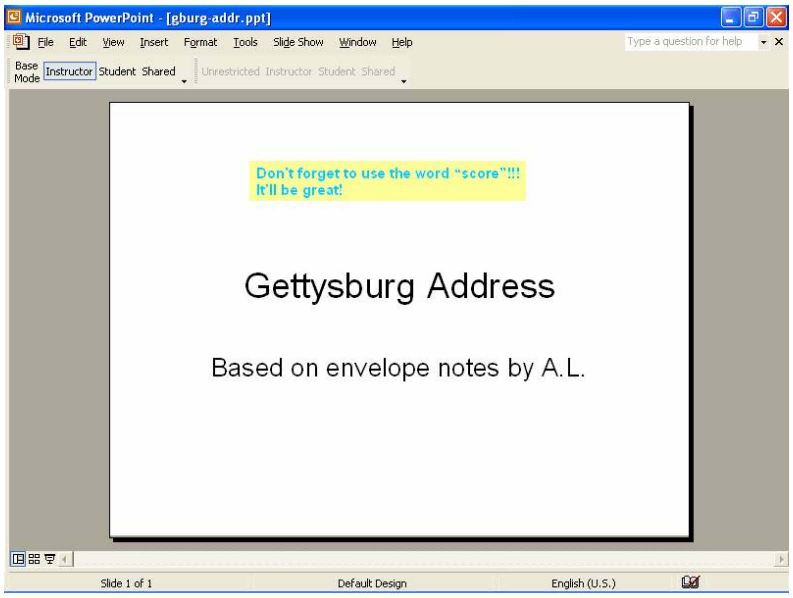
Figure 3.27: The instructor is restricting the selected textbox by clicking on the right-hand toolbar's `Instructor` button.


Microsoft PowerPoint - [gburg-addr.ppt]
File Edit View Insert Format Tools Slide Show Window Help
Type a question for help
Base Mode Instructor Student Shared
Unrestricted Instructor Student Shared
Slide 1 of 1
Default Design
English (U.S.)

Gettysburg Address

Based on envelope notes by A.L.

Figure 3.28: The textbox of Figure 3.27 has disappeared because it is no longer visible in base mode. The instructor is now switching to instructor mode by clicking on the left-hand toolbar's `Instructor` button.

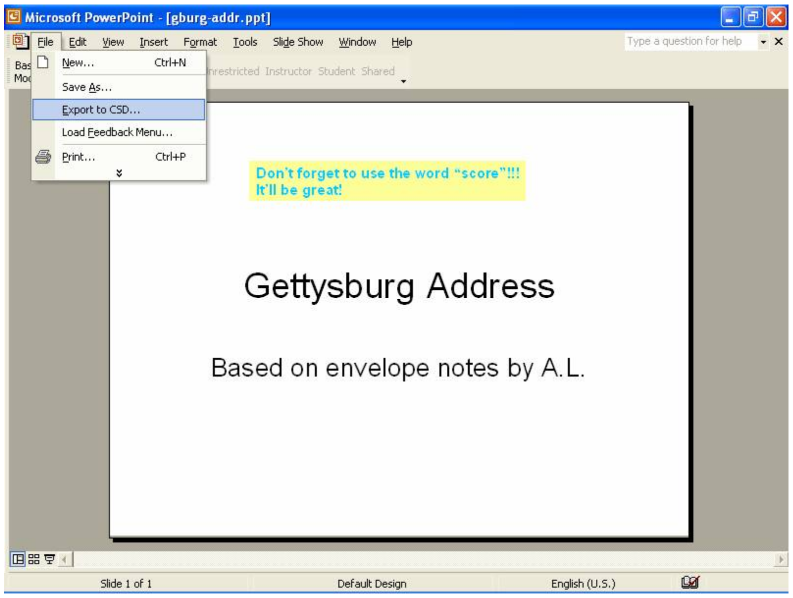

Microsoft PowerPoint - [gburg-addr.ppt]
File Edit View Insert Format Tools Slide Show Window Help
Type a question for help
Base Mode Instructor Student Shared
Unrestricted Instructor Student Shared
Slide 1 of 1
Default Design
English (U.S.)

Don't forget to use the word "score"!!!
It'll be great!

Gettysburg Address

Based on envelope notes by A.L.

Figure 3.29: PowerPoint is in instructor mode, and the textbox of Figure 3.27 has reappeared because it is visible in this mode.



Don't forget to use the word "score"!!!
It'll be great!

Gettysburg Address

Based on envelope notes by A.L.

Figure 3.30: The instructor is done editing this presentation and exports it to Presenter's slide format using the `Export to CSD` option in the `File` menu. (The `Load Feedback Menu` option configures the feedback menu for the presentation, see Chapter 4.)

modes may still require extra effort, *e.g.*, if an instructor note is spatially related to content in the base mode and that content moves during an edit in the base mode.

The second toolbar is the `Restriction` toolbar, the right-hand toolbar in Figure 3.26. An object is restricted if it is on one of the glass panes. Selecting an object displays its restriction status on the toolbar. If the object is unrestricted, only the `Unrestricted` button is highlighted. If the object is restricted to one or more modes, the buttons for those modes are highlighted. (If an object is restricted to more than one mode, it will be visible in any of those modes.) Clicking the `Unrestricted` button will make the selected shapes unrestricted. Otherwise, clicking a highlighted button will turn off restriction to that mode while clicking an unhighlighted button will turn on restriction to that mode.

Selecting multiple objects mostly works the same way as selecting a single object except that if objects' restrictions differ, the toolbar displays the most restrictive configuration.

That is, if one selected object is restricted to a mode but another is not, the mode's button will not be highlighted. Figures 3.26–3.28 demonstrate the use of the restriction toolbar.

Finally, instructors can export presentations to Presenter's deck format by clicking the `Export to CSD` button under the `File` menu, as in Figure 3.30. The resulting deck encodes the appropriate visibility information, and Presenter will automatically display the instructor pane overlaid on the base presentation to the instructor, and likewise in other roles.

3.4 Patterns of Presenter Use

Presenter has been broadly adopted across a wide range of computer science courses and, more recently, in other disciplines. Results from these many courses show that students and instructors perceive that Presenter has a positive impact on their classes. Furthermore, these many adoptions of Presenter show important patterns of use that validate, and sometimes contradict, the design decisions that led to Presenter's current implementation. This section briefly summarizes high-level statistics of Presenter usage and impact on classes and then details a set of use patterns with interesting or surprising design lessons.

Since Spring 2002, we have studied Presenter use in 37 different computer science courses and one statistics course, taught by 21 different instructors. Over 2,000 students at three different universities attended these classes.[7] Most courses used Presenter in the majority of their sessions (although most courses also included some sessions when instructors did not use Presenter). Classes where Presenter has been used include: introductory programming courses and data structure courses; advanced undergraduate courses in architecture, digital design, theory, compilers, graphics, operating systems, algorithms, languages, and software engineering; Master's level courses in compilers, databases, artificial intelli-

[7] Presenter and its source code are freely available for download. We believe that many more instructors have downloaded and used Presenter without participating in our studies. For example, we know from personal communication and published reports of a philosophy and an electrical engineering course that used Presenter at two institutions besides the three reported above. An informatics course at yet another institution based course projects on extending Presenter.

gence, human-computer interaction, programming languages, architecture, and transaction processing; and a graduate course in graphics. Six of the Masters' level courses were taught as distance learning courses. Class sizes ranged from 7 to 211 students with an average size of 57 students.

We studied system usage by observing classes, capturing sessions with a logging tool, and conducting surveys of students and instructors. In addition, we received detailed usage notes from some of the instructors. For distance learning courses using Presenter, we collected replayable, synchronized logs of video, audio, and ink using ConferenceXP WebViewer [107].

Overall, students were enthusiastic about the system's ability to create a more spontaneous and interactive classroom environment. Figure 3.31 shows the cumulative responses over all surveyed courses of student responses. Students indicated the system had a positive impact on both attention and understanding of the material. For example 57% of the respondents felt it increased their attention to class, while only 10% felt it decreased attention. Respondents were very encouraging of the system's future use.

On the whole, instructor reaction to the system was very positive; typical instructor comments included: "It works great. It didn't take any adaptation. I just talked/discussed and when I needed an example, I just wrote. As easy as using the board", and "Being able to diagram and spontaneously work examples, instead of having to use a pre-scripted PowerPoint slide deck - felt like teaching a real class".

Instructors made substantial use of the facilities of Presenter. However, their style of use varied widely, reflecting different teaching styles and different sub-disciplines. The predominate use was inking, integrated with the slide material, either writing directly on top of the slide or in marginal notes. In classes we observed, instructors used ink on most slides.

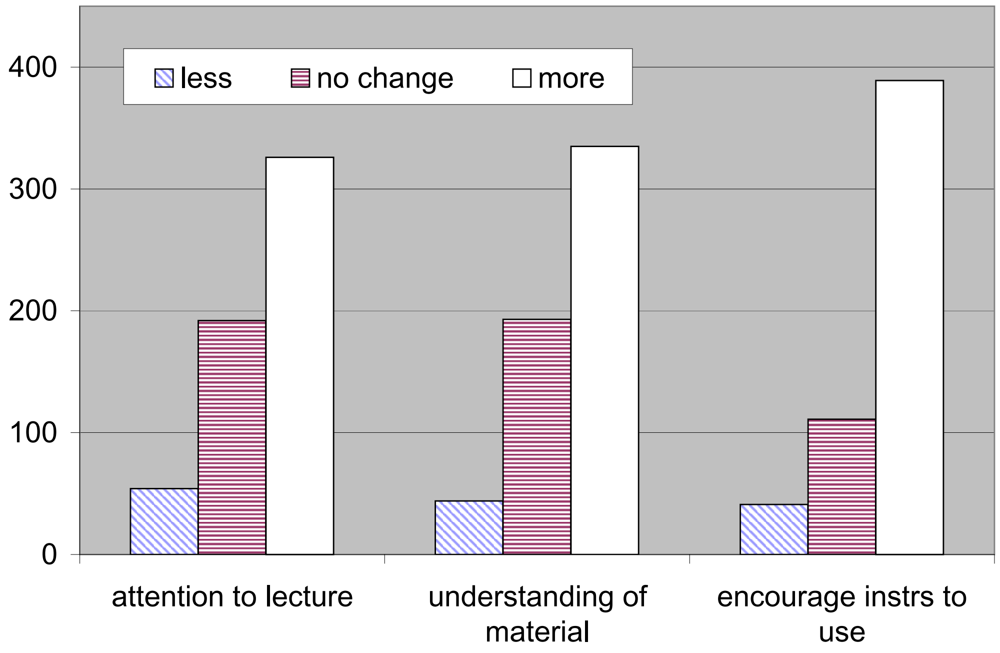


Figure 3.31: Survey results for students asked about their reactions to the use of Presenter in the classroom. ($N = 550$; students of classes taught by Presenter project group members are excluded.)

3.4.1 Examples of Use

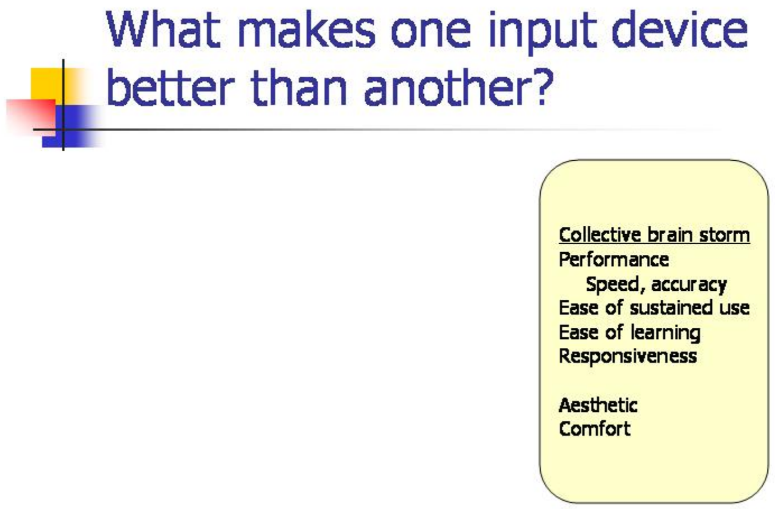
This section details patterns of system use that reveal interesting and valuable design lessons, either by supporting our initial design decisions, contradicting those design decisions, or revealing new and unexpected aspects of the design space.

Basic inking

Presenter's most basic ability, allowing instructors to write over slides, has been successful in facilitating interactions between students and instructor. Several instructors surveyed about Presenter noted that critical uses of the system included "summariz[ing] student comments" or "writing students' answers to questions I had posed". In our classroom observations we noted numerous examples where the instructor used the system to respond to student questions. Often this was done using the blank whiteboard slide or by shrinking the slide to create space in the margins for writing.

Instructor notes also supported interaction with students. For example, one instructor planned frequent brainstorming exercises in which he posed a question and solicited a list of answers from students. The instructor used ink (and scrolling mylar) to collect the list of student responses. Before class, he also generated lists of responses for each question that he wanted discussed. He used instructor notes to make those lists unobtrusively available during class, as in Figure 3.32, and prompted students to contribute any of these responses that did not come up otherwise.

Instructors also used ink extensively to connect spoken discussion to locations on the slide. This inking, which we call "attentional marks", draws viewers' attention to specific content on the slides. Examples of attentional marks include checks next to bullet points under discussion and underlining important phrases. We observed a wide range of attentional marks used for grouping, emphasis, navigation, indication of progress, and identification of key points. Use of attentional marks emphasizes the value of putting ink in the context



What makes one input device better than another?

<u>Collective brain storm</u>
Performance
Speed, accuracy
Ease of sustained use
Ease of learning
Responsiveness

Aesthetic
Comfort

Figure 3.32: A collective brainstorm exercise from a human-computer interaction course. The shaded area on the right is an instructor note listing responses the instructor wants to discuss. In class, the instructor solicited responses from students and wrote them on the slide, ensuring that the topics on the right were mentioned.

of the slides.[8]

Physical presence

An aspect of Presenter's use that manifested itself in many forms was the participation of the Tablet PC as a physical artifact in the classroom. One instructor made frequent, creative use of the physical tablet in discussions. He used the computer like a TV talk-show host's microphone: walking among the students' seats (the "audience"), waving the tablet to solicit written contributions to the public display, and passing it around to transfer control of the discussion among students. He also used the tablet's physical presence when he wanted to show that a traditionally drawn computer science tree (an ever-widening fringe of nodes depending from a root at the top) looks like a natural tree turned upside down. The instructor picked up his tablet as he explained this notion, faced the display toward the students, and then rotated it upside down at the critical moment in the discussion. Students using the Structured Interaction Presentation (SIP) System (described in Chapter 5) used the tablets as physical focal points for interaction with neighbors, talking over and gesturing toward their tablet screens. Some students using SIP also folded the tablet screen flat to the table (rather than letting it stick up at the usual laptop angle) to increase their privacy by keeping other students from seeing their screens clearly. Several instructors also use Presenter's wireless connectivity to increase their freedom of motion about the classroom, positioning themselves so they can both point to the public display and control the tablet comfortably or walking to the oft-neglected back row while retaining control of the presentation.

The tablet's physical presence also caused several problems. As with a traditional computer-projection system, instructors sometimes made deictic (pointing) gestures to the tablet rather than to the screen. Such gestures were indecipherable for students, as they could not see the instructor's screen at all. (On the other hand, for instructors of distance learning classes that were forced to stand "in a box" to stay in the capture frame of a cam-

[8] We describe attentional marking in detail elsewhere [12].

era, Presenter's inking facilities acted as a surrogate for physical deixis.) Some students also mentioned being distracted by the instructor looking at the tablet rather than the public screen, as described in Section 3.2.2 (page 31). Finally, instructors using a fixed (rather than wirelessly connected) tablet, have to pace back and forth to shift from control of the slides at the computer to interaction (*e.g.*, pointing) with the public display.

The physical form factor of the various tablets used by instructors also affected classroom use. The tablets' lightness allowed instructors using wireless connections to carry them comfortably around the classroom. On the other hand, one instructor found that she could not use the larger brands of tablet because they did not fit comfortably between her fingers and elbow. The same instructor suggested that tablets should come with a sturdy handle on the back of the screen to ease concerns about dropping the expensive computers. The slim, portable tablets were also easy to lose track of when set down flush with a desk in some corner of the classroom, *e.g.*, while the instructor talked with a student or illustrated a concept with two-handed gestures. Finally, some of the standard forms of interaction with computers and laptops were difficult or impossible with the keyboardless tablets. One instructor accidentally erased a slide and then exclaimed, in frustration "It's going to be hard for me to do control-Z [the Windows undo keystroke] on this thing!" Several instructors had trouble logging on to tablets that emit the traditional "control-alt-delete" login keystroke using a tiny or hidden physical button.

Some impacts of the physical presence of tablets seem obvious during design (such as the differences in resolution of ink capture from tablets with different sized screens), but the variety of surprising patterns (like the eye-gaze problems) and innovations (like the TV talk-show host style of use) described above emphasize the importance of understanding clearly how software designs intersect with and intrude on the physical space in which users perform their tasks.

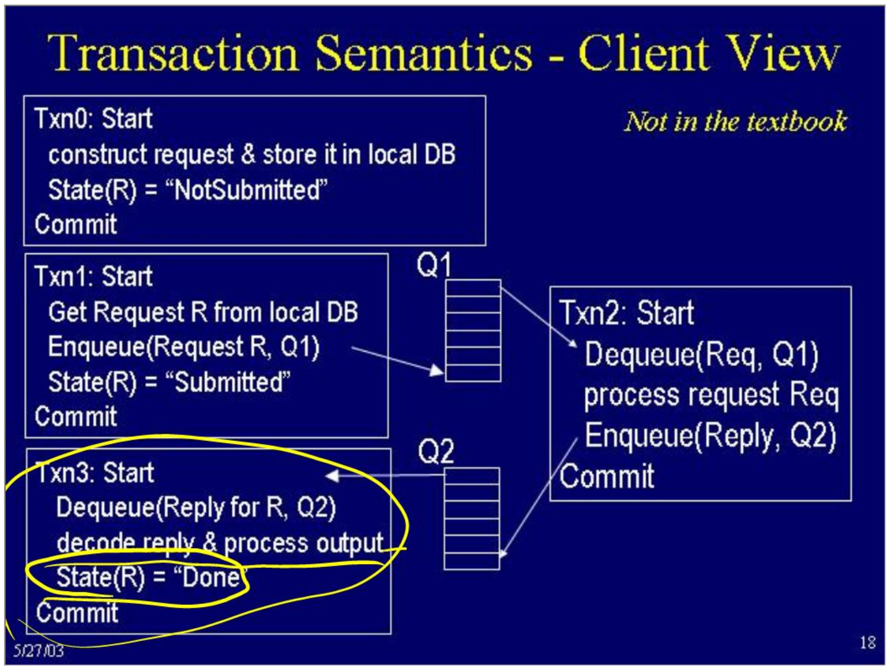
Stand-alone developer use

One common pattern of use by Presenter's *developers* was running a single, unconnected instance of Presenter and swapping among roles at runtime. For example, a developer can run Presenter, starting as an instructor; open a Presenter deck with instructor notes, initially visible; switch roles to student, at which point the instructor notes disappear; leave feedback (see Chapter 4 for more details); and return to the instructor role to view the aggregated feedback. This ability has proven remarkably useful to facilitate single-computer, single-instance debugging of what is otherwise a distributed, multi-user application. Quick swapping among roles also allows rapid, low-overhead demonstrations for and design discussions with potential users. A Presenter developer needs only a single-computer (not even, necessarily, a Tablet PC) to show most of the application's functionality. These low-overhead demonstrations are useful to seize opportunities with instructors that have never used Presenter, to support design discussions grounded in the system artifact itself with instructors already using Presenter, and to support spontaneous discussions in unusual venues (*e.g.*, on an airplane).

However, Presenter's current implementation still falls short of ideally supporting this style of use. For example, role-specific view customizations — *e.g.*, the types of feedback aggregation to display (again, see Chapter 4 for more details) — reset to default values each time the role changes. Therefore, debugging and demonstration sessions require extra steps after each role switch to reconfigure the view. Ideally, swapping roles in Presenter should replicate switching between instances/computers in different roles as closely as possible. Indeed, debugging sessions would even benefit from lightweight (in both performance and user interface action terms) simulation of Presenter's network layer.

Hygienic erase

A curious pattern of use that emerged independently across several instructors was the "hygienic erase". In this pattern, an instructor discusses a slide, marking it up with ink as



Transaction Semantics - Client View

Not in the textbook

Txn0: Start
construct request & store it in local DB
State(R) = "NotSubmitted"
Commit

Txn1: Start
Get Request R from local DB
Enqueue(Request R, Q1)
State(R) = "Submitted"
Commit

Q1

Txn2: Start
Dequeue(Req, Q1)
process request Req
Enqueue(Reply, Q2)
Commit

Q2

Txn3: Start
Dequeue(Reply for R, Q2)
decode reply & process output
State(R) = "Done"
Commit

5/27/03

18

Figure 3.33: A slide from a databases course just before the instructor erases the slide and then transitions moments later to the next slide — a "hygienic erase".

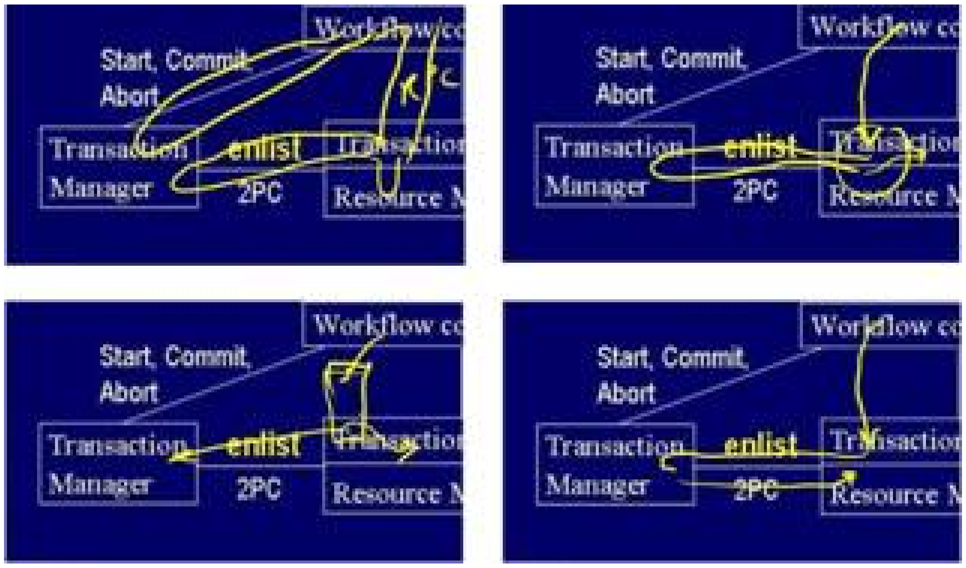


Figure 3.34: A sequence of screenshots from a databases course in which an instructor repeatedly draws on and erases the slide to illustrate successive concepts.

she talks. Just before transitioning to the next slide, the instructor hits the slide erase button. The transition and erase are often in such rapid succession that they are clearly two parts of the same larger action. Figure 3.33 shows one slide just before a hygienic erase. As with most hygienic erasures, there is no clear reason *why* the instructor erased. If the instructor was not going to return to the slide, the ink need not be erased. If the instructor was going to return to the slide, the ink may well have been useful. Even if it were unneeded, the ink would be easy to erase later. Contrast these “hygienic erasures” with purposeful use of the slide erase to discuss successive aspects of a single slide as in Figure 3.34.

We have no explanation for this behavior (besides the throw-away excuse of “hygiene” implicit in our nomenclature). Instructors have not been able to explain their erasures and often do not even notice performing them. Surprising patterns such as hygienic erase may reveal fruitful human-computer interaction research directions.

Breakdowns

Characteristic “breakdowns” in Presenter’s use are also illustrative patterns (although we often adjusted Presenter’s implementation to avoid recurrence of such problems). The “mylar scrollbar”, described in Section 3.2.1, is one feature that caused frequent breakdowns. Instructors found few valuable uses for this feature and therefore largely ignored it. Unfortunately, its placement immediately adjacent to the slide’s writing surface resulted in frequent accidental activations of the control (see Figures 3.10–3.14). Because of these breakdowns, the scrollbar is now hidden in Presenter’s default user interface configuration.

Accidental activation of the scrollbar and other controls proximate to the writing area (*e.g.*, the filmstrip) seems to be a danger of pen-based interfaces. We conjecture that, using a mouse, instructors know that their physical movements are separated by a transformation from the movement of the cursor, and so they attend to visual feedback for cursor location. Using a pen, instructors feel a direct connection to the cursor and therefore attend less to visual feedback and more to the feel of writing. As a result, they are more likely to “write off the edge” of the purely virtual windows that make up the user interface.

A variety of other features and interface options in Presenter have changed or been discarded over time because of breakdowns in use. The divider between the filmstrip and the current slide, which allows resizing the filmstrip, was widened after instructors had difficulty "catching" it with the pen. An early semantic highlighting mechanism — a quick mechanism for drawing highlights around semantically coherent chunks of text such as bullets — was discarded because of a mismatch with instructors' expectations of which semantic chunks were important. The semantic highlight also exposed instructors' difficulty in using the tablet pen's barrel switch (the equivalent of the right mouse button — both unintentionally clicking it and failing when trying to intentionally click it. Finally, deictic (pointing) gestures to the screen with the pen that either fail to touch the screen or leave only a tiny mark on the screen (and are therefore all but invisible to students) led to the development of the remote cursor option. These breakdowns demonstrate the rich set of interface problems and opportunities presented by the Tablet PC form factor. Furthermore, instructors' and students' critical roles in exposing these problems emphasizes their importance not only in co-designing innovative additions to Presenter, but in winnowing features that lacked practical use.

A surprising pattern of *dis*use was instructors' ignorance of clearly available features. Although problems with feature discovery are not uncommon, the disused features in this case were prominently displayed among only a handful of tools in Presenter's control bar (see Section 3.1 and Figure 3.6). Nonetheless, several instructors used Presenter for full terms, including dozens of hours of classroom use, without discovering features available in that control bar. One telling example is an instructor who frequently filled all the available space around his slides with cramped diagrams. In a post-term survey, the instructor wrote that if he were to use Presenter again, he would make "more whitespace for writing" on his slides. However, when asked about Presenter's slide minimize button, which generates whitespace for writing around a slide's margins, he responded "Not sure what this is". Addressing such feature discovery problems may require mildly intrusive techniques like highlighting unused but valuable features in the interface or briefly walking instructors through the interface when they first use Presenter.

3.5 Adoption Issues

Understanding how to create a viable adoption path to Presenter for instructors using traditional presentation technology is central to the Presenter research agenda. Presenter's underlying goal was to shift instructors toward a more interactive teaching style. Accomplishing this requires not just creating a technology that affords flexibility and interaction but supporting instructors' decision to use the technology.

Iterative design and deployment of Presenter revealed several key issues in creating a viable adoption path: (1) Instructors need an easy transition to a new presentation technology. (2) A reliable fallback, especially for experimental technologies, vastly increases students' and instructors' willingness and comfort to experiment with a new system. (3) Providing multiple levels of "commitment" to a system (in terms of equipment, time, and infrastructure on the one hand and functionality on the other) broadens the audience for an innovation and encourages instructors to invest their resources in progressively greater functionality.

Technologically, Presenter provides an easy transition for instructors from PowerPoint. PowerPoint is a widely used slide design and presentation delivery environment. Presenter was designed from the start to enhance PowerPoint-like slideware's support for flexible, interactive presentations while maintaining the framework of slide-based presentation. Thus, the huge base of PowerPoint users represents a broad pool of potential adopters for Presenter. As Good and Bederson did with the CounterPoint System [51], we chose to grease these users' transition from PowerPoint to our system by maintaining as much of the PowerPoint design environment as possible while managing presentation execution in a separate application. Presenter includes an add-in to PowerPoint (see Section 3.3.3) that introduces a small amount of extra functionality useful for Presenter presentations and supports simple export from PowerPoint to Presenter-readable files.

Actually building Presenter into PowerPoint would have eased instructors' transition still further. However, this choice would have caused several other problems. First, building Presenter as an add-in would have eased PowerPoint users' transition to Presenter while gravely damaging the adoption path for instructors who dislike PowerPoint. (The standalone Presenter application therefore runs over simple, image-based slide decks that can be generated easily from many different sources.) Second, PowerPoint's interoperability routines have technical restrictions that make retrofitting Presenter to execute as a PowerPoint add-in unwieldy. Third, Presenter was designed to support extensions to include student interaction. Even with the broad use of PowerPoint, adoption by a full class of students requires adoption by a large group of individuals, some of whom likely harbor animosity to PowerPoint.[9] Overall, adopting PowerPoint's design environment while distancing Presenter's execution environment from PowerPoint's created an easy adoption path for a large number of instructors while neither derailing adoption by other potential user groups nor limiting development of the execution environment.

This bifurcated structure — design in PowerPoint, execution in Presenter — has also caused some problems in adoption. One surprising complaint users have expressed with Presenter's PowerPoint integration is a desire to re-import a Presenter presentation, with its ink annotations, back into PowerPoint. This reimport would smoothly support instructors' *re*design process inside PowerPoint, *e.g.*, correcting mistakes exposed in class and smoothing bumpy transitions. Instructors also want to make new uses of PowerPoint, such as turning ink created during class into instructor notes for reference in future terms. Another adoption problem arising from our custom presentation execution environment is the lack of certain PowerPoint execution facilities in Presenter. For a research project, it was simply not reasonable to reimplement all execution features of PowerPoint, but the lack of the more heavily used features (*e.g.*, animation) has discouraged some instructors from transitioning to Presenter. This issue is especially important for instructors who are reusing old

[9]Note that Ratto *et al.* address this problem in their student interaction system by exposing a standardized World Wide Web interface [85], while the TurningPoint interaction system [105, 106] integrates with the simple, commercially available "CPS" student response system [40].

presentation material that relies on missing features. The *extensive* reuse of documents and parts of documents is one aspect of the presentation domain that designers must account for.

Another crucial issue in adoption of Presenter has been graceful degradation in the face of failures. Instructors are under tremendous time pressure in class, and students have little patience for technological failures [6]. As a result, failures of presentation technology that cause significant delay in class severely hamper the technology's adoption. Presenter again relies on PowerPoint to address this concern. In Presenter's two-computer deployment (see Section 3.3.2), the instructor controls the presentation from a wireless Tablet PC and runs the public display on another computer tethered to a data projector. With this configuration, the instructor can easily keep her presentation open *in PowerPoint* on the projection computer as a backup. If Presenter fails, the instructor goes to the projection computer, switches applications, selects the current slide, and continues the presentation, albeit without Presenter's functionality. This fallback procedure is simple and easily comprehensible for instructors. Furthermore, if the fallback fails as well, those failures are usually clearly attributable to problems outside Presenter — *e.g.*, if the instructor trips over the projection computer's power cord, she feels that the same problem would have derailed any presentation system.[10]

The alternative "solution" of debugging software failures before they reach the classroom is admirable but essentially impossible. A few of the misadventures of Presenter's most patient and accident-prone user, an early adopter otherwise uninvolved in the Presenter project, provide a cautionary tale. In the three cases described here, the instructor perceived as system failures circumstances that might be excused as minor errors, structural problems with the setup, or even mildly unusual correct operation of the tablet. In each case, the fallback procedure described above provided a safety net that kept these "problems" from becoming disasters. Before describing these cases, note that this instructor has

[10]That's assuming the instructor never recognizes the increased risk of tripping over the cord when walking around carrying a Tablet PC, of course!

(before and since) frequently used Presenter successfully across several course offerings.

In the first case, the instructor tried to load the wrong file into Presenter. Presenter gave an error message and would have allowed the instructor to load the correct file after dismissing the message. (Indeed, a researcher did just that after the class was over.) Unfortunately, rather than a user-friendly error message, Presenter simply spewed out a text version of the underlying exception generated during the problematic load. The instructor took one glance at this error message and set the tablet aside. He had neither the time nor the inclination to debug errors during class! Nonetheless, the fallback option preserved the flow of class.

In the second case, the instructor lost access to the presentation file he needed. Before each class, the instructor generally made a few last-minute updates to the day's presentation file, exported the source to Presenter's file format, and synchronized a network folder containing all this data onto his laptop (the projection computer). The PowerPoint source and Presenter file for the presentation was therefore readily available on the laptop's local drive. He also habitually waited to download the files to his tablet computer (the instructor's computer) until he reached the class. On one day, the class's network connection was down and his network folder with the Presenter slides was therefore unavailable. In the rush to start class, he was unable to transfer the Presenter slides from the projection computer to the tablet. After a couple of minutes of effort, he gave up and, again, the fallback option preserved the flow of class. Some responsibility for this incident surely falls on Presenter's deck building and slide distribution procedures, yet calling this a Presenter bug would be a stretch.

In the final case, the instructor set Presenter up successfully and then set the tablet aside while waiting for class to start. He then handled some administrative issues without slides. When he finally retrieved his tablet, he found the screen had gone completely blank. Nervous about the state of the computer, he waved the pen at it, but the screen remained blank. Once again, he set the tablet aside and would have switched to the fallback. However, a researcher observing the class took the tablet and waved the pen in range of the screen's digitizer, disabling the screensaver and returning the tablet to normal operation.

Presenter *could* have been designed to handle all of these cases, even the final one. Indeed, since the first two cases occurred, Presenter has been modified to at least partially address both of them. (Loading an incompatible file still causes an error, but the error message no longer includes gobbledygook. Slide transmission between computers within Presenter has been improved and is an area of current work.) However, the real lesson of these cases is that the advertised availability of a simple fallback for a new technology helps defuse unexpected problems and keep them from damaging the adoption process for that technology. Indeed, the ability to print backup mylar transparencies from PowerPoint has been a crucial feature in some instructors' adoption of PowerPoint itself!

Finally, Presenter supports multiple levels of commitment (*e.g.*, of time and equipment) through a diverse set of supported configurations. Section 3.3.2 details these configurations and their requirements. Instructors have started using Presenter with everything from the single-computer, single-display configuration requiring only an isolated Tablet PC to the infrastructure-based, wireless, two-computer configuration. Some instructors have also begun using Presenter with one of these configurations and decided to adopt it only when they discovered they could switch to a configuration with more functionality (*e.g.*, mobility) or less commitment (*e.g.*, lack of network connections). One instructor even adopted Presenter on a non-pen-based laptop. On the other hand, Presenter's stress on PowerPoint as a design environment has also discouraged adoption by some potential users. Users who do not own PowerPoint, do not want to use PowerPoint, own an older version of PowerPoint (before 2000), or already have source from other presentation software have been loath to commit themselves to the transition to PowerPoint. Theoretically, building Presenter files from other presentation sources is feasible. Unfortunately, in practice, we have not had the necessary resources to create build environments for the small number of interested users of each other presentation system.

These strategies for securing adoption in Presenter align with Rogers's established principles for adoption of innovations [86]. Easing the transition from PowerPoint to Presenter corresponds to the principle of compatibility: “the degree to which an innovation is perceived as consistent with the existing values, past experiences, and needs of potential adopters” [86, page 250]. Furthermore, tying Presenter closely to an existing, familiar technology supports instructors’ comparison of Presenter to the previous technology, corresponding to the principles of observability, “the degree to which the results of an innovation are visible to others” [86, page 251], and relative advantage, “the degree to which an innovation is perceived as being better than the idea it supersedes” [86, page 250]. Providing a reliable fallback and multiple levels of commitment both correspond with the principle of trialability, “the degree to which an innovation may be experimented with on a limited basis” [86, page 251]. Finally, all three of these strategies seek to reduce instructors’ perception of the complexity of the technology, “the degree to which an innovation is perceived as relatively difficult to understand and to use” [86, page 250]. In terms of Rogers’s principles, the key lesson in securing adoption for Presenter is to attend not just to standard computer science research and development concerns (like “relative advantage” and “complexity”) but also to the critical roles of trialability and compatibility.

3.6 Contributions

The key contributions laid out in this chapter were the Presenter System itself, a widely adopted, flexible presentation system that integrates ink in the context of slides; the set of design principles underpinning Presenter, including the three key principles of high-quality ink in slide context, separation of views, and distributed architecture; and insights and experiences that inform these principles. Presenter is also the foundation for the systems described in Chapters 4 and 5. Both systems embellish Presenter’s standard presentation features with new functionality for student interaction, and both exploit the design choices made in Presenter, *e.g.*, making extensive use of separated views to present new control and information displays to students and instructors.

Chapter 4

THE CLASSROOM FEEDBACK SYSTEM: LIGHTWEIGHT, STUDENT-INITIATED, CONTEXTUAL FEEDBACK

Student-instructor interaction is vital to student learning. However, soliciting student feedback in large, university-level lecture classes (with about 50 students or more) is challenging; as a result, lectures tend to lack interaction [15]. Yet, as educational institutions serve more students and face tighter resource constraints, the large lecture is likely to persist, especially at the introductory level, creating a need for innovative approaches to large class challenges.

We have designed a feedback system to address this problem. We identified challenges to interaction in large classes by reviewing existing literature and observing courses. Based on these challenges, we designed the Classroom Feedback System (CFS), a computer-mediated feedback system to promote student-initiated interaction (Figure 4.1). CFS is implemented as an extension of Classroom Presenter (see Chapter 3). Students view lecture slides on the classroom display and on their wireless devices, and the instructor controls the presentation from her own device. Students can annotate any point on a slide with feedback from a configurable list (*e.g.*, MORE EXPLANATION, EXAMPLE, and GOT IT). CFS anonymizes feedback before displaying it to the instructor within slide context. CFS differs from previous audience-initiated feedback systems in that it provides rich context: merging comments with slide context to aid the students in crafting meaningful feedback and the instructor in interpreting that feedback. CFS is novel in empowering students to provide simple yet descriptive feedback on their own initiative.

Following design experiment methodology [24], we began by studying our target set-

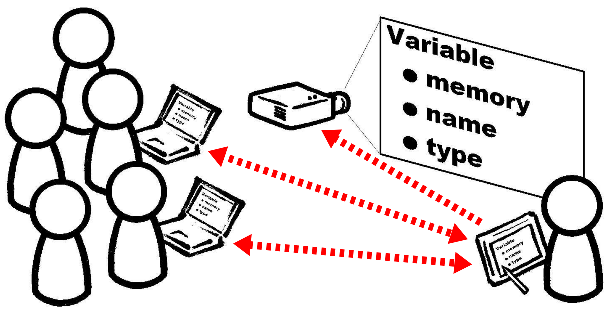


Figure 4.1: Overview of system setup, consisting of the instructor's device, students' devices, and a projector for the classroom display. Students use their devices to place feedback that is summarized on the instructor's device. Dotted arrows represent wireless transfer of the presentation and feedback.

ting, large university-level classes, through observations of two distance learning courses and one large lecture course. Next, we experimented with a paper prototype of the feedback system; students annotated paper slides with free text and category comments. These initial observations and experiments led to an understanding of the domain and design principles for our system. We performed a pilot study with a pen-and-paper and an electronic prototype of the system in a large class. Our pilot study confirmed that students have feedback to give, they are willing to provide it through an electronic medium, and an instructor can respond effectively to this feedback, changing her pace and explaining confusing concepts. Finally, we engineered a learning environment using the full-featured CFS in an introductory programming course. This chapter focuses on how the system addressed interaction challenges in this most recent experiment, but uses evidence and design experiences from prior studies throughout.

This chapter's structure reflects its principal contributions: a set of challenges associated with interaction in large classes in Section 4.1, a set of design principles based on primary challenges to interaction in large lecture courses in Section 4.2, design of a feedback system (described in Section 4.3) to address these challenges, and finally, experimental results demonstrating CFS's success in promoting interaction in a large class and exploring its relationship to the challenges in Section 4.4. Our results show that students did indeed take advantage of the newly available feedback channel, and the instructor responded to the feedback, adapting the presentation accordingly and creating opportunities for continued interaction. Together, these contributions establish the value of a computer-mediated, student-initiated feedback system for large classes and set the stage for future work exploring enhancements to the design and long-term evaluations of its use.

4.1 Challenges to Interaction in Large Classes

The education community has long discussed the challenges of facilitating student-instructor interaction in large classes [48, 50]. One study of 51 classes with up to fifty students (well below the enrollment of many colleges' larger classes) found that student participation drops with increasing class size [43]. This work focuses on the lack of student-initiated interaction in large lectures. This lack poses a serious threat to the learning environment because interaction is important to student learning [69]. Based on literature and experiments with prototypes of CFS, we have identified several primary factors inhibiting student-initiated interaction in large classes: feedback lag, student apprehension, comment verbalization, and the single-speaker paradigm.

Feedback Lag: Timing is a critical factor affecting student participation. A fast-paced lecture can leave inadequate time for the student to process presented material [15], let alone interrupt with a question. Discussion with students participating in our pilot study pointed to a phenomenon we call "feedback lag." Students in the study were unsure of the value of their questions on a topic until the topic was finished. Then, when the lecture moved on, they believed the "window of opportunity" to ask their questions had passed. As one student in our pilot study wrote, "the lecture was proceeding so quickly that it didn't seem very relevant to ask a question. . . "

Student Apprehension: Students are intimidated by the number of students in large classes [17]. They will remain silent rather than interrupt class to ask a question they perceive as stupid or unimportant. Such feelings of apprehension are very common among college students, particularly in larger classes [17, 77]. In our electronic pilot study, 6 of the 12 participants reported apprehension as a factor limiting their participation, *e.g.*: “Didn’t want to sound incompetent” and “Just a little shy”.

Comment Verbalization: A factor closely related to student apprehension is student ability to verbalize comments. Instructors of large classes in our department have often observed that students have trouble communicating their confusion in words. Considering student apprehension, students who are unable to verbalize a comment are unlikely to interrupt lecture simply to announce their confusion.

Single-speaker Paradigm: Large classes can sustain only limited time for spoken feedback. Spoken feedback is hindered by the nature of the voice channel in the classroom [50]. When one student speaks to the instructor to ask a question or make a comment, no one else in the class can speak, and the instructor’s attention focuses on that one student. This paradigm poses no problem in a class of ten, but if each student in a class of two hundred asked even one question requiring ten seconds to articulate, over a half hour would be consumed just listening to the questions. In our pilot study, 3 of 12 participants reported class size as a factor limiting their participation, *e.g.*: “too many people”.

We believe that these challenges to participation represent the core obstacles to interaction in the large lecture classroom. Effective approaches to addressing these problems must also consider two important design challenges. First, solutions should require no more than marginal attention of both instructors and students, because both parties need to focus attention on the class itself. Second, given the size of the large class, solutions must plan for managing increased volumes of feedback to ensure its usefulness to the instructor.

4.2 Design Principles

We present a set of design principles for computer-mediated feedback systems targeted at large classes. These principles follow directly from the interaction and design challenges discussed in Section 4.1. Our goal is not to outline the only solution in the space of computer-mediated feedback systems. Instead, these principles describe one route to leveraging the affordances of networked computing to address the challenges and thus promote interaction. Along with these principles, more general lessons about collaborative software also apply such as attempting to deliver value to all the stakeholders that put effort into the system [52].

4.2.1 Non-Verbal Communication

Networked computers provide an alternative to speech, sidestepping the single-speaker problem. Any number of students can provide feedback simultaneously through individual computers. If these computers share a network, a computer system can silently and unobtrusively collect the individual student feedback for the instructor. At no point in this process is exclusive control of a communication channel necessary.

4.2.2 Anonymity

We believe that anonymity can address the challenge of student apprehension. Indeed, a related experiment with anonymous, in-class, written student feedback in one of our study classes found participation levels of over 60% out of 150 enrolled students [92]. In our experiments, written student feedback that was anonymous included classes of comments we had never heard spoken during our observations — *e.g.*, one student complained of boredom and several students complimented the instructor's teaching.

Anonymizing student feedback in a computer-mediated system is simple and rapid. Digital feedback is easily separated from identifying information, and computer systems avoid pitfalls of written feedback such as recognizable handwriting.

Establishing student confidence in this anonymity may present more of a challenge. A student giving feedback on a paper form with no name attached clearly understands the protocol protecting her identity and the risks associated with that protocol. To build similar understanding of a computer-mediated system, students should be encouraged to explore the instructor interface to the feedback system.

4.2.3 Shared Context for Feedback

Students and instructors already view a shared artifact in the classrooms we are targeting: the projected display. This artifact can be used as a shared context for feedback by attaching student's to text or locations from the shared view. This mechanism addresses several challenges to interaction while introducing little extra burden on students and instructors.

The presence of context scaffolds students' verbalization by giving them concrete foci for their feedback. Moreover, because the students are already attending to this context in the class, it does not burden them with new material to absorb. Indeed, many students already "annotate the shared display" by taking their notes on pre-printed copies of the slides used in their classes.

The use of context can also circumvent feedback lag if the context of students lagged questions remains available. As long as a student can still see the context that inspired a comment and the instructor can also see the context, the student's comment in that context can still spark valuable interaction. Therefore, a feedback system should keep context available long enough for students to express their comments and instructors to observe those comments.

The instructor also benefits from shared context. Like the student, the instructor is already attending to projected information, making feedback in the shared context easier to understand. We expect that shared context will make understanding student feedback a simple task for instructors in many cases.[1]

[1]Note that recovering the context of feedback anchored only by the time of its occurrence in the lecture (as in some previous work [22, 85]) might be complicated by the feedback lag effect.

4.2.4 Rapid, Automatic Synthesis of Feedback

The instructor must be able to synthesize and respond to the feedback students provide. Ignoring student difficulties articulating their questions, this synthesis does not generally pose a problem for spoken feedback. However, once a class uses simultaneous feedback on the scale of dozens or hundreds of students, synthesis of feedback can become a challenge. Computer-based aggregation should offer real-time synthesis of feedback to the instructor.

This challenge is not limited to computer-based feedback; any feedback system that brings together substantial, simultaneous student input faces the same difficulties. In our related work with Classroom Assessment Techniques [13], summarizing and understanding the 93 to 133 approximately one-paragraph student responses took between three and five person-hours. Obviously, this cycle time puts CATs well outside the realm of real-time feedback tools. Other instructors have experimented with visual aggregation, *e.g.*, by having students display cards to signal some concept or problem [15, 55]. However, these paper-based techniques threaten students' anonymity and lack the shared context described above. (Chapter 5 presents an extension of Classroom Presenter that addresses this issue for structured classroom activities like CATs.)

4.2.5 Simple Interfaces

Both students and instructors are already engaged in activities that demand their attention in class. To account for the limited attention available for new tasks, both student and instructor interfaces to the system should be simple.

4.2.6 Closure

While soliciting student input is a crucial first step to interaction in the classroom, it is also critical for the instructor to receive feedback about how well his or her response addresses the feedback that instigated it. Without a mechanism for students to augment or modify their feedback, this closure cannot happen through the feedback system: the student gives feedback, the instructor responds, and the exchange ends. The instructor in the electronic pilot study specifically requested this ability for students to modify existing feedback, saying: “I felt like I responded [to the feedback]. . . [but] I wasn’t sure. . . was it enough of a response because there wasn’t a way to tell me that ‘Ah yes, that answers the question’. . . ” In order to provide closure on feedback, we set as a design principle that students should be able to augment or modify feedback at least enough to indicate that the instructor has responded adequately to an episode of feedback.

4.3 Designed System

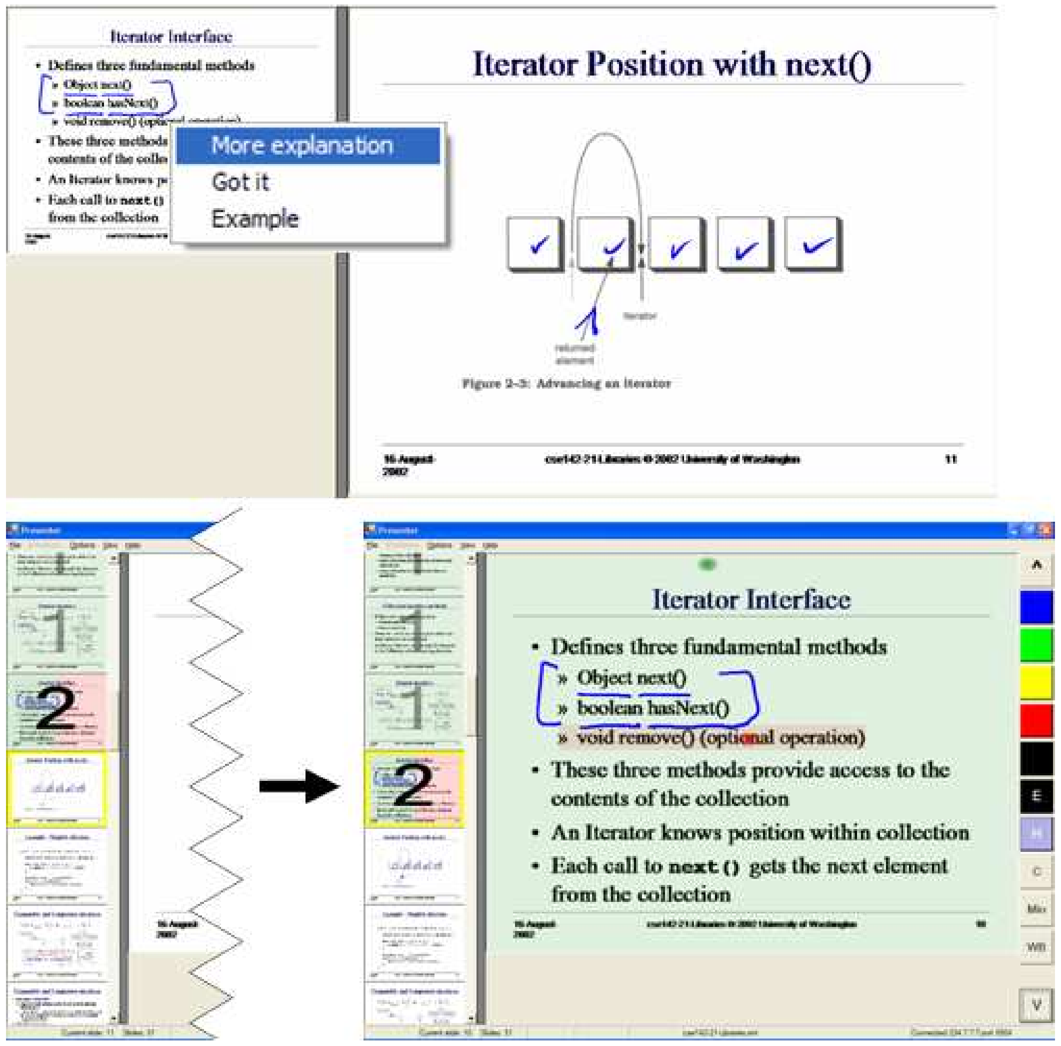
In response to the challenges faced by large lecture classes, we designed a technological system to promote interaction called the Classroom Feedback System (CFS). The increasing presence of technology in the classroom offers affordances that make this system possible: many college classrooms have data projectors and computers to drive them; high bandwidth wireless networking is becoming more common in classrooms; and many instructors use prepared slides to mediate their presentations. We assumed these affordances in designing CFS and leveraged them to address the challenges from Section 4.1. Our central design principles, described in Section 4.2, establish the connection among challenges, affordances, and implementation.

CFS is built atop Classroom Presenter (see Chapter 3), presentation software that allows the instructor to navigate through and write on a slide-based presentation from a Tablet PC. CFS is composed of a student view (on student devices), instructor view (on the instructor’s private device), and shared view (slides displayed to the entire class as in “normal” presentations). Students use their view to annotate regions on the slide from a fixed list of possible feedback. Each student’s display shows only the feedback she placed. (See Figure 4.2(a).) The instructor view is the instructor’s private interface to the system, showing anonymized aggregations of student feedback. (See Figure 4.2(c).) The shared view is projected to the entire class, and is just the “normal” slide view. This arrangement makes students' feedback both anonymous and private — *i.e.*, only the instructor and the student who left a piece of feedback see that feedback.

Students can use any networked computer capable of running Classroom Presenter for CFS. Each student view (see Figure 4.2(a)) displays the current and previously visited slide with the student's feedback superimposed. Both slides are displayed on students' devices to expand the available slide context and address feedback lag. Students generate feedback from a fixed list of possible annotations. When the student right-clicks anywhere on the slide, a menu of annotation categories appears. The student selects one of these, and the selected category's name appears as colored text at the location clicked. Each category is associated with a distinct color. The feedback is also sent to the instructor's device. If a student decides his feedback has been addressed, a click on the feedback removes it. Using a small, fixed menu of categories keeps the student interface simple. Supporting annotation directly on the slide leverages shared context.

The instructor controls the presentation from the instructor view (see Figure 4.2(c)). This view shows aggregated feedback through two mechanisms: (1) each student's annotation is represented by a colored translucent circle at a position corresponding to the one where the student left her feedback and (2) all annotations on a slide region are further aggregated into a histogram overlay — a highlight on the region with intensity proportional to the number of annotations and colors reflecting the types of annotations. The circles maintain exact context information while the highlights make mass feedback prominent and eye-catching, alerting the instructor without requiring constant attention. The instructor's filmstrip view of the slide deck (on the left in the figure) summarizes feedback on other slides with a single histogram overlay for the entire slide. Several other visualizations are also available, all built atop Classroom Presenter's layered slide rendering architecture (see Section 3.2.4).

An episode from the last day of our main study illustrates CFS's use. A student raised her hand to ask a question but was not seen by the instructor. After about a minute, the instructor transitioned to the next slide, and the student abandoned asking the question aloud.



Iterator Position with next()

16 August 2002 cse142-21 Libraries © 2002 University of Washington 11

Iterator Interface

- Defines three fundamental methods
  - » Object next()
  - » boolean hasNext()
  - » void remove() (optional operation)
- These three methods provide access to the contents of the collection
- An Iterator knows position within collection
- Each call to `next()` gets the next element from the collection

16 August 2002 cse142-21 Libraries © 2002 University of Washington 10

Figure 4.2: Views of the student (top) and instructor (bottom) interfaces to CFS, including annotation data from the study course. In the top image, the student is posting a MORE EXPLANATION annotation on the last slide visited. The lower left image is cropped from the left side of the instructor's view: the student's feedback has just combined with existing feedback to yield a count of two annotations (with color encoding the categories) on the thumbnail of the previous slide. In the lower right, the instructor has transitioned back to the slide with the feedback. The new feedback is a small circle on the word "optional". Another student's feedback (color-coded as GOT IT) is at the top of the slide.

Instead, the student posted MORE EXPLANATION on the previous slide (Figure 4.2(a)). The instructor looked at his screen about four seconds later — our logs record his interaction with CFS at this time — and probably noticed the annotation then. Figure 4.2(b) shows how the feedback was reflected in the filmstrip view. The instructor continued his discussion of the current topic for 40 seconds, perhaps waiting for a comfortable breaking point. He then transitioned back to the annotated slide (Figure 4.2(c)) and responded to the student's feedback. Although the student did not then remove her feedback, she discussed this episode in a survey response: "I think I asked for more explanation of a given word and he explained well. It was even on a slide that he had already passed by."

A key parameter to the system is the choice of feedback categories. We suggest that they be important to the course, be few in number, share thematic similarities (as a mnemonic), and include at least one option for positive feedback. The last of these suggestions was inspired by students' fervent desire for and extensive use of a positive feedback category in our pilot study. In our experiments, instructors have chosen categories that represent student commentary on the slides themselves and categories that give suggestions to the instructor.

We envision students using CFS to express their confusion or interest. The instructor monitors her view peripherally and chooses when to respond to feedback she considers important. She might also set time aside for feedback on some slides. Other uses include improving slides from term to term based on archived feedback, designing feedback opportunities as part of classroom assessment activities, using feedback to support voting, and refining upcoming lectures based on feedback in current ones. These scenarios form a vital part of the system design by encouraging instructor buy-in to the system. Indeed, the instructor in our main study initially saw CFS's archival use as its primary value.

4.4 In-Class Feedback Experiment

We ran an experiment with the prototype system in a large university-level introductory programming class in order to understand how a real class would use and adapt to CFS. During the experiment, we focused on understanding what interaction occurred and how it changed in response to CFS. Were the challenges that motivated us present in the class before introducing the feedback system? Did interaction increase with the introduction of CFS? How did the instructor and students respond to the system? How did use of CFS affect the challenges to interaction? The experiment provided interesting answers to these questions and revealed surprising new uses and impacts of CFS. Overall, students did provide substantially more input with CFS than without, addressing impediments to spoken participation and changing the flow of the class; however, both students and instructor were dissatisfied with aspects of the system and its use.

Section 4.4.1 establishes the setting of the experiment: describing the class, instructor, and students in the study; and detailing the data collected in the study. The following sections describe analysis of the study data. Section 4.4.2 describes CFS's effect on interaction. Section 4.4.3 describe how the students and instructor responded to CFS. Section 4.4.4 describes how CFS affected the challenges described in Section 4.1.

4.4.1 Experiment Setting

The experiment took place in a CS1 (introductory programming) course. The course started with an approximate enrollment of 150 and ended with 120. Typical lecture attendance ranged from 90 to 115. Students attended three 50-minute lectures weekly during a nine-week academic term. 50 minute lab sections of about 20 students led by a graduate teaching assistant met once weekly. Because the course has no prerequisites, the students varied widely in experience with computing.

Lecture usually proceeded as follows: the instructor begins with administrative announcements pertaining to upcoming exams or homework assignments, then asks for and fields questions, and finally begins his prepared lecture. Even before using CFS and Classroom Presenter, the instructor used computer-projected materials exclusively (slides, program demonstrations, course web pages). He switched between these media an average of 7 times per lecture. He generally adhered to a sequential presentation of his slides, rarely returning to a previously visited slide, and was almost always able to cover all slides he prepared for the lecture (20 on average). As the instructor put it: "I wrote the slides, I'm supposed to know what they are, and they're supposed to come in some kind of logical order, … it's not like I'm trying to look at them and decide which one to go to. I've already decided that."

Spoken student participation was limited as shown in Table 4.1. However, whenever the instructor saw a student raise his or her hand for a question the instructor tried to answer it. Overall student perception of the class was consistent with previous offerings, with course evaluations slightly higher than for the same course in previous quarters.

In order to compare interaction behavior with and without the system, the class used PowerPoint for the first three weeks of the term, Classroom Presenter for the next three, and CFS for the last three. CFS was used for a total of seven class meetings spanning the last three weeks of the term. 12 student volunteers participated in the study. To compensate them for their time, each received a $10 gift certificate. We supplied each student with a wirelessly networked laptop computer at each class session. On average, 8 of the 12 students actually picked up their laptops each lecture. (Attendance in the class was spotty as in many large classes.)

The research team met with students prior to their use of CFS to familiarize them with the technology and to assign each a unique ID number. We also spoke with them as they collected and returned their laptops. We met with the instructor (who was not a member of the research team) regularly to get his opinions about the class and CFS.

One of the tasks in an early meeting with the instructor and students was to negotiate the feedback categories to use in CFS. We envision that the task of negotiating feedback categories with the students would normally fall to the instructor. However, the research team scaffolded category negotiation in this experiment to reduce the burden on the instructor and better understand how students and instructors viewed the categories. In this experiment, the instructor settled on three feedback categories: the MORE EXPLANATION category suggests that the instructor elaborate on a topic; the EXAMPLE category requests an illustrative example of a topic; and, the GOT IT category indicates that the student understands a topic. The topic referred to depends on the placement of the feedback on the slide.

Data

We collected a variety of qualitative and quantitative data in an effort to “triangulate” interesting phenomena. The data sources are explained below with the names by which we refer to them italicized. We collected *meeting notes* during all meetings with the student participants and the instructor. Two researchers recorded their handwritten *observations* (137 pages total) at all class meetings — both before and during CFS use — from five minutes before until ten minutes after class, identifying students by ID where possible. We collected all class handouts. CFS recorded *logs* of all navigation, writing, and feedback annotation and removal (both tagged by ID). 11 of the 12 volunteers chose to complete a *survey* with responses identified by ID (see the survey text in Appendix Section B.1). We put a brief, *class-wide survey* on the course evaluations (42 completed it). The class-wide survey included only aggregate results: no CFS IDs and no correlation of responses across questions. We collected publicly available *course evaluation* data. At the end of the term, we interviewed the instructor and transcribed the audio recording of the *interview* (2000 lines of typewritten data).

4.4.2 Effect on Interaction

CFS was successful in promoting interaction. Students provided a large quantity of feedback through CFS. As Table 4.1 shows, there was a statistically significant and substantial increase in student input with the system. Discounting GOT ITs (which rarely initiated interactions), the change is not statistically significant but still suggestive of increased interaction, considering that only one in ten students in class used the system while all could contribute aloud.

The instructor felt the student feedback — with the exception of GOT ITs merited response in class. He felt that ignoring a feedback comment was as egregious as ignoring a spoken question. In response to the suggestion of skipping such comments, he said: “No, … if they raised their hand, I would answer the question.” 7 of 11 students in the survey believed the instructor responded to almost all of their feedback.

At the same time, CFS did not seem to hinder traditional interactions. As shown in Table 4.1, the average number of student voicings per class before CFS and during its use were consistent with each other. In the class-wide survey (which included the whole class, not just the dozen CFS users), most students said the feedback system did not effect their classroom experience, suggesting that it did not change their participation in class.

Our data suggests that broader adoption of CFS would not disrupt traditional participation. We did observe students with laptops participating aloud in class. All 11 students in the survey felt that they were no more or less comfortable participating aloud while using CFS, and 10 of the 11 felt that their level of spoken participation was unchanged by CFS.[2]

4.4.3 Responses to CFS

Both students’ and the instructor’s opinion of CFS was mixed. Students generally liked the system but were split on whether they would want to use it again. The instructor did feel that CFS encouraged communication and supplemented spoken questions, but was frustrated by difficulties in interpreting feedback and a fear that the feedback would derail his presentation.

[2]The student who reported a reduced level of spoken participation reported more than three spoken interactions per class before CFS and none with it. That prior level of input was probably erroneous as it would account for all voicings in the class.

Table 4.1: Comparison of total student input per class before CFS (15 classes) and during its use (7 classes). "Spoken" indicates spoken student participation. "All" further adds CFS feedback annotations (discounting mistaken annotations and lumping together annotations expressing a single idea). "All except GOT IT" discounts GOT IT annotations. (*****: significant at $p < .1$, using heteroscedastic, two-tailed t-tests.)

	Spoken pre-CFS	Spoken	All	All except GOT IT
# per class	2.4	2.6	**15.9***	7.9
p-value	——	.91	**.04***	.14

Student response to CFS

8 of 11 students in the survey reported that they enjoyed using CFS, but only 6 of these reported that they would want to use CFS again. All 3 students who did not enjoy using the system and one who did cited the form factor of the laptops as a primary obstacle (*e.g.*, "It was cumbersome, it was very hard to use it and write any notes."). One student also objected to anonymous communication on principle. 5 students reported suffering technical difficulties. (According to our observations, all students probably did suffer at least some technical difficulties due to instabilities in the prototype implementation.) However, most of the students who reported technical difficulties also reported that they would want to use the system again.

Class-wide opinion of CFS was dominated by a surprising lack of awareness of its effects. 42 students responded to class-wide student evaluations regarding CFS. Of these, one student reported that CFS had an overall negative effect on classroom experience while 7 reported an overall positive effect. 15 students felt that responses to CFS sometimes helped them follow lecture while only 5 felt that responses sometimes hindered them. More telling is the fact that about half of the respondents were unaware of how the instructor's response to feedback affected them. CFS apparently did not impact the classroom experience so obviously that students uninvolved with the experiment tended to notice it.

Instructor response to CFS

The instructor's response was also mixed. Overall, he characterized the value of feedback through CFS this way: "It was less helpful than if [the students] had raised their hands. If they weren't going to raise their hands at all then I guess it was more helpful." His dissatisfaction with CFS centered on his perception that it caused "inexplicable jumps" in lecture: changes to the flow of lecture that were incomprehensible to all the students except the one that left the feedback. (Recall that only the instructor and the student who gave a piece of feedback could see that feedback.)

The instructor felt that these shifts in the lecture lose students, decrease their engagement in the class, and frustrate them. Indeed, he characterized his state while engaging in an "inexplicable jump" in response to one set of feedback as "apparently having had some sort of brain seizure." He perceived this problem to be severe enough that it would keep him from adopting the system: ". . . I would hesitate to use the system as it currently is because of [the inexplicable jumps]."

In practice, our data suggests that these shifts were much less traumatic for the students than they were for the instructor. We might expect students disturbed by these jumps to report problems with CFS in the class evaluation or survey. However, as mentioned earlier, only one student reported any negative effects of CFS in the evaluation, and no students reported feedback derailing the class in the survey. The observers also found the impact of these shifts much less pronounced than the instructor. During the particular episode that the instructor characterized as "like a brain seizure," neither observer in the class could see feedback given by students (just like most students in the class). Despite specifically watching the instructor for evidence of interactions and the effects of CFS on the class, neither observer was aware that the instructor was responding to feedback. Furthermore, in the class-wide survey most students reported not even noticing any effect of CFS on the classroom. However, even if students were unaware of the instructor's discomfort, that discomfort remained a substantial impediment to the success of CFS.

Despite the jumps, the instructor still felt that CFS had a positive impact. He pointed out that the increased student control of the class was a benefit: "... there's a real feeling that the students are participating in the presentation of the [lecture] with this stuff. ... they're helping drive it, and that's great." Indeed, he repeatedly mentioned that even the jumps might be acceptable if he could expose the feedback causing the jump (while retaining anonymity) to the class as a whole, and he found a way to work around this problem using Presenter ink. At least 11 times, the instructor circled, underlined, or otherwise drew attention to the piece of the slide he felt was relevant to a feedback annotation. Not only did this reinforce his discussion of the topic, but it also served to communicate to the student who gave feedback that he was responding. As he said: "... [by circling words on the slide] I could show the person that asked I'm trying to answer your question." Furthermore, the instructor cited the EXAMPLE annotation as the one category that was consistently comprehensible and to which it was easy to respond. Finally, he was comfortable with occasions when he received feedback on the current slide, responded to it, and saw that feedback cleared before moving on to the next slide, saying: "... if I could see that the screen had gone clean [of feedback annotations] again, I think that makes sense."

4.4.4 Addressing Challenges to Interaction

The data from the use of CFS suggests that some interaction challenges were addressed by use of the system while others interacted in surprising ways with system design choices and the classroom climate. Overall, satisfaction with CFS correlated strongly with students' perception of difficulties interacting in class. 8 of 11 students surveyed reported that factors in class inhibited them from participating aloud. The same 8 students reported that they enjoyed using the system while the other 3 reported that they did not. All 6 students who said they would want to use CFS again were also among the 8 who reported factors inhibiting their participation. In the remainder of this section, we discuss each challenge and how its manifestation in class changed with CFS.

Feedback lag

Feedback lag was a prominent problem before CFS. This challenge manifested itself as a lack of opportunities for students to interject their comments and questions. The instructor explicitly asked for questions only 16 times over 22 observed lectures, and 11 of these instances occurred at the start of lecture when the instructor was discussing administrative issues. Of the 16 requests for questions, the instructor received responses 7 times (out of 48 student comments and questions over all observed lectures). Students also had a sense that *un*solicited questions would interrupt the flow of lecture. 3 of 11 students surveyed reported an aversion to disrupting the flow, *e.g.*, one student cited: “Difficulty finding an opening or break to ask questions. Not wanting to slow class down.”

The instructor, for his part, said that he tried to be open to questions that resulted in elaborating points he was discussing. However, he was leery of breaking up the flow of the class, saying: “I write the slides with the idea that there’s a narrative, and it’s really painful to keep interrupting that and go back.” He felt that shifts in the lecture could lose students and therefore favored questions that fit into the topic of the moment. As he said: “. . . I’ve got a bunch of stuff I’m going to say about a slide and, and it’s got a trajectory and I say it and if there’s confusion about it then. . . [t]his is the time to talk about it. . . ”

CFS proved valuable to students in addressing feedback lag. Of the 3 students who cited feedback lag as an impediment to spoken participation, only one mentioned it as an impediment to feedback through CFS. CFS seems to have helped address feedback lag for that one student as well. As he put it: “With the system in place, it became much easier to ask the professor to explain a topic more or give an example while minimizing the interruption to the class.”

Two distinct strategies for overcoming feedback lag with CFS emerged. Two students reported using the strategy we had expected: waiting until the instructor finished discussing a point or slide before annotating it. Sometimes, this involved waiting until a slide transition. As one student put it: “there were a couple of circumstances where I was going to ask for an example via the system, but waited until the next slide to see if he already had an example prepared."

We were surprised to discover that students also overcame feedback lag by annotating *ahead* of the lecture, on points that the instructor had not yet discussed. Many instructors have identified this premature annotation as a potential problem when trying CFS out. Indeed, asking a question *aloud* about a point ahead of the discussion would likely be socially unacceptable. However, CFS's private channel between student and instructor and the persistence of annotations (*i.e.*, the fact that they were still available when the instructor reached the point in question) render this behavior acceptable. For example, two students asked for more explanation of a bullet at the bottom of a slide while the instructor was still discussing an earlier bullet (as recorded in the logs of the students' feedback annotations and the instructor's ink annotations). When the instructor reached the bullet in question a minute after the annotations appeared, he gave it special attention. Soon, both students annotated the bullet with GOT ITs. In the interview, the instructor described this pattern of student-guided discussion as a particularly effective and non-intrusive use of CFS. The episode described in Section 4.3 also illustrates how CFS helped to overcome feedback lag. The student in that episode tried to speak up but eventually the opportunity passed, and she used CFS feedback instead. Indeed, that student was one of the ones who reported feedback lag as a problem on the survey.

Two students who did not report suffering feedback lag with spoken questions reported suffering it with CFS. Because of the speed with which the instructor sometimes covered slides, the students sometimes did not have time to annotate the previous slide before it was replaced. As one student put it: "There was one time that I wanted to click on more explanation about a topic, but. . . by the time I tired [sic] to click the topic, [the instructor] had already zipped past two slides." Broader navigational capabilities might address these students concerns.

Student Apprehension

Students felt apprehensive about speaking in class. 6 of 11 students surveyed cited inhibiting factors, with four mentioning nervousness and three mentioning the large class size. Some evidence suggests that CFS helped to address student apprehension. None of the 6 students who mentioned apprehension as limiting their spoken participation mentioned apprehension as limiting their participation through CFS. Only one student mentioned apprehension as a limiting factor in CFS at all. That student wrote: "if I were more concerned with what the other students thought of me, I would have been wary of having the students behind me notice my mouse click…" This insight is a reminder that the "guarantees" of anonymity we build into computer systems may not protect users' identities in real environments. Still, this student was not personally apprehensive. Two specific episodes in class reinforce this disparity of apprehension between CFS and spoken participation. In each one, the instructor tried and failed to elicit spoken participation by a student who had given feedback. Neither feedback lag nor the single-speaker paradigm adequately explains the student's reticence to respond as there was an express opening to speak. Apprehension is a likely explanation, especially since the student who made both feedback annotations was one of those who cited nervousness as an an inhibiting factor in the survey.

Although anonymity helped address apprehension, it also created significant obstacles to the instructor's understanding of feedback, sometimes because he could not see correlation of feedback (*i.e.*, which of multiple comments came from the same student), and sometimes because he could not see the specific identity of students giving feedback. As an example of the correlation problem, one student annotated three Java classes out of a set of six with MORE EXPLANATIONs to indicate which ones confused him. Without knowing that these distinct annotations came from the same student, the instructor had difficulty interpreting them as expressing a single concept. Also, he was unable to judge how many students were, in fact, confused. Another problematic interaction was when students annotated over their feedback with a GOT IT rather than clearing the feedback. Without correlation, these annotations could represent lack of consensus about whether to talk more or move on. Not knowing students' identities also meant that the instructor could not evaluate feedback with respect to his knowledge of the student who gave it (as he did with spoken feedback), and he could not follow up on feedback outside of class (as he sometimes did with spoken feedback).

Some positive evidence pointed to the need for correlation, as well. The instructor actually requested correlation of feedback, saying: "I want to know that one person [as opposed to several] wants some more information about the slide." The one mechanism that *was* correlated, removing feedback, met with approbation from the instructor: "... if I could see that the screen had gone clean [of feedback annotations] again, I think that makes sense."

Comment Verbalization

A small amount of evidence that CFS helped to address articulation emerged from the data. Communication of issues through CFS were often resolved. About one in every 3 non-GOT IT annotations was either cleared or marked with GOT IT to indicate it had been addressed. These and perhaps other feedback were successfully answered. (At least one student mentioned a successfully answered feedback that was not cleared or marked GOT IT in the survey. Another student explicitly mentioned not bothering to clear feedback or use GOT IT.) On the other hand, only 3 of the 11 students surveyed found the feedback categories sufficient to express their feedback while 6 others requested the ability to type in their own questions or comments, suggesting difficulty in articulating their questions through placement and the existing categories.

The instructor, for his part, found interpreting the EXAMPLE feedback straightforward but expressed difficulty in interpreting MORE EXPLANATION and GOT IT. He simply could not find any useful interpretation of GOT IT feedback, saying: "I didn't know how to interpret it other than 'You're boring me, I already understand it.' " The instructor frequently did respond to MORE EXPLANATION; however, he lacked confidence in his interpretation of the feedback, saying: “I never really did get a good solid feeling for the kind of conversation I was able to have with people as a result of [More explanation feedback]. . . . I could talk for a little longer and then I kind of run down and I think, . . . ‘Was that what the confusion was?’ It was really hard to tell whether I was addressing the issue or not.” Still, the instructor was able to make deductions about the meaning of some annotations. He described one complex deduction based on context in the interview: “I interpreted that [annotation on utility functions] to mean . . . give me some examples. And since the examples are right there underneath, you know, search, sort, and initialize, . . . I thought to myself that it wasn’t clear that those were explicit examples.”

Single-speaker Paradigm

In a technical sense, the single-speaker paradigm is no longer a problem in CFS: multiple students *can* express themselves at the same time. In practice, a new issue arises: whether the instructor can manage *many* “speakers”. For example, during one heavy period of feedback with 7 annotations by 4 different students on a single slide, neither observer was aware that the instructor was responding to feedback,[3] despite specifically watching for evidence of interactions and the effects of CFS. However, the instructor was quite flustered by this episode, describing himself as “apparently having had some sort of brain seizure” while responding to the feedback. The instructor felt this was a regular problem with CFS. With more students and more feedback, the problem could escalate. Further research will show whether aggregation techniques and practice will allow instructors to cope with the “multi-speaker paradigm.”

4.5 Future Work

Although this work shows the value of contextual feedback, it also suggests new ways to address the challenges we identified.

[3]At the time, the observers saw only the shared view; later, they could “peek” at the instructor’s view.

In preliminary meetings with the student volunteers, several students suggested that they would feel more at ease asking questions if they knew other students in the class had questions. Allowing students to see aggregated feedback might encourage more response. On the other hand, even anonymized, exposing a student's feedback to other students might also make that student more apprehensive. In general, the space of privacy and anonymity policies and their specification by the instructor and students is a rich area for research.

Mollifying our strict anonymity policy to reveal correlations among feedback or even reveal students' identities (to the instructor alone) might reduce difficulties interpreting the meaning of multiple feedback annotations and enable new interactions. Allowing extra categories (as ActiveClass does [103]) or supporting free text might help students express themselves and provide extra confidence to instructors in their interpretation of feedback. Displaying extra information about a comment on mouseover would be one mechanism to address all of these extensions: popping up associated freeform text, highlighting correlated feedback annotations, and revealing the identity of the student that gave a comment.

Allowing the instructor, like students, to modify feedback would enable in-system mechanisms for following up questions without violating students' anonymity (as the instructor in our study requested). Since the instructor will likely follow up questions after class, these mechanisms should work with archival data.

Four students in the survey mentioned being uncomfortable with the laptop form factor, *i.e.*, having trouble managing their notes and the computer in the same desk space. Moving to a smaller form-factor device with pen-based input such as a Tablet PC might address this concern while reducing the attention needed to interact with the system (because of the pen-based direct-manipulation interface). Supporting student notetaking on the same device as CFS or other student interaction systems (as with SIP in Chapter 5) should further reduce demands on students' desk space.

The preliminary experiment validated the design of the computer-mediated feedback system and provided evidence for system use in the classroom. Longer term experiments, experiments with more participants, and experiments across different types of classes might expose interesting new issues. Indeed, one student reported apprehension in the survey because of the small size of the computer-mediated group: “If there were more people ... using the system, ... I would feel more anonymous and more likely to make comments.” Studying the use of the system throughout a term will provide valuable insights about the change of use over time. Along with new effects of the system on the class, larger-scale studies would also allow exploration of new uses of the system, such as archival improvement of lecture materials across terms.

4.6 Conclusions

We have described the design of the Classroom Feedback System (CFS), a system for promoting interaction in large classes. Through the design and deployment of the computer-mediated feedback system, we successfully engineered a more interactive learning environment with a new medium for student-instructor interaction. In the process, we identified key challenges to interaction and grounded these challenges in literature, discussions with instructors, and data from large classes. Our design principles, realized in an implemented system, describe how the computer-mediated communication system targets these challenges. Analysis of an experiment with our designed system demonstrated its success in promoting interaction and revealed interesting interplay with the challenges we identified.

The implications of the work extend already to any computer-supported system for interaction in large classes. Any such system will face these challenges and might benefit from integrating our design principles.

Chapter 5

THE STRUCTURED INTERACTION PRESENTATION (SIP) SYSTEM

This chapter describes the Structured Interaction Presentation (SIP) System, which extends traditional slideware to support student interaction in the classroom using networked computers. SIP's presentation design environment enables instructors to craft interactive exercises integrated with their presentations. SIP's execution environment manages student participation in the exercises and incorporates their response into the public presentation in real-time. We have implemented a prototype of the SIP System, used it to design a suite of interactive presentations, and run one experimental hour-long class using SIP.

The key contributions of the SIP project are the concept of integrating highly flexible interactive exercises into presentation design and execution, a pedagogical framework justifying this concept, samples of such integrated exercises (several of which appear in Appendix A), the design insights we gained both during the design process and during evaluation, and the SIP prototype itself.

This chapter begins by establishing a pedagogical framework for interactive slideware. Section 5.2 then introduces the SIP System using two sample exercises created in SIP. Next, Section 5.3 describes key design goals and decisions that informed SIP's design as well as key design choices that are *not* implemented in this prototype. Section 5.4 describes SIP's architecture in detail, covering both the implemented prototype and intended aspects of the system that are not yet implemented. Finally, Section 5.5 describes the experimental class session that used SIP and its evaluation.

We argue that by generalizing the slideware, we are creating a "score" to complete the "conductor-of-orchestra" metaphor for the interactive classroom introduced by Roschelle and Pea [87].

5.1 Slideware as Score

In "A Walk on the WILD Side", Jeremy Roschelle and Roy Pea describe a progression from traditional lectures — "sage-on-the-stage" — through small group learning environments — "guide-by-the-side" — to a new paradigm of whole class interaction. They call this new paradigm "conductor-of-orchestra": a class-wide learning experience designed by the instructor and supported by wireless devices in which the instructor acts as conductor and the students as orchestra [87]. With the instructor guiding them, the class works together to create a joint learning experience.

This is a powerful vision of the potential for computer-supported collaborative learning. However, Roschelle and Pea's vision lacks a critical element of orchestral performance: the score. The score mediates the processes of composition, sharing, execution, and reflection over a musical piece. The score for an interactive classroom must similarly mediate composition, sharing, execution, and reflection over a learning experience. The score would embody the interwoven episodes of discussion, collaboration, lecture, reflection, and problem solving that make up the classroom symphony and support instructor and students in the vivid interplay of learning. Furthermore, the score would represent a powerful artifact for sharing course and exercise plans among instructors.

The closest approximation of the interactive score currently in use is the slide-based presentation. Slides organize the instructor's discussion. As a public entity, they also play a broader mediating role: focusing student attention, mediating communication (through reference to shared context), and acting as a framework for reflection. In mediating these cognitive and communicative processes, slide-based presentations also naturally shape them.

Unfortunately, current slideware systems do not support the interactive vision of the "conductor-of-orchestra". Indeed, while students laud slideware for enforcing instructors' preparation and organization, they decry it for creating a passive classroom environment [53]. Hordes of passionate detractors have inveighed against slideware in the literature as well, warning of the powerful negative impact these tools have on the presentation and teaching practices of their users (*e.g.*, [33, 104]).

The need for an interactive score and the availability of influential but limited slideware tools suggests a new direction: redesign slideware to better support the vision of an interactive classroom. Our work with SIP explores this solution. We make a conscious choice of the pedagogical environment we envision for the classroom — an interactive and engaged environment as Roschelle and Pea describe and as constructivist theory propounds — and then use design experimentation to construct the tools that will open up the environment we envision.

SIP provides the facilities needed to structure interactive exercises — that is, to act as the interactive score — by extending the slide-based presentation paradigm to encompass these exercises. SIP supports design, execution, sharing, and reflection on interactive classroom activities. Furthermore, because SIP builds on a familiar and widespread slideware tool (PowerPoint), it benefits from the strengths of slideware as a mediating artifact and provides instructors and students a natural adoption path from well-organized, well-prepared, but passive classes toward the learning orchestra.

Consonant with our goal of providing an interactive score for the classroom is the goal of integrating the score into the cognitive process of active learning. As Hollan *et al.* put it, "Work materials from time to time become elements of the cognitive system itself. Just as a blind person's cane or a cell biologist's microscope is a central part of the way they perceive the world, so well-designed work materials become integrated into the way people think, see, and control activities, part of the distributed system of cognitive control [58, page 177]."

As an element of the classroom cognitive process, SIP promises a set of advantages for teaching and learning. By embodying the organizational details of interactive exercises (order, timing, roles of participants, *etc.*) and even operationalizing these details, SIP offloads cognitive effort of interaction management from classroom participants, especially the instructor. SIP shifts interaction planning effort to the instructor's design of the presentation, and this early effort is repaid in reduced in-class effort and potential for reuse in future offerings of the same class or future uses of similar interactive exercises. From a pedagogical perspective, reifying interaction design in the design of presentations will encourage instructors to consider interaction early in their design process and to think about how the interactive exercises relate to the surrounding lecture slides.

5.2 SIP by Example

This section presents SIP's functionality by example, describing the construction and execution of one interactive exercise and the execution of another. Both exercises are from the hour-long experimental class session described in Section 5.5. The screenshots below are taken from that presentation and display actual student data. The session's topic was risk assessment (in a public health context); I acted as the instructor; and the students were volunteer faculty, staff, and students from my department (few of whom had any formal background in risk assessment).

Throughout this example, note that the analogous relationship of the SIP presentation to its class session and a musical score to its orchestral performance. The composer uses the score to record and scaffold her thought process; the score communicates the composer's intent to the conductor and performers; and the conductor and performers use the score to organize and guide their performance. Similarly, the instructor uses the presentation to record and scaffold her thought process; the presentation communicates the instructor's intent during the class (and potentially to other instructors who use her slides); and the instructor and students use the presentation to organize and guide their learning experience.

5.2.1 Design of an Attitude Change Poll

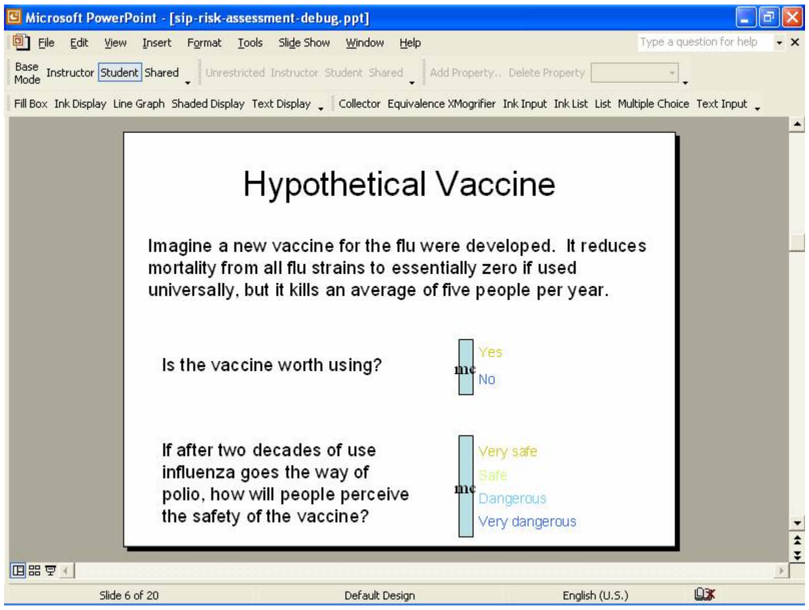
One of the instructor's key goals in the risk assessment class session was to change students' attitudes about public risk perception. In particular, he wanted students to believe that the perceived risk of a safety technology increases as the danger the technology addresses recedes. (Perception of asbestos is a prime example of this increased perceived risk.)

Unfortunately, straight lecture is particularly ineffective at changing attitudes [15]. Instead, the instructor decided to create a pair of interactive exercises focused on a hypothetical safety technology, a one-time vaccine for the flu. The first exercise was to be a poll at the beginning of class in which students would express their beliefs about public perception of flu risk. The second exercise would offer the same poll again and give students an opportunity to consider (and discuss) whether their beliefs had changed and why.[1] Each exercise would be a pair of multiple choice questions asking whether the vaccine is worth using and how the public would perceive the risk of the vaccine after the flu had been largely eradicated.

To design the exercise, the instructor laid out the static content and placeholders the interactive elements ("widgets") in the first exercise as shown in Figures 5.1 and 5.2. The shaded textboxes labeled "mc" and "fillbox" are placeholders for SIP widgets. These placeholders are automatically replaced with widgets when the presentation is exported to SIP's presentation database. Instructors create new placeholders either by copying and pasting them from existing presentations or by clicking the appropriate button on the toolbar at the top of Figure 5.1.

The instructor next configured the properties of the SIP widgets. A property is either an input or output parameter of a widget. For example, a text input widget that solicits text from the user might have a "text" output property that encodes the text the user types and a "font size" input property that sets the size of the text. Each property has a static value, and input properties can also be controlled by a simplified SQL query that is used to update the

[1]Refer forward to Figures 5.4 and 5.5 for a gestalt of the final exercise.


Microsoft PowerPoint - [sip-risk-assessment-debug.ppt]
File Edit View Insert Format Tools Slide Show Window Help
Type a question for help
Base Mode Instructor Student Shared
Unrestricted Instructor Student Shared
Add Property.. Delete Property
Fill Box Ink Display Line Graph Shaded Display Text Display
Collector Equivalence XMogrifier Ink Input Ink List List Multiple Choice Text Input
Slide 6 of 20
Default Design
English (U.S.)

Hypothetical Vaccine

Imagine a new vaccine for the flu were developed. It reduces mortality from all flu strains to essentially zero if used universally, but it kills an average of five people per year.

Is the vaccine worth using?

mc

Yes

No

If after two decades of use influenza goes the way of polio, how will people perceive the safety of the vaccine?

mc

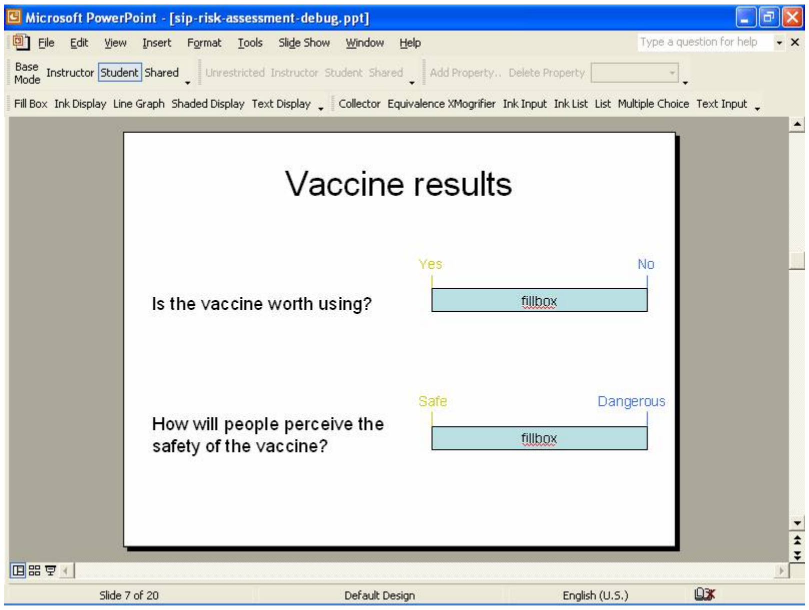
Very safe

Safe

Dangerous

Very dangerous

Figure 5.1: Design view of an interactive poll from the experimental class (see Section 5.5). Most slide elements are normal PowerPoint shapes, but the two shaded boxes labeled “mc” are placeholders for SIP multiple choice widgets. Figure 5.4 shows the student’s view of the slide during class.


Microsoft PowerPoint - [sip-risk-assessment-debug.ppt]
File Edit View Insert Format Tools Slide Show Window Help
Type a question for help
Base Mode Instructor Student Shared
Unrestricted Instructor Student Shared
Add Property.. Delete Property
Fill Box Ink Display Line Graph Shaded Display Text Display
Collector Equivalence XMogrifier Ink Input Ink List List Multiple Choice Text Input
Slide 7 of 20
Default Design
English (U.S.)

Vaccine results

Is the vaccine worth using?

Yes

No

fillbox

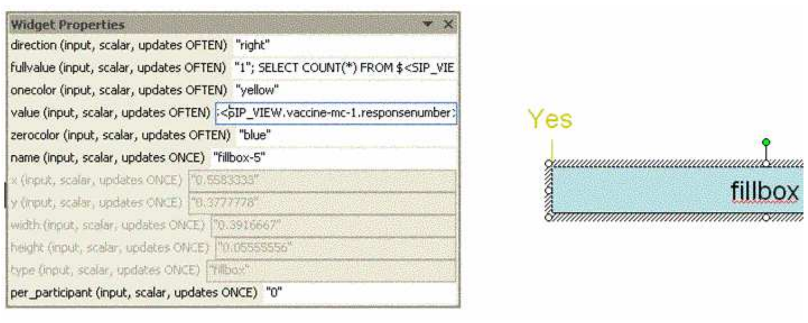
How will people perceive the safety of the vaccine?

Safe

Dangerous

fillbox

Figure 5.2: Design view of the result slide for the interactive poll in Figure 5.1. The shaded boxes labeled “fillbox” are placeholders for SIP widgets. Figure 5.5 shows the student’s view of the slide during class.



Widget Properties

direction (input, scalar, updates OFTEN)	"right"
fullvalue (input, scalar, updates OFTEN)	"1"; SELECT COUNT(*) FROM $<SIP_VIE
onecolor (input, scalar, updates OFTEN)	"yellow"
value (input, scalar, updates OFTEN)	:<SIP_VIEW.vaccine-mc-1.responsenumber:
zerocolor (input, scalar, updates OFTEN)	"blue"
name (input, scalar, updates ONCE)	"fillbox-5"
x (input, scalar, updates ONCE)	"0.5583333"
y (input, scalar, updates ONCE)	"0.3777778"
width (input, scalar, updates ONCE)	"0.3916667"
height (input, scalar, updates ONCE)	"0.05555556"
type (input, scalar, updates ONCE)	"fillbox"
per_participant (input, scalar, updates ONCE)	"0"

Figure 5.3: Design view as the instructor edits one of the “fillbox” widgets from Figure 5.2. These properties control the interactive semantics of the widget.

input property’s value whenever the properties it depends on change.

Eventually, property editing in SIP will conform to standard PowerPoint interface mechanisms, using visual design elements or “wizards”. For now, the properties panel directly exposes the values and queries described above. Figure 5.3 shows the instructor configuring the properties of the fill box.

Having completed this first exercise, the instructor wanted to create a similar exercise at the end of the lesson that would compare students’ first answers with their later answers. To accomplish this, he simply copied the two slides representing the first poll to the end of his presentation. He then copied the “fillbox” widgets on the result slide so that there are two pairs of them, one for students’ initial responses and one for their later responses. He also rearranged the static slide elements slightly to make clear what the new slide represented. Finally, he changed the widget name referred to in the new “fillbox” widgets’ queries so that they summarize the new poll rather than the old one.[2]

In a final pass, the instructor realized that the multiple choice questions make sense on the students’ view but not on the projector view. He selected the multiple choice widgets and used the “Student” visibility button (one of the small buttons at the top of Figure 5.1)

[2]In practice, the design of the first interactive poll also used copy-and-paste from similar previously designed exercises. Support for copy-and-paste is crucial to allow this kind of sharing.

to restrict visibility of those objects to just the student devices. He then put text suggesting students perform the exercise on their devices over the same area as the multiple choice questions and restricted that text to the “Public” view.[3]

Through this design process, the instructor encoded the structure, flow, and visual form of the polls in the SIP presentation. Appendix Figures A.14–A.19 and A.38–A.41 show all relevant views of the slides in this exercise, both during its design and execution.

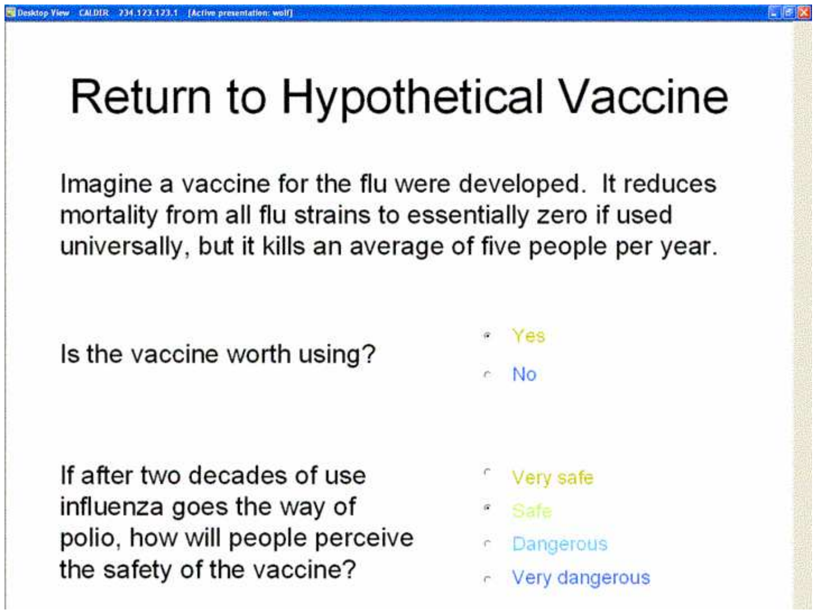
5.2.2 Execution of an Attitude Change Poll

When the instructor is ready to execute a SIP presentation, she exports the presentation to a database through a simple form (again, provided as an add-in to PowerPoint). SIP stores the presentation slides as images with the widget placeholders elided and separately lodges a description of each widget and its properties. The execution environment can then render the slides with SIP widgets overlaid, configured with student data.

In class, the instructor runs the SIP presentation using an extension of Classroom Presenter. Classroom Presenter is a presentation system that runs on a Tablet PC, a pen-based mobile computer, and allows the instructor to handwrite over and navigate through computer-projected slides. Slides and ink are multicast wirelessly from the instructor’s device to student devices and the projected display. Chapter 3 describes Classroom Presenter in detail.

When the instructor navigates to one of the poll slides, SIP automatically presents the appropriate view to each participant, including interactive widgets. Figure 5.4 shows the question screen of the second poll as a student fills it out. When he’s ready, the instructor advances to the results slide. Figure 5.5 shows the results from the test session. Note that students’ opinion on whether the vaccine was worth using changed little while their opinion about the perceived risk of the vaccine changed more. This change sparked exactly the kind of productive discussion the instructor had anticipated. Appendix Figures A.14–

[3]In practice, the instructor also added four more “fillbox” widgets restricted to instructor visibility that helped him manage the flow of the exercise by displaying how many students had responded to each poll.


Desktop View CALDER 234.123.123.1 [Active presentation: wolf]

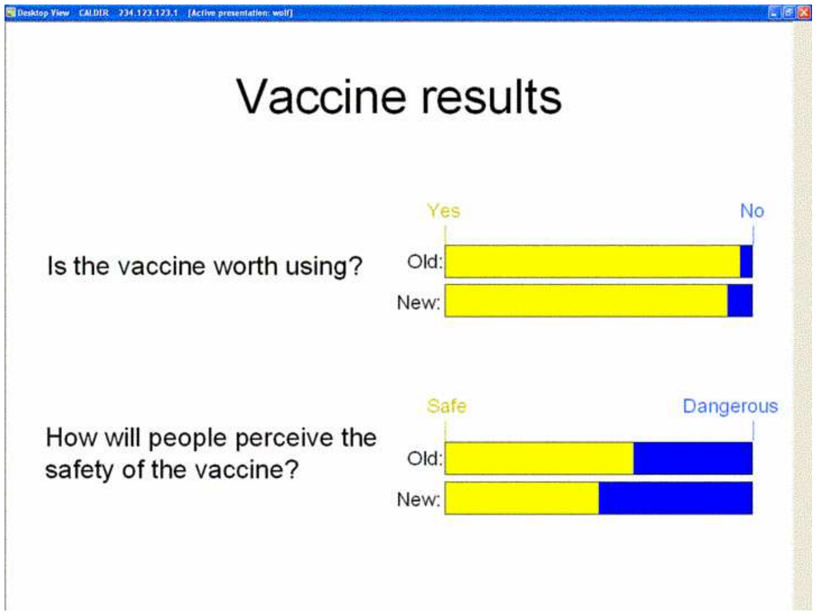
Return to Hypothetical Vaccine

Imagine a vaccine for the flu were developed. It reduces mortality from all flu strains to essentially zero if used universally, but it kills an average of five people per year.

Is the vaccine worth using? [✓ Yes] No

If after two decades of use influenza goes the way of polio, how will people perceive the safety of the vaccine? Very safe [✓ Safe] Dangerous Very dangerous

Figure 5.4: A student's view of the second prompt of the hypothetical flu vaccine poll.


Desktop View CALDER 234.123.123.1 [Active presentation: wolf]
Yes
No
Old:
New:
Safe
Dangerous
Old:
New:

Vaccine results

Is the vaccine worth using?

How will people perceive the safety of the vaccine?

Figure 5.5: A student's view of the second result slide of the hypothetical flu vaccine poll. Note that results from both the first and second poll are displayed.

A.19 and A.38–A.41 show all relevant views of the slides in this exercise, both during its design and execution.

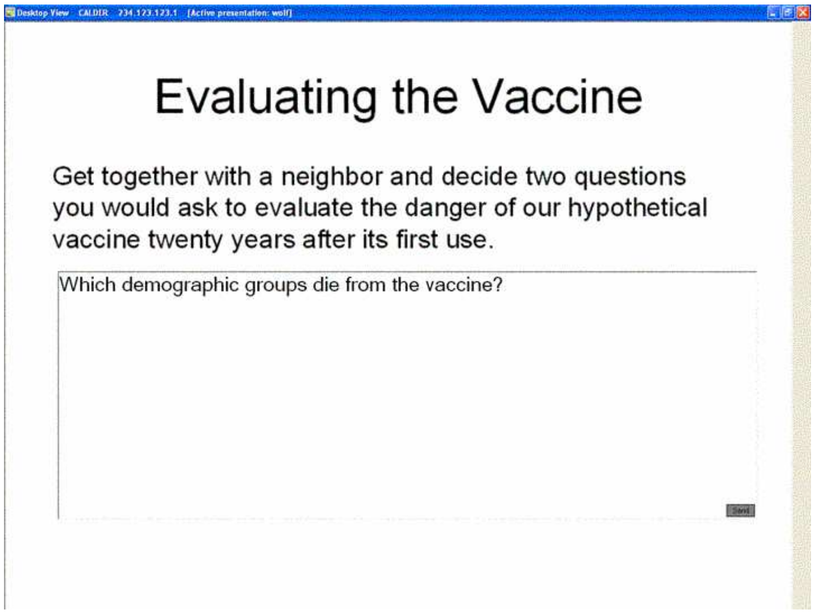
5.2.3 Distributed Human Computation Exercise

SIP supports not just closed-form inputs like multiple-choice questions but also free-form inputs like text and drawings. Aggregating such inputs on the basis of objective qualities — *e.g.*, the length of a text entry, the inclusion of keywords in a text entry [37], or the average slope of ink segments in a diagram — is straightforward. However, aggregating these inputs based on an understanding of their meaning is beyond the capability of current natural language and image/sketch understanding. For example, automatically clustering textual responses by their semantic similarities and ordering them by their relevance, value for discussion, and clarity of exposition are all challenging tasks, made more difficult by the lack of topic constraints in SIP.

This second exercise demonstrates SIP's solution to this problem. The exercise comes from the same interactive presentation as the previous example and also centers around risk assessment in a public health context. One of the instructor's goals was that students leave the class session prepared to ask informed questions about public health risks. Toward that end, he envisioned an exercise in which each student contributes a question to evaluate the hypothetical flu vaccine mentioned above. Naturally, he also wanted to intelligently discuss the questions with the whole class. To make that discussion manageable, he planned to rapidly select a few clearly-worded questions that would be representative of the diversity of responses the class generated.

The exercise he used is an innovative interaction pattern in SIP called "distributed human computation" (DHC). In a DHC exercise, every student supplies one or more responses to a prompt through their devices as in Figure 5.6. In this particular case, students were asked to pair with their neighbors and together contribute two responses (one on each student's device).

After the students provide their input, the instructor transitions to the next slide. Fig-

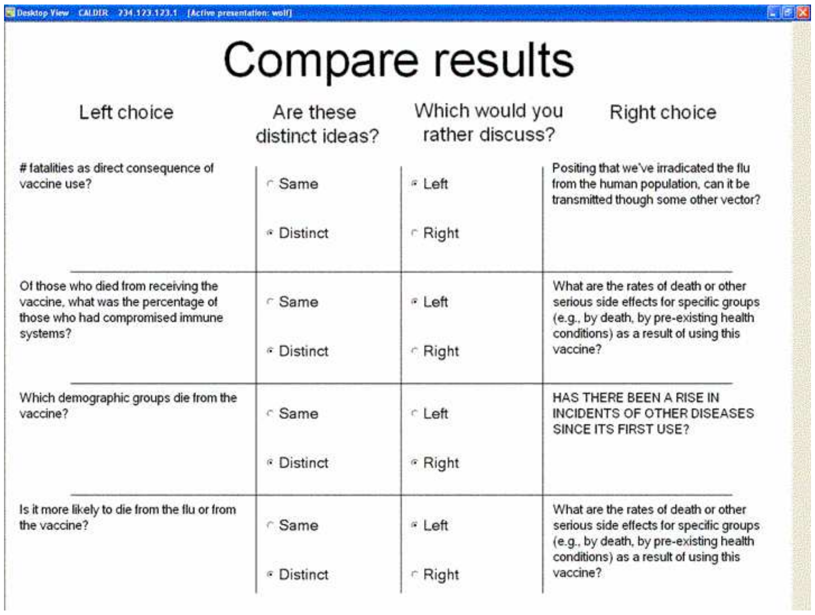

Desktop View CALDER 234.123.123.1 [Active presentation: wolf]
Send

Evaluating the Vaccine

Get together with a neighbor and decide two questions you would ask to evaluate the danger of our hypothetical vaccine twenty years after its first use.

Which demographic groups die from the vaccine?

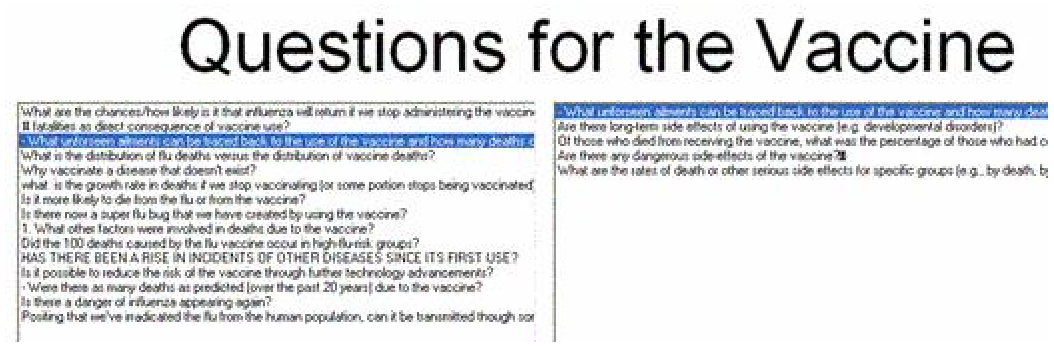
Figure 5.6: A student’s view of the prompt for a distributed human computation exercise.


Desktop View CALDER 234.123.123.1 [Active presentation: wolf]

Compare results

Left choice	Are these distinct ideas?	Which would you rather discuss?	Right choice
# fatalities as direct consequence of vaccine use?	Same [✓ Distinct]	[✓ Left] Right	Positing that we've irradicated the flu from the human population, can it be transmitted though some other vector?
Of those who died from receiving the vaccine, what was the percentage of those who had compromised immune systems?	Same [✓ Distinct]	[✓ Left] Right	What are the rates of death or other serious side effects for specific groups (e.g., by death, by pre-existing health conditions) as a result of using this vaccine?
Which demographic groups die from the vaccine?	Same [✓ Distinct]	Left [✓ Right]	HAS THERE BEEN A RISE IN INCIDENTS OF OTHER DISEASES SINCE ITS FIRST USE?
Is it more likely to die from the flu or from the vaccine?	Same [✓ Distinct]	[✓ Left] Right	What are the rates of death or other serious side effects for specific groups (e.g., by death, by pre-existing health conditions) as a result of using this vaccine?

Figure 5.7: A student’s view of the critical comparison phase of a distributed human computation exercise.



Questions for the Vaccine

What are the chances/how likely is it that influenza will return if we stop administering the vaccin
# fatalities as direct consequence of vaccine use?
- What unforeseen ailments can be traced back to the use of the vaccine and how many deaths c
What is the distribution of flu deaths versus the distribution of vaccine deaths?
Why vaccinate a disease that doesn't exist?
what is the growth rate in deaths if we stop vaccinating (or some portion stops being vaccinated
Is it more likely to die from the flu or from the vaccine?
Is there now a super flu bug that we have created by using the vaccine?
1. What other factors were involved in deaths due to the vaccine?
Did the 100 deaths caused by the flu vaccine occur in high-flu-risk groups?
HAS THERE BEEN A RISE IN INCIDENTS OF OTHER DISEASES SINCE ITS FIRST USE?
Is it possible to reduce the risk of the vaccine through further technology advancements?
- Were there as many deaths as predicted (over the past 20 years) due to the vaccine?
Is there a danger of influenza appearing again?
Positing that we've eradicated the flu from the human population, can it be transmitted though so

- What unforeseen ailments can be traced back to the use of the vaccine and how many dea
Are there long-term side effects of using the vaccine (e.g. developmental disorders)?
Of those who died from receiving the vaccine, what was the percentage of those who had o
Are there any dangerous side-effects of the vaccine?
What are the rates of death or other serious side effects for specific groups (e.g., by death, b

Figure 5.8: The instructor's view of aggregated results from a distributed human computation exercise (cropped). Section 5.5.2 lists all the results from this exercise, grouped and ranked as they were in the class (page 168).

ure 5.7 shows one student's view of the slide. Here, each student analyzes four randomly selected pairs of responses and determines for each pair whether they are equivalent (in this case, "Do these two questions address the same underlying issue?") and which the student would rather discuss. This data is used to aggregate — group and rank — responses, but the exercise is also pedagogically valuable, forcing students to critically analyze their fellow students' responses.

Finally, the instructor transitions to a third slide to discuss the results. Figure 5.8 shows the instructor's view of the aggregated results. The student responses have been collected into equivalent groups. The left column shows a representative from each group, ordered by how much students wished to discuss each group. Each representative is the element of the group students most wanted to discuss. The right column shows all elements of the group selected on the left, again ordered by student preference. The current selection in the right column is also shown on the public display.

The distributed human computation exercise exploits students' engagement in a pedagogically sound activity (critiquing other students' responses) to address the aggregation problem described above. Without recourse to natural language understanding, SIP can still aggregate student responses, clustering them by their similarity to each other and ranking them by whatever criterion the instructor deems appropriate.

An instructional designer interested in creating new SIP widgets might also explore other models for the student analysis phase of DHC. For example, each student might be given a list of some of the other students' responses and asked to select the response(s) that most closely match their own. (This model resembles ActiveClass's extensible polls [103].) Alternatively, natural language understanding or progressive refinement based on initial student data might be used to tailor the sequence of response pairs given to each student.

5.3 SIP Design Principles

As with the Classroom Presenter and Classroom Feedback System (CFS) projects (see Chapters 3 and 4), the SIP research project is structured as a design experiment [24]: an iterative investigation of the design and use of an intervention. As such, two of the primary contributions of the SIP project are the set of design decisions we made in creating SIP and the experiences and evidence that support (or contradict) those decisions.

As with Presenter and CFS, these principles developed from observation of the target venue and from studies of the literature. SIP's principles also arise from experience designing and studying Presenter and CFS. This section, describes the core design principles that contributed to SIP's design as well as a set of design principles that are *not* embodied in the SIP prototype implementation design but which emerged as critical missing elements.

5.3.1 Embed Interaction into the Mediating Slides

The theme of exploiting the mediating nature of slides (as described in Chapter 2) is central to SIP's design. SIP exploits the mediating slides by embedding the machinery of interaction directly into the slides themselves: students provide input by typing, clicking, or drawing on designated areas of the slides; the results of interactive exercises appear as displays on the slides; and the instructor navigates interactive exercises by navigating through the slides.

Embedding interactive exercises into the slides has several advantages: (1) intuitive navigation and control of exercises, (2) offloading some of the in-class cognitive effort of exercise organization onto out-of-class slide preparation, (3) mediating transitions between lecture-style class segments and interactive exercises, and (4) encouraging sharing and reuse of interactive exercises. This section addresses each of these four points in turn and then notes some of the drawbacks of embedding interaction into slides.

Instructors use a familiar and natural navigation metaphor in SIP to control the timing of interactive exercises. The input and display phases of each exercises are tied to individual slides in the deck. The instructor simply advances through the deck, reaching planned exercises in a natural progression. In unusual cases when an instructor wants to skip over, jump to, or return to an exercise, she uses Presenter's normal slide navigation mechanisms (including slide scanning and preview features) to find a target slide and jump to it. When an instructor is ready to close input for a particular interaction, she navigates away from the input slide, and the input area disappears from students' screens.

In this context, SIP externally supports instructors' cognitive processes. The slides' ordering offloads prior cognitive effort in organizing activities and arguments. Like a chef using spatial layout of cooking utensils and ingredients to "remember" which pairs go together [58], SIP remembers organization of the interactions that will occur in class and even operationalizes more detailed organizational choices (such as requiring only numerical responses to a certain question).

Embedding interactive exercises into slides also scaffolds students' participation in the exercises. Students already use slides as cues to the presentation's organization, expecting and associating spoken content with slide content. If the slides also indicate when students should provide input, they naturally expect to participate. As one student SIP user put it: "I felt as though I really didn't have much choice but to participate when the lecture reached an activity. (I think this is a good thing.)" (The emphasis is from the original response.)

Therefore, SIP mediates the transition between student-focused activity and instructor-focused lecture, a particularly difficult transition for instructors. Instructors discussed their concern about this transition with us in design discussions. McConnell similarly describes instructors' fear of losing control, especially in large classes. Instructors fear that transitions from "normal" lecture to active learning and back will telescope into serious interruptions [68]. SIP makes the projected slide a natural reference point for students to determine whether the class is currently meant to be student- or instructor-focused. Furthermore, with the slide displayed on each student's device, that reminder of the current focus is always available in the student's visual field.

Finally, embedding interactive exercises into slides encourages sharing and reuse of these exercises. Slides are often a key artifact for a previous instructor of a course to prepare a future instructor or even for an instructor to refresh herself on her own course. Indeed, this kind of archival use was frequently cited by instructors as an advantage of Presenter and the Classroom Feedback System — for remembering which slides were interesting, confusing, or incorrect. With embedded interactive exercises, an instructor sharing or reusing slides receives those exercises along with the other slide content and notes. Eliding the exercises is more challenging for this new use than keeping them. Furthermore, an instructor who receives SIP exercises with a set of slides restructure those exercises to fit into her other presentations.

Along with the advantages of slides, SIP also reaps the disadvantages. The slide "deck" format discourages branching, contingent organization of exercises (although Presenter's navigation facilities allow some dynamic, manual reordering). Keeping all the SIP clients on the same slide makes extended, self-paced, individual or group activities challenging. As with static slides, it can be difficult to coherently distribute large sequences of SIP content across multiple slides. Slides tend to be short and simple, a style that Tufte derides [104] and Norvig mocks [79]. However, SIP also fulfills Tufte's vision of rich instructional artifacts by encouraging the solicitation and manipulation of significant amounts of data. Students in the experimental SIP class held similar excited discussions about "Tuftian" slides dense with data (*e.g.*, Figure 5.10) and slides showing the data they produced themselves (*e.g.*, Figure 5.9).

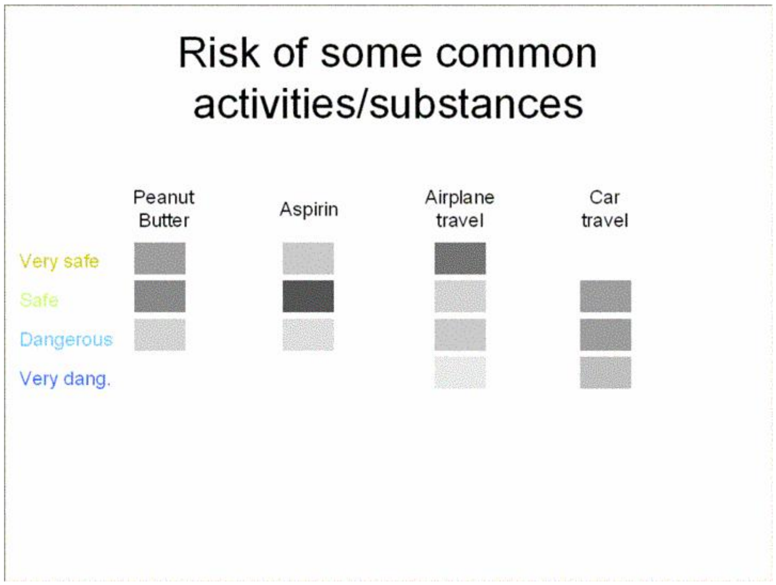


Figure 5.9: Results from an exercise asking students about the risks of four common activities and substances. (See Appendix Figures A.21–A.24 for more on this exercise.) This slide led to more than six minutes of animated discussion in class.

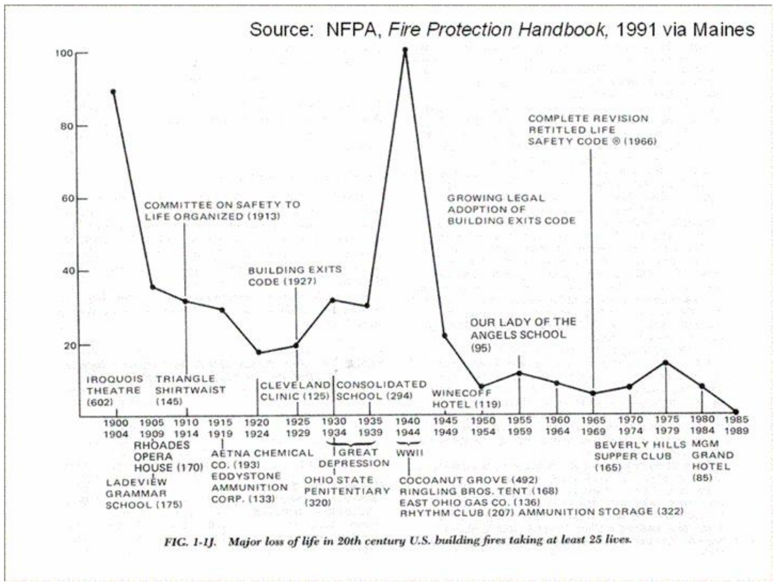


Figure 5.10: A plot of historical deaths from fire used to discuss the changing threat of fire over the course of the 20th century. The class spent almost four minutes discussing the meaning of this slide, trying to assimilate its dense information. (See Appendix Figures A.33–A.34 for more on this slide.)

SIP's focus on *pre*planning exercises and embedding them in the slides also restricts instructors' ability to create and modify interactive exercises on-the-fly. Instructors might want to dynamically generate interactive exercises for a variety of reasons, *e.g.*, polling the whole class about a point of confusion brought up by a single student's question or folding a current news event into an exercise. Furthermore, limiting instructors' flexibility in this way impedes our goals of supporting emerging innovative practices in the classroom and increasing the overall flexibility of "lecture" presentations.

Although this is currently a weakness in SIP, the seeds of dynamic exercise creation and editing are already present in the system. Instructors can already "edit" exercises using Presenter ink. Also, Presenter supports inserting slides and transitioning to a whiteboard workspace. In SIP, these interspersed slides could support simple, "prefabricated" exercises. For example, one flexible, prefab multiple choice exercise might insert two slides, the first with four multiple choice options labeled "(a)" through "(d)" and the second with a histogram showing response totals. The instructor can then use digital ink to write in a question and response labels. A more ambitious design would make the inked labels into SIP widgets that forward their contents to the histogram column labels on the next slide.

Finally, Truong *et al.* argue that sustainable adoption of an intervention like SIP precludes reliance on proprietary and expensive commercial software [103]. We recognize integration with the commercial PowerPoint software as a danger. Indeed, our choice to eschew PowerPoint for execution of presentations in SIP (and Presenter) reflects the same adoption concerns with respect to SIP's *student* users (see Section 3.5). Unfortunately, the benefits entailed from embedding interactions into the presentation and the adoption benefits of segregating the technologies seem to constitute an irreconcilable tradeoff.

5.3.2 Make Interaction Part of the Design Process

SIP supports carefully designed, premeditated interactive exercises. In this sense, SIP was a response to the undesigned, spontaneous nature of the Classroom Feedback System (CFS). CFS enabled students to contribute to the slides at any time, in any place. While this loose, student-initiated model did solicit feedback that would have been lost otherwise, instructors had trouble responding to the feedback "on their feet." With SIP, instructors define the parameters of interaction beforehand. While students' particular responses to interactive exercises may be unexpected, instructors can design the timing and nature of these opportunities to fit well with the class's flow.

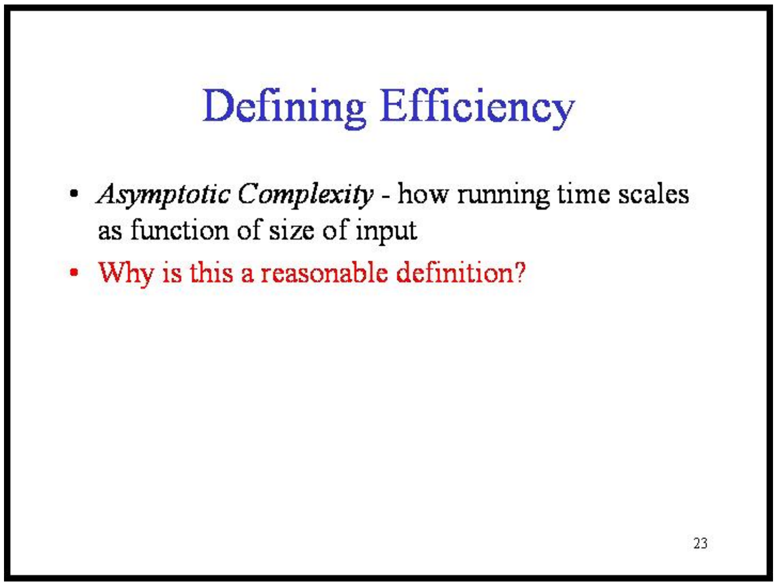
Folding interaction design into the presentation design process has other advantages as well. Affording interaction planning during instructors' presentation design process encourages them to adopt interaction as a teaching technique. Unifying these two design processes also fills an existing need. Instructors *already* often design interaction as they design their slides, inserting cues for interactive exercises even without operational support for interactions, as in Figures 5.11 and 5.12.

5.3.3 Support Innovation

As mentioned in Chapter 2, the most interesting and exciting innovations in computer-mediated communication systems arise from users' natural patterns of practice. A key design goal in SIP is to nurture these user innovations. SIP enables this innovation by (1) breaking interactive exercises down into simple constituents ("interaction widgets") that can be recombined into new exercises, (2) supporting multiple input and display modalities, and (3) facilitating creation of new interaction widgets. This section describes each of these stratagems.

Many classroom interaction systems support predefined interactive exercises. These exercises range from simple multiple choice questions summarized by histograms [61] to creative exercises like binned short response questions [37] or extensible polls [103]. However, the structure of these interactive exercises, and even many aspects of their timing and layout, is predefined by the underlying system. For example, [61] fixes the style of summary display as histograms, and each input question or poll associates directly with a single display for that question.

In contrast, SIP allows instructors to build exercises tailored to their needs from basic

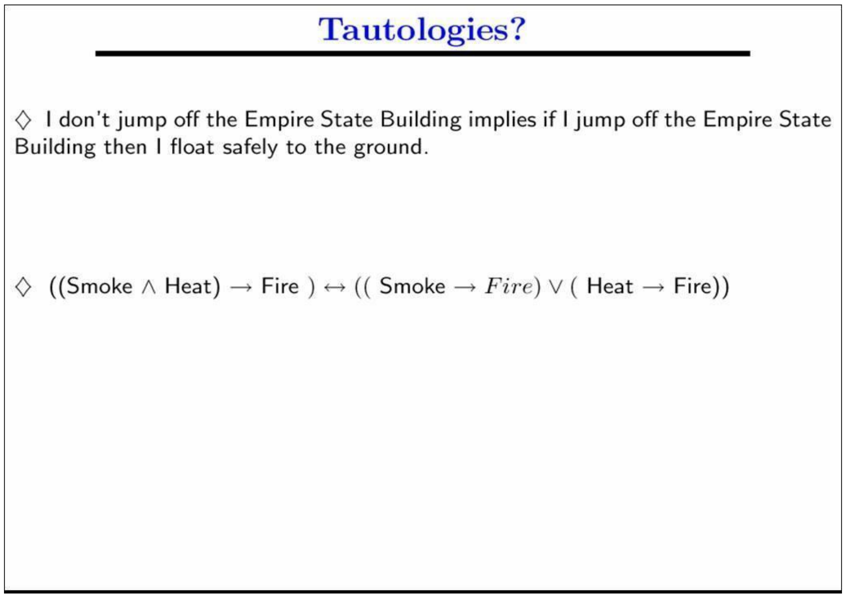


Defining Efficiency

- *Asymptotic Complexity* - how running time scales as function of size of input
- Why is this a reasonable definition?

23

Figure 5.11: A normal PowerPoint slide from a data structures and algorithms class. Despite the lack of direct support for interaction, the instructor has embedded an interactive exercise into the slide.

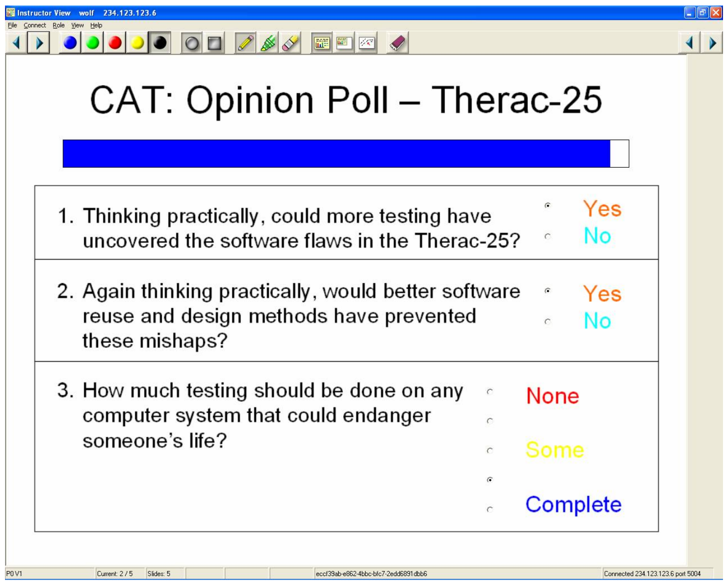


Tautologies?

◇ I don't jump off the Empire State Building implies if I jump off the Empire State Building then I float safely to the ground.

◇ ((Smoke $\wedge$ Heat) $\rightarrow$ Fire) $\leftrightarrow$ ((Smoke $\rightarrow$ $Fire$) $\vee$ (Heat $\rightarrow$ Fire))

Figure 5.12: A normal PowerPoint slide from an artificial intelligence class. As with Figure 5.11, the instructor has embedded an interactive exercise into the slide despite the lack of direct support for interaction.


Instructor View wolf 234.123.123.6
File Connect Role View Help

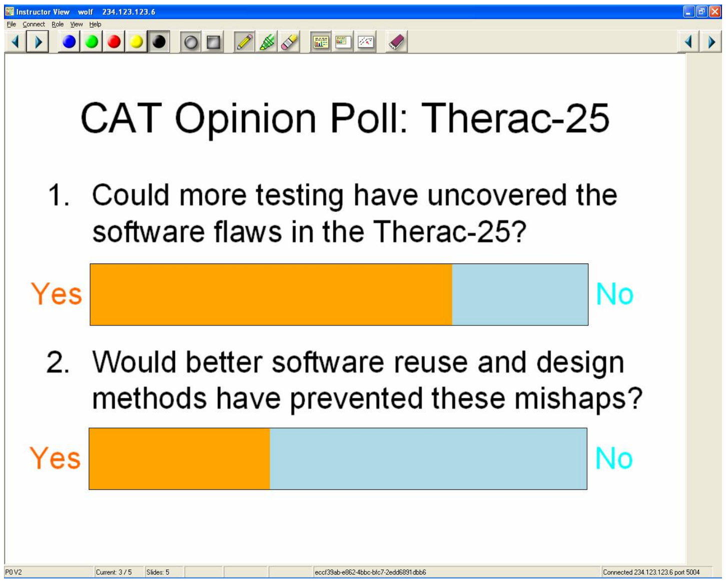
CAT: Opinion Poll – Therac-25

1. Thinking practically, could more testing have uncovered the software flaws in the Therac-25? [✓ Yes] No
2. Again thinking practically, would better software reuse and design methods have prevented these mishaps? [✓ Yes] No
3. How much testing should be done on any computer system that could endanger someone's life? None ○ Some [✓ ○] Complete

P0 V1 | Current: 2 / 5 | Slides: 5 | eccf39ab-e862-4bbc-bfc7-2edd6891dbb6 | Connected 234.123.123.6 port 5004

Figure 5.13: The instructor's view of the exercise input slide from the presentation of Appendix Section A.3. (Note, the presentation has never been used in a class, and all data in Figures 5.13–5.15 is for demonstration only.)

components, called "widgets". These widgets include inputs like multiple choice selections, text inputs, and drawing areas and displays like shaded bars for histograms, line graphs, and text boxes. Appendix Section A.3 shows a presentation that builds a complex exercise from simple SIP widgets. For convenience, several slides from that presentation are included here. Figure 5.13 shows a set of three multiple choice questions on one slide. The bar across the top, which is visible only to the instructor, summarizes the response rate across all three questions. In the example shown, most students have submitted responses. Figures 5.14 and 5.15 show two different summaries the instructor designed, each of which aggregates responses to the first two multiple choice questions differently. The instructor carefully crafted this idiosyncratic exercise by combining static PowerPoint objects with just three types of SIP widget: the multiple choice input widget, the fill box display widget (for the instructor's progress bar and the bars in Figure 5.14), and the shaded display widget (for the grey boxes in Figure 5.15).


Instructor View wolf 234.123.123.6
File Connect Role View Help
P0 V2
Current: 3 / 5
Slides: 5
eccf39ab-e862-4bbc-bfc7-2edd6891dbb6
Connected 234.123.123.6 port 5004

CAT Opinion Poll: Therac-25

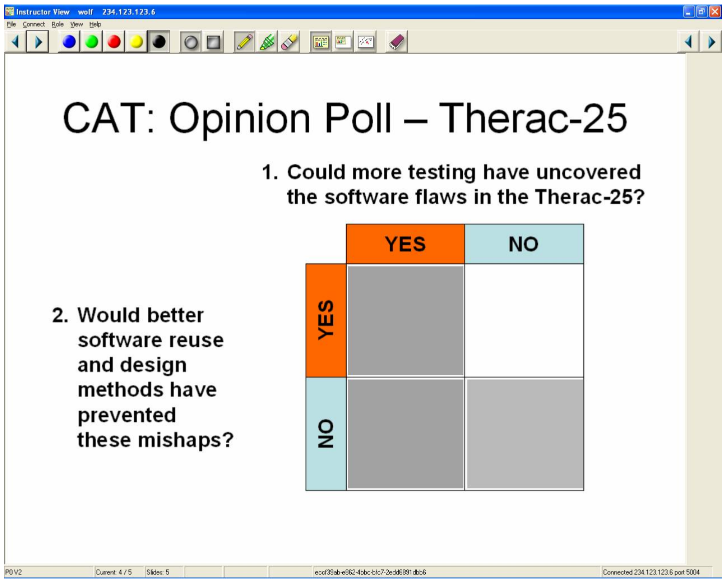
1. Could more testing have uncovered the software flaws in the Therac-25?

Yes No

2. Would better software reuse and design methods have prevented these mishaps?

Yes No

Figure 5.14: A slide summarizing the first two questions from Figure 5.13. More students answered “Yes” to question #1 than to question #2.


Instructor View wolf 234.123.123.6
File Connect Role View Help
P0 V2
Current: 4 / 5
Slides: 5
eccf39ab-e862-4bbc-bfc7-2edd6891dbb6
Connected 234.123.123.6 port 5004

CAT: Opinion Poll – Therac-25

1. Could more testing have uncovered the software flaws in the Therac-25?

2. Would better software reuse and design methods have prevented these mishaps?

	YES	NO
YES		
NO		

Figure 5.15: An alternate summary of the first two questions from Figure 5.13. This summary shows different data than Figure 5.14. For example, this summary shows that few or no students answered “No” to question #1 and “Yes” to question #2.

Like the ClassTalk System [37], SIP also supports innovative exercises by making a wide range of input modalities available for exercises. SIP includes input widgets for typed or inked text, multiple choice selection, numerical inputs, inked graphs and other drawings, and list selection.

Finally, interaction designers can extend SIP with new widgets. Each widget need only extend a standard base class, call a standard constructor in that base class, and access SIP properties through a simple API. Furthermore, SIP's suite of tools includes a testing environment for new widgets (described in Section 5.4.3). Making a new widget available to a user requires uploading it to her SIP database, but the testing environment also scaffolds this process. We envision SIP widgets eventually being even easier to plug into existing SIP installations, *e.g.*, by referencing a web-based source for the new widget.

5.3.4 Support Formative Assessment

SIP was originally conceived as an electronic scaffold for a set of formative assessment techniques called Classroom Assessment Techniques (CATs) [13]. These are short, ungraded, in-class assessments designed to help students and instructor gauge students' learning progress. CATs use a wide variety of different exercise styles to support different assessment goals such as short answer questions, diagramming and list construction. Instructors using CATs are encouraged to combine and tailor elements of different exercise structures to accommodate their particular goals.

SIP supports CATs and other formative assessments by providing lightweight, flexible interaction mechanisms and rapid aggregation of the resulting data. Because interactive exercises in SIP are built into the slides, transitions to SIP CATs are quick and easy. Furthermore, SIP's broad range of input and display widgets allow instructors to customize their exercises to meet their particular goals. Appendix Sections A.2 and A.3 show two SIP exercises based on CATs [13]. (Appendix Section A.3's exercise is partially reproduced in Figures 5.13–5.15.) These simple exercises demonstrate the detailed control of layout and display available to instructors in SIP. In particular, Figures 5.14 and 5.15 show two very different visualizations of student input on the same pair of questions from a CAT exercise. The exercise's designer wanted to use these different visualizations to expose the links between students' subjective perceptions of different software engineering concepts.

For CATs that cannot be represented by SIP's existing interaction widgets, instructional developers can extend SIP with new widgets. For example, Section 5.2.3 describes an innovative way to quickly aggregate free text input from students by having students judge the relationships among their fellow students' responses. All but one of the widgets used for this interactive exercise are standard SIP widgets: lists, multiple choice questions, text inputs, and text displays. However, one additional widget, custom-designed just for this type of exercise, was added to group and rank responses based on the results of students' comparisons.

5.3.5 Scale to Large Classes

SIP promises particular value in overcoming the logistical obstacles involved in scaling interaction to large classes. Some classroom exercises are most effective if their results can be presented and discussed immediately. However, in a large class, the overhead of passing out and collecting paper materials is substantial, and evaluating results from pen-and-paper exercises in real-time is impossible. Another problem for activities in large classes is ensuring and monitoring participation. Networked electronic communication and software aggregation can overcome many of these problems. One of our key concerns in SIP was addressing some of the scaling issues less amenable to computational solutions. For example, how can a computer-mediated communication system aggregate free-form text or ink input? ClassTalk cleverly addresses this problem for simple inputs by binning according to containment of a predefined set of keywords [37]. ActiveClass facilitates students' own organization of free-form responses by allowing students to extend multiple choice questions with their own new responses or respond using the existing selection [103]. SIP uses the flexible "distributed human computation" approach, described in Section 5.2.3.

5.3.6 Provide an Adoption Path

As with Presenter and the Classroom Feedback System, SIP was designed to provide a natural adoption path leading classes from passive lectures toward active learning. SIP accomplishes this by building atop Microsoft PowerPoint, a widely-adopted presentation technology. Instructors already using PowerPoint use their familiar presentation design environment, enhanced now with facilities to create interactive presentations.

Within PowerPoint, SIP further eases adoption by relying on familiar mechanisms for design of interactive exercises like “cutting and pasting” interactive widgets and alignment tools. Section 5.4.1 gives a detailed example of how these techniques help create complex interactive exercises out of simple components.

Unfortunately, the prototype SIP implementation falls short of providing an easy adoption path for instructors in several ways. While SIP presentations are designed inside PowerPoint, an instructor must adopt Presenter to execute those presentations. Furthermore, SIP requires a separate database server to store student data, requiring yet another application for adoption. SIP also assumes that students have their own computers. Imagining classrooms ten years from now, we believe this is a reasonable assumption, but students may not have those devices in today’s classrooms.

Finally, as mentioned with Presenter, having a reliable fallback is critical for adoption of an experimental presentation technology. Instructors’ and students’ classroom time is too valuable and scarce to consume the time with technical problems. Therefore, a rapid and effective plan must be available in case the system fails unexpectedly. Instructors can, theoretically, fallback to Presenter or PowerPoint if SIP fails, but neither of these support SIP’s interactive aspects. Therefore, we instead made killing and restarting SIP after a failure as easy and quick as possible. Still, the lack of an independent, full-featured fallback reduces SIP’s “trialability” [86] (see Section 3.5) and therefore its ease of adoption.

5.3.7 Integrate with Assessment

A long-term goal for SIP is to integrate smoothly with assessment processes in the classroom. SIP's current design takes a crucial first step toward this goal by persistently storing all student contributions, including the history of revisions as students resubmit responses to a single exercise. Eventually, SIP must include simple mechanisms for manipulating this data and transferring it, through standardized educational formats, to related software such as grade databases. With some effort, this kind of data manipulation is already possible by directly accessing SIP's underlying database. Indeed, exactly this style of *post hoc* access was used to analyze student data from the experimental class described in Section 5.5. During the experimental class, persistent storage of student data also allowed students to see comparisons of their responses across two exercises, one at the start of class and one at the end. Similarly, an instructor could design exercises that compare data from the beginning and end of a term or across a series of course offerings.

5.3.8 Design Shortcomings

SIP's current design hews to many of our design goals and shows the strength of integrating interaction execution and design into the mediating slides; however, SIP's current implementation also falls short of our design goals in two critical ways. This section describes SIP's most important shortcomings and discusses the potential for addressing these problems in the future.

Easy interaction specification

The single most important missing piece in the current SIP implementation is a mechanism for specifying interactions that is accessible to a broad spectrum of instructors. Currently, designing an interactive exercise from scratch requires some SQL programming (to link widgets' input properties to other widgets' output properties). While SIP's underlying machinery leverages a small amount of SQL into rich interactive behavior (as described in

Section 5.4.4), even that small amount of code is beyond the skills of most instructors and probably beyond the *commitment level* of all of them.

Copying-and-pasting exercises provides a partial solution. Instructors can copy exercises, including the SQL property queries, from one presentation to another. Instructors can even accomplish substantial modifications to the exercises by changing only the data (rather than the code) of those exercises, drastically altering the exercise's appearance — *e.g.*, multiple choice question or response text, exercise layout, graph colors, or the timing of the exercise within the presentation. However, copy-and-paste alone will not fully satisfy instructors' need to create innovative exercises that respond to their individual needs.

Empowering instructors to specify their own interactive exercises is essentially an "end-user programming" problem [34]. The end-user programming literature presents an array of solutions to this problem — *e.g.*, "wizards", visual programming, and programming by demonstration — with different tradeoffs in expressiveness, ease of use, and design and implementation difficulty. Several of these solutions might be appropriate in SIP. The most promising first approach is a simple, visual interface for linking input properties directly to the output properties that drive them, enabling users to accomplish the most common SIP tasks painlessly.

A related, and less well-understood, problem is helping end-users *debug* the programs they create. In SIP's current incarnation, debugging is already a challenge because the "programs" that link SIP properties are designed for use by full classes of students. Testing such programs as a single instructor on a single computer, which may not even be a Tablet PC, is a challenge. In Presenter, we supported single-computer debugging of the application by simulating some aspects of the distributed application on the single computer. This strategy faces greater challenges in SIP because of the complexity of the data the distributed clients generate (*i.e.*, student input). An end-user programming interface will compound this problem as users may not understand their own programs or even be aware that they are creating programs.

Meta-interaction management

The second key weakness in SIP's current implementation is lack of support for management of classroom processes above the level of individual interactive exercises. SIP currently focuses on enabling effective design and use of individual exercises in class. However, a full-fledged classroom interaction system must also address a variety of critical meta-interaction issues including management of student groups, reliable and secure access control, integration with grading and other external applications, and management of course flow across class sessions. Such capabilities have been explored in a variety of other systems. Indeed, incorporating SIP into a framework like the WISE System [65] or building atop a language like the ClassSync Modeling Language [21] are natural future directions for adding such functionality. SIP's database already exposes the data needed to address these meta-interaction issues.

SIP also lacks support for a final meta-interaction issue not present in many other systems: "interaction policies" that users can understand, specify, customize, and rely on. Management of anonymity in formative assessments illustrates the need for such support. In a pen-and-paper formative assessment, an instructor specifies a policy such as "write your response on a sheet of scratch paper without your name on it". Students understand the intent of this anonymity policy, the mechanisms supporting the policy, and the extent to which their anonymity is protected (*i.e.*, the extent to which their handwriting is recognizable within the class group). A student can even easily and privately opt out of the policy by writing her name on her response. In a computer-mediated communication system, the instructor might want to specify various different levels of anonymity — *e.g.*, fully anonymous, anonymous within project groups, or anonymous but with a record of *whether* each student submitted. More importantly, the instructor and students will want to clearly understand the anonymity policy and the mechanisms that enforce that policy.

Our experience with the Classroom Feedback System (CFS) and SIP have already provided a slew of examples where students mistrusted or misunderstood anonymity policies.

In CFS, one student commented about being concerned that anonymity could be compromised by students looking over each others' shoulders. In SIP, several students commented about how their anonymity in the SIP System made them more comfortable responding to SIP exercises. However, identity *was* actually tied to responses in SIP (in the form of computer IDs that were publicly displayed on each device) and *could have* been used in displays but were not. Students assumed they were anonymous when the instructor neither told students that these IDs would be unused nor that they could be used.

Anonymity is in the eye of the beholder, and a principled anonymity policy will require careful and transparent design. Other interaction policies will be similarly complex to manage. Designing systems that allow effective specification and clearly comprehensible publication of policies is a rich area of future work. One potential starting point is value-sensitive design [46].

5.4 SIP Architecture

This section describes the SIP prototype implementation's architecture in detail along with description of some aspects that (as noted) have not yet been implemented. The Structured Interaction Presentation (SIP) System extends the Classroom Presenter System for slide-based presentation (see Chapter 3) to support the design and execution of interactive exercises. Instructors can design SIP presentations on any .NET-enabled computer with PowerPoint and access to a SQL database server. In the classroom, the instructor uses a wirelessly networked Tablet PC to manage a SIP presentation, as with Classroom Presenter. Students can connect to the system through any networked device with .NET support, although using Tablet PCs allows full participation in ink-based exercises. Again, all participants must be able to access a shared SQL server over the network.

Our vision for SIP is a system that enables instructors to design and orchestrate interaction across the class through the medium of the slides. To fulfill this vision, we have constructed a four-part architecture for SIP as shown in Figure 5.16. Instructors create their

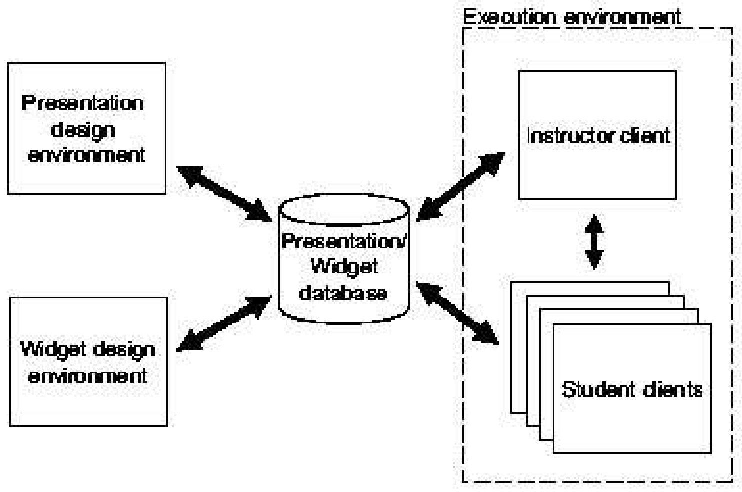

Execution environment
Presentation design environment
Widget design environment
Presentation/ Widget database
Instructor client
Student clients

Figure 5.16: SIP architecture, including environments for authoring presentations, authoring widgets (interactive slide elements), and executing presentations as well as a database for storing SIP presentations and data.

presentations in the presentation design environment, an add-in to on Microsoft PowerPoint orthogonal to Classroom Presenter's add-in (see Section 3.3.3). Besides laying out static elements (*e.g.*, lines, text boxes, and circles) as in normal PowerPoint, instructors can also lay out and configure "widgets" — elements with interactive semantics representing input opportunities for students and displays that aggregate or summarize student input. SIP's presentation execution environment extends Classroom Presenter. The instructor uses an enhanced Presenter client to navigate, annotate, and otherwise control the presentation while students use the enhanced Presenter clients to complete interactive exercises. Besides displaying slides and ink, the enhanced Presenter clients include an extra "layer" (see Section 3.2.4) that displays SIP widgets, accepts user input, and exchanges data with the SIP database server. For designers, SIP includes a tool for developing, testing, and sharing the interactive widgets from which instructors compose interactive SIP exercises. Finally, the database back-end ties these elements together, storing widget type information, completed SIP presentations, and student data.

In this section, I describe each of the first three parts of the architecture in turn. Then, I describe the database back-end and its relationship with each of the other parts.

5.4.1 SIP Design Environment

The SIP design environment is integrated with Microsoft PowerPoint, a traditional slideware system. The instructor uses PowerPoint for editing. When a presentation is ready for execution, a PowerPoint add-in exports the presentation directly to the database that stores SIP presentations during execution. Despite the need for a database during *execution*, the PowerPoint file containing a SIP presentation is fully self-contained for *design*. That is, each SIP PowerPoint file contains all the layout and configuration information for its embedded interactive exercises.

We envision the design process eventually feeling much like design of current, static slideware presentations. Instructors lay out both static and dynamic elements on the slides as they currently lay out static elements (e.g., textboxes, lines, and rectangles). Instructors link dynamic elements together through visual tools, properties forms, and “wizards”.

This model will allow instructors to carefully craft the interactive experience, arranging interactive elements to reflect the timing, appearance, and interactive semantics they imagine. The mixture of static and dynamic elements, broken down into simple components, will also allow construction of specialized configurations that might be unanticipated in a system of “canned” interactive exercises inserted as pre-constructed slides. For example, instructors might create a graph display for student drawings out of simple components: a SIP digital ink display, two arrows representing the axes, and textboxes for axis labels.

The SIP prototype approaches this goal. Instructors lay out interactive elements, called widgets, as textbox placeholders. Each widget is a bundle of data and functionality including: a simple PowerPoint representation (i.e., appropriate text for the placeholder); a collection of input and output properties that define how the widget can be linked to other widgets, such as a “text” output property or a “font size” input property for a text input widget; and a reference to code that will render the widget in the SIP execution environment.

SIP widgets are stored as textboxes with PowerPoint meta-data describing their properties. Because the meta-data format is native to PowerPoint, SIP presentation files are standard PowerPoint files and can be saved and edited even by instances of PowerPoint that lack the SIP add-in. Our decision to make SIP presentations standard PowerPoint files editable without the SIP add-in reflects the belief that instructors will be more comfortable adopting new technologies that "degrade gracefully" to technologies with which they are familiar and of which they are confident. (See Section 3.5 for more discussion on this point.)

Eventually, property editing in SIP will conform to standard PowerPoint interface mechanisms. For now, the properties panel exposes the underlying SQL queries that encode the connections among properties (as described in Section 5.4.4). To simplify property editing, these queries may include a set of macros that expand to common fragments of SQL code. These queries allow outputs of one widget (e.g., a text input widget) to control, individually or in aggregate, the inputs of another widget (e.g., a shaded display widget). Each query-controlled property is tagged to run either once or every time one of its inputs changes. Custom properties, such as the "mynumber" property of Figure 5.24 below, allow easy parameterization of queries for related widgets and facilitate copy-and-paste creation and multiple selection editing of related widgets.

Although SIP does not currently support "wizards" for automatically laying out common interaction patterns, it is straightforward to copy and modify existing exercises to construct new ones. Instructors can copy SIP widgets (and all their associated meta-data) across slides or presentations just as they would copy standard, static slide elements.

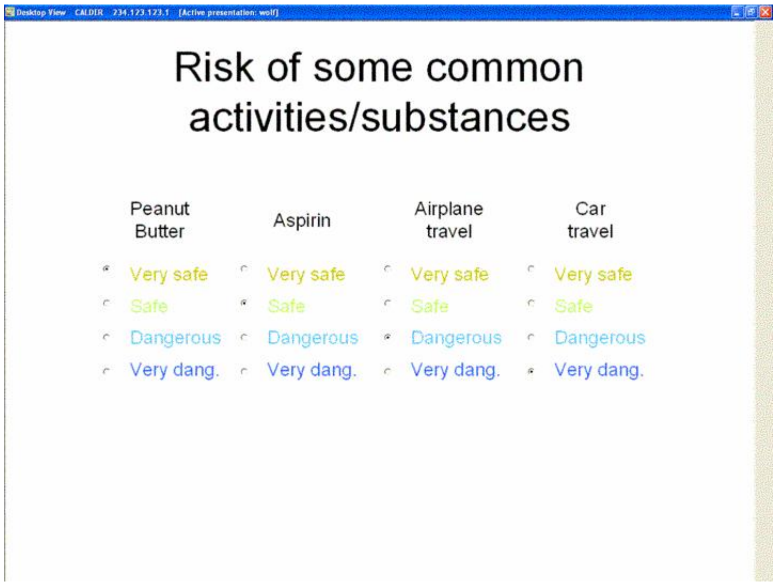
Instructors can also edit multiple widgets' properties at the same time. Selecting more than one widget at once displays just the properties common to all the widgets in the properties panel. The properties panel displays a single common value for those properties that also share the same value across the selected widgets. All other property displays are blank. Editing any property display sets all the selected widgets' values for that property to the resulting text. This multiple selection editing style makes it easy to construct large interactive exercises out of structured arrangements of similar widgets.

SIP, like Presenter, supports separate views for students, instructor, and the public display. This allows instructors to restrict the visibility of both static elements and dynamic elements. Restricting visibility is even more important in SIP than in Presenter since many interactive slide elements and their related static elements will only be appropriate to certain roles. For example, instructions explaining how to perform an exercise might be restricted to the student display, or status bars tracking student progress might be restricted to the instructor's display.

Figures 5.17–5.24 demonstrate the use of custom widget properties, copy-and-paste, and multiple selection editing. Figures 5.17 and 5.18 show the prompt and result slides of an exercise in which students assess the risk of various activities and substances. Figures 5.19–5.24 show the design process behind the results slide.

In Figure 5.19, the instructor has just selected a widget — of the shaded display widget type — to add to the presentation from the upper left toolbar. The design environment automatically polls the underlying database for the set of available widgets and arranges them in toolbars at the top of the figure. When the user selects one of these widgets, the add-in positions a textbox placeholder on the current slide, labeled with the widget type. The instructor can move, resize, cut, copy, or paste the placeholder as with any other slide element. In Figure 5.20, the instructor has moved the widget into its place in the first row and column of the results slide.

When the instructor selects the widget placeholder (as in Figure 5.20), the SIP properties panel appears and displays the properties specific to the widget. The properties panel allows editing of property contents (including query-based links to other widgets' properties that define the interactive semantics of the widget). In this case, the instructor fills out all the properties of the widget, as shown in Figure 5.21. The instructor changed the widget's name to "modern-display-1" (the first display of results from the "modern risk" exercise). The widget will interpolate (shade) between its "zerocolor" (white) and its "onecolor" (black). There will be only one instance of the widget per presentation, as indicated



Desktop View CALDIR 234.123.123.1 [Active presentation: wolf]

Risk of some common activities/substances

Peanut Butter	Aspirin	Airplane travel	Car travel
[✓ Very safe]	Very safe	Very safe	Very safe
Safe	[✓ Safe]	Safe	Safe
Dangerous	Dangerous	[✓ Dangerous]	Dangerous
Very dang.	Very dang.	Very dang.	[✓ Very dang.]

Figure 5.17: The prompt from an exercise in the experimental SIP class. There are four multiple choice questions (one per column), each with four responses.

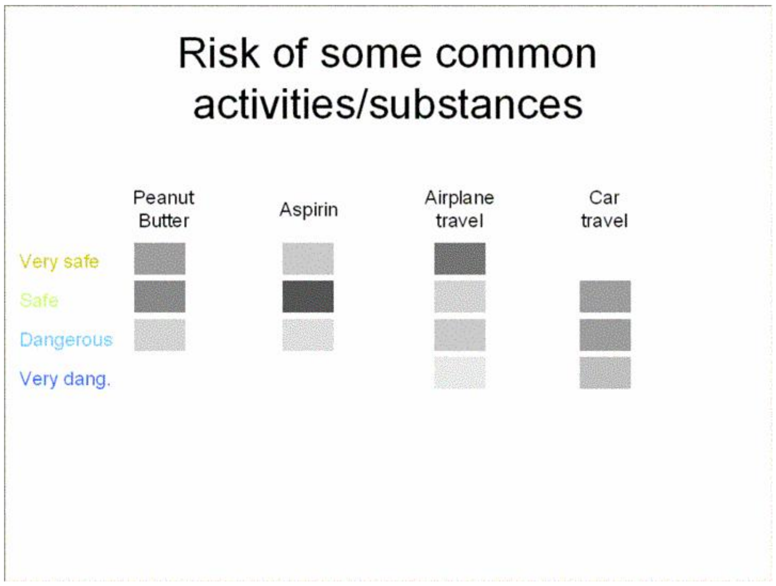
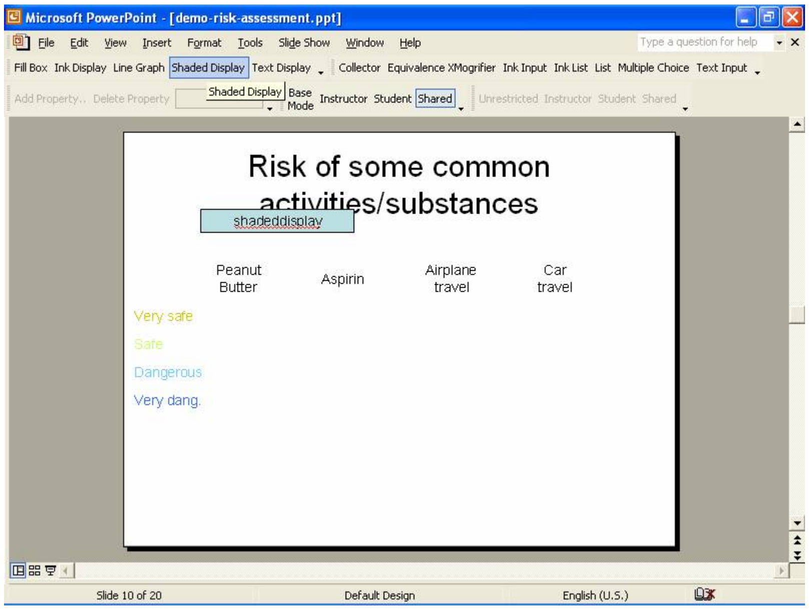


Figure 5.18: The results from the exercise of Figure 5.17 (with actual data from the experimental SIP class. A darker shaded box means more students responded with that box's row's answer to that box's column's question.

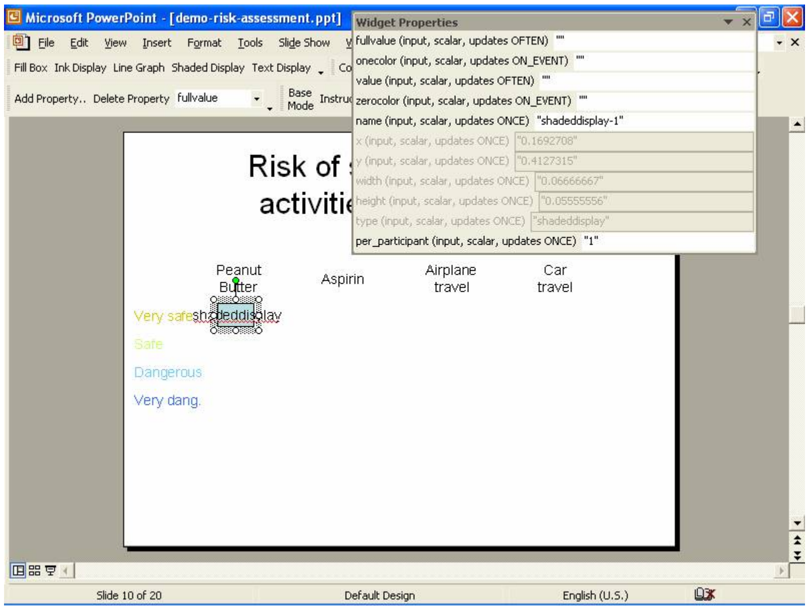

Microsoft PowerPoint - [demo-risk-assessment.ppt]
File Edit View Insert Format Tools Slide Show Window Help
Fill Box Ink Display Line Graph Shaded Display Text Display
Collector Equivalence XMogrifier Ink Input Ink List List Multiple Choice Text Input
Add Property.. Delete Property Shaded Display Base Mode Instructor Student Shared
Unrestricted Instructor Student Shared
Slide 10 of 20
Default Design
English (U.S.)

Risk of some common activities/substances

shadeddisplay

	Peanut Butter	Aspirin	Airplane travel	Car travel
Very safe				
Safe				
Dangerous				
Very dang.				

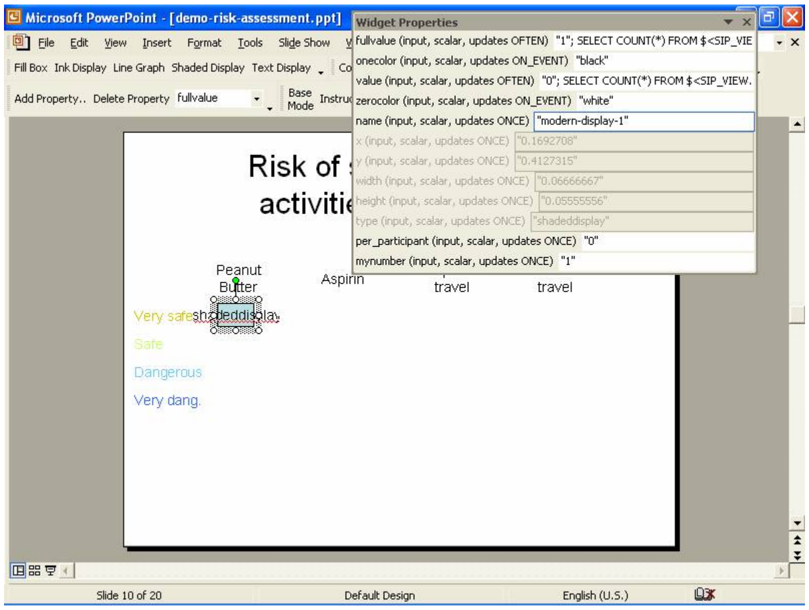
Figure 5.19: The instructor begins constructing the results slide of Figure 5.18 by placing a shaded display widget.


Microsoft PowerPoint - [demo-risk-assessment.ppt]
File Edit View Insert Format Tools Slide Show
Fill Box Ink Display Line Graph Shaded Display Text Display
Add Property.. Delete Property fullvalue Base Mode
Slide 10 of 20
Default Design
English (U.S.)

Widget Properties

fullvalue (input, scalar, updates OFTEN) ""
onecolor (input, scalar, updates ON_EVENT) ""
value (input, scalar, updates OFTEN) ""
zerocolor (input, scalar, updates ON_EVENT) ""
name (input, scalar, updates ONCE) "shadeddisplay-1"
x (input, scalar, updates ONCE) "0.1692708"
y (input, scalar, updates ONCE) "0.4127315"
width (input, scalar, updates ONCE) "0.06666667"
height (input, scalar, updates ONCE) "0.05555556"
type (input, scalar, updates ONCE) "shadeddisplay"
per_participant (input, scalar, updates ONCE) "1"

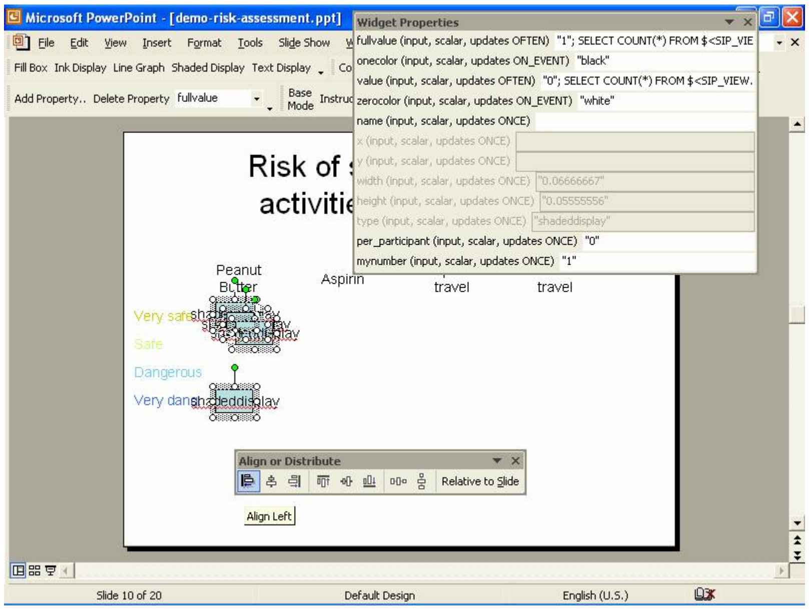
Figure 5.20: The instructor has selected, moved, and resized the shaded display widget of Figure 5.19.


Microsoft PowerPoint - [demo-risk-assessment.ppt]
File Edit View Insert Format Tools Slide Show
Fill Box Ink Display Line Graph Shaded Display Text Display
Add Property.. Delete Property fullvalue
Base Mode
Risk of
activitie
Peanut Butter
Aspirin
travel
travel
Very safe
Safe
Dangerous
Very dang.
Slide 10 of 20
Default Design
English (U.S.)

Widget Properties

fullvalue (input, scalar, updates OFTEN)	"1"; SELECT COUNT(*) FROM $<SIP_VIE
onecolor (input, scalar, updates ON_EVENT)	"black"
value (input, scalar, updates OFTEN)	"0"; SELECT COUNT(*) FROM $<SIP_VIEW.
zerocolor (input, scalar, updates ON_EVENT)	"white"
name (input, scalar, updates ONCE)	"modern-display-1"
x (input, scalar, updates ONCE)	"0.1692708"
y (input, scalar, updates ONCE)	"0.4127315"
width (input, scalar, updates ONCE)	"0.06666667"
height (input, scalar, updates ONCE)	"0.05555556"
type (input, scalar, updates ONCE)	"shadeddisplay"
per_participant (input, scalar, updates ONCE)	"0"
mynumber (input, scalar, updates ONCE)	"1"

Figure 5.21: The instructor has configured the widget properties of the widget from Figure 5.20.


Microsoft PowerPoint - [demo-risk-assessment.ppt]
File Edit View Insert Format Tools Slide Show
Fill Box Ink Display Line Graph Shaded Display Text Display
Add Property.. Delete Property fullvalue
Base Mode
Risk of
activitie
Peanut Butter
Aspirin
travel
travel
Very safe
Safe
Dangerous
Very dang
Align or Distribute
Relative to Slide
Align Left
Slide 10 of 20
Default Design
English (U.S.)

Widget Properties

fullvalue (input, scalar, updates OFTEN)	"1"; SELECT COUNT(*) FROM $<SIP_VIE
onecolor (input, scalar, updates ON_EVENT)	"black"
value (input, scalar, updates OFTEN)	"0"; SELECT COUNT(*) FROM $<SIP_VIEW.
zerocolor (input, scalar, updates ON_EVENT)	"white"
name (input, scalar, updates ONCE)	
x (input, scalar, updates ONCE)	
y (input, scalar, updates ONCE)	
width (input, scalar, updates ONCE)	"0.06666667"
height (input, scalar, updates ONCE)	"0.05555556"
type (input, scalar, updates ONCE)	"shadeddisplay"
per_participant (input, scalar, updates ONCE)	"0"
mynumber (input, scalar, updates ONCE)	"1"

Figure 5.22: The instructor is in the process of creating a column of four widgets from the widget in Figure 5.21. He first made three copies of the widget, then moved the bottom copy into position. Now, he is aligning the widgets, and next, he will distribute them. All of these are standard PowerPoint operations.

by the "0" ("false") for the "per_participant" property. The instructor has also added a new property called "mynumber" (using the `Add Property..` button on the lower left toolbar at the top of Figure 5.21). As described below, this property indicates which response number the widget displays (of the four responses to each question in Figure 5.18).

The shade of any shaded display widget is determined by the ratio between its "value" and "fullvalue" properties. In this case, both properties use modified SQL queries to connect to the results of the exercise of Figure 5.18.[4] The widget's "fullvalue" is calculated with the query:

```
SELECT COUNT(*)
FROM $<SIP_VIEW.modern-mc-1.responsenumber> t
WHERE sip_responsenumber NOT LIKE ''
```

In other words, total all the responses to the first multiple choice question, leaving out students who chose no response. The "value" query is similar except that it counts only responses that match the "mynumber" property:

```
SELECT COUNT(*)
FROM $<SIP_VIEW.modern-mc-1.responsenumber> t
WHERE sip_responsenumber LIKE '$<mynumber>'
```

Although these queries may look daunting, they are mostly standard boilerplate and can be copied from other, completed exercises.

The instructor then copies this widget three times and uses the standard PowerPoint alignment and distribution tools to line them up as he wants them. (As the instructor copies the widgets, the SIP add-in automatically renames them in numerical sequence to avoid name conflicts.) Figure 5.22 shows the alignment step. Finally, he copies the whole column of four widgets three more times to create the array of widgets he wants, as in Figure 5.23. Finally, the instructor selects each row of four in turn, setting the "mynumber" property

[4]Section 5.4.4 describes the form of SIP property queries in detail.

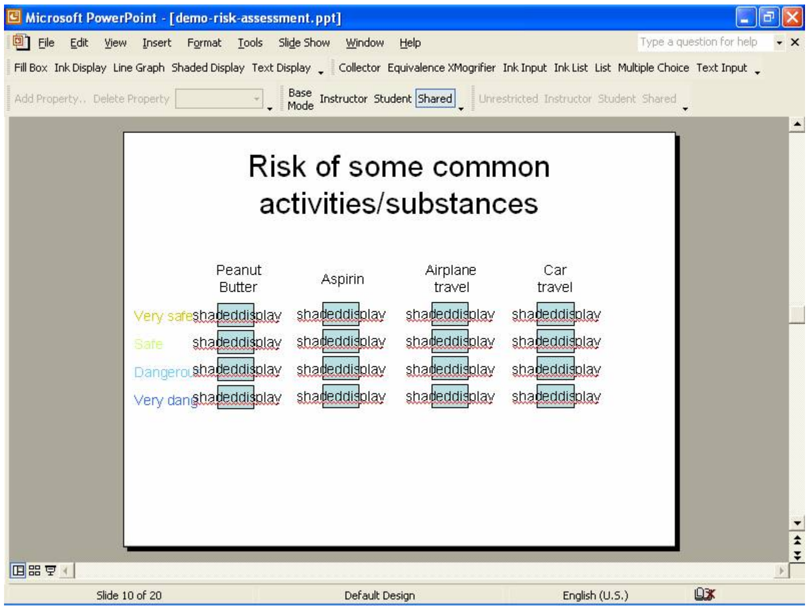
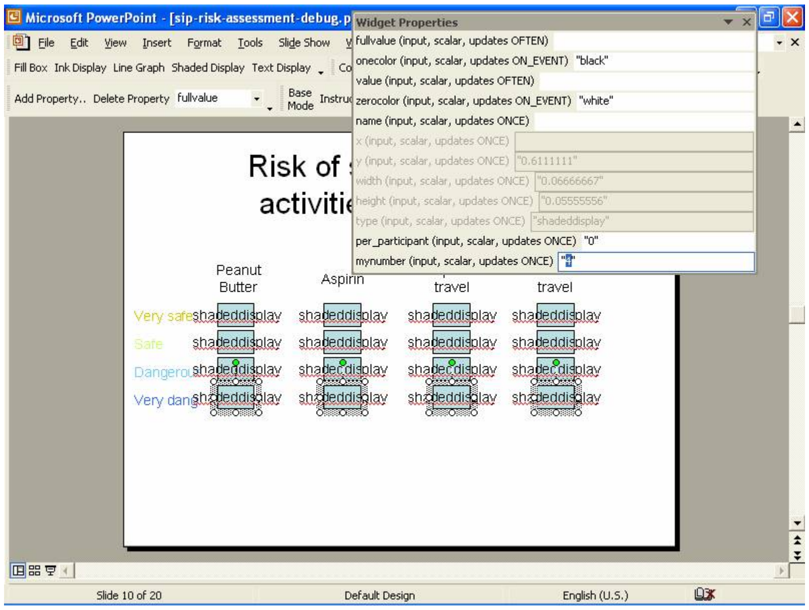

Microsoft PowerPoint - [demo-risk-assessment.ppt]
File Edit View Insert Format Tools Slide Show Window Help
Fill Box Ink Display Line Graph Shaded Display Text Display
Collector Equivalence XMogrifier Ink Input Ink List List Multiple Choice Text Input
Add Property.. Delete Property
Base Mode Instructor Student Shared
Unrestricted Instructor Student Shared
Risk of some common activities/substances
Peanut Butter
Aspirin
Airplane travel
Car travel
Very safe
Safe
Dangerous
Very dangerous
shadeddisplay
Slide 10 of 20
Default Design
English (U.S.)

Figure 5.23: The instructor has laid out all the widgets for the slide in Figure 5.18. However, they have not been configured to summarize the correct question (for each column) and response (for each row).


Microsoft PowerPoint - [sip-risk-assessment-debug.p
File Edit View Insert Format Tools Slide Show
Fill Box Ink Display Line Graph Shaded Display Text Display
Add Property.. Delete Property fullvalue
Base Mode
Risk of
activiti
Peanut Butter
Aspirin
travel
travel
Very safe
Safe
Dangerous
Very dangerous
shadeddisplay
Slide 10 of 20
Default Design
English (U.S.)

Widget Properties

fullvalue (input, scalar, updates OFTEN)	
onecolor (input, scalar, updates ON_EVENT)	"black"
value (input, scalar, updates OFTEN)	
zerocolor (input, scalar, updates ON_EVENT)	"white"
name (input, scalar, updates ONCE)	
x (input, scalar, updates ONCE)	
y (input, scalar, updates ONCE)	"0.6111111"
width (input, scalar, updates ONCE)	"0.06666667"
height (input, scalar, updates ONCE)	"0.05555556"
type (input, scalar, updates ONCE)	"shadeddisplay"
per_participant (input, scalar, updates ONCE)	"0"
mynumber (input, scalar, updates ONCE)	"4"

Figure 5.24: The instructor configures one of the rows to summarize the correct response number by setting the “mynumber” property of all the widgets in the row to “4”. The instructor will configure each row and column similarly.

to “1”, “2”, “3”, and “4” down the line. Similarly, he selected each column in turn and set which multiple choice question its queries referred to. Figure 5.24 shows the instructor setting the fourth row’s number to “4”.

When an instructor is ready to execute a presentation, she exports the presentation to the SIP database. The SIP add-in exports the presentation using an augmented version of Classroom Presenter’s format. The slides themselves are standard Presenter slides; SIP hides the widget placeholders and then uses the Presenter add-in to generate “clean” slides. SIP separately generates and includes meta-data descriptions of the widgets in the presentation file. The add-in exports this file to the database. Usually, the instructor will also choose to create a presentation *instance* (representing a particular execution of the presentation with its own set of student data) at the same time. The add-in instantiates the widget descriptions in the instance in the format described in Section 5.4.4. The execution environment can then render the Presenter slides with SIP widgets overlaid, configured with current data from the database.

5.4.2 SIP Execution Environment

SIP presentations are executed as a layer on top of the Classroom Presenter System, described in detail in Chapter 3. Briefly, Presenter is a presentation system that runs on a Tablet PC, a pen-based mobile computer, and allows the instructor to handwrite over and navigate through computer-projected slides. Slides and ink are multicast from the instructor’s computer to one or more computers driving public displays.

Theoretically, SIP could have executed as an add-in to PowerPoint, without the need for a separate application and potentially without the need for a database. For many of the same reasons as those described in Section 3.5, it does not. A subtle problem endemic to SIP is the mismatch between SIP’s conception of a “presentation instance” and PowerPoint’s. Consider the fact that many instructors reuse slides from one term to the next. In this context, a “presentation” is the set of slides designed to accompany and augment discussion of a particular topic. A “presentation instance” is the artifact in the form it takes in a given term’s discussion. PowerPoint does distinguish these concepts, but the “presentation instances” (“shows” in PowerPoint) are purely transient and rely on the underlying presentation. In SIP, a presentation instance must contain student data and be persistent (so it can be restarted, *e.g.*, when a presentation spans multiple class periods). Also, even were SIP’s execution environment built into PowerPoint, we would still want the option of an external database to store student data. The external database ensures persistence of the data and supports rich and efficient queries for connecting widget input properties to upstream output properties.

SIP uses Presenter’s distributed architecture and extensible slide rendering to support student involvement in interactive exercises. Student devices synchronize with the instructor device just as the public display devices do in Presenter, by maintaining common ink and navigation state through a multicast network. Each SIP client (student, instructor, or display device) renders the normal Presenter slides but also renders any SIP widgets on the current slide. A widget manager mediates between the widgets and a database which maintains data associated with the widgets’ input and output properties. The manager sends updates to the database as user actions change widget properties and notifies widgets to update themselves when they are affected by changes to widgets linked to them. The data managers communicate with the database through a .NET remoting layer, and both the managers and the remoting layer carefully track and cache data to minimize network communication and recalculation of queries over the database. For now, all communication with the database is unicast because of the individual nature of student input, which therefore would not benefit greatly from a multicast strategy. However, much of the data exchanged between student clients and the database ends up being replicated. Therefore, we envision eventually using a hybrid multicast/unicast approach.

The SIP database records and timestamps every change in a widget input or output property. Because SIP widgets can currently be configured directly with SQL queries, interactive exercises have unrestricted access to this recorded data, even across presentations. Thus, instructors can create exercises that use archived data or track changes in data. For example, an instructor might give an exercise at the start of a term to judge students' knowledge of the course matter, give a similar exercise at the end of the term, and display a slide with display widgets that draw on data from both exercises to support discussion of how students' understanding of the course matter progressed over the term. A growing set of "macros" allows easy construction of the more common SIP queries — such as directly tying one widget's input to the output of another widget in the same presentation — and represents a starting point for defining a more intuitive language to express these queries.

SIP also uses the database to facilitate presentation management. Every SIP presentation is stored as an entry in the database. Students and instructors launch SIP/Presenter and connect to their desired presentation through a simple browser for the presentation meta-data in the database (title, author, current instructor, etc.). Each presentation can also be instantiated multiple times, easing reuse of presentations. Each such presentation instance can be opened and closed multiple times across any length of time, allowing single presentation instances to span multiple class sessions transparently.

The SIP database also supports *post hoc* investigation of class data, e.g., to assign grades based on class participation or visualize data in unplanned ways. Currently, instructors have three tools for this *post hoc* inspection of data: SIP itself (through reopening the presentation instance that generated the data or crafting a new presentation that accesses data from old presentations), the widget authoring environment described below (which provides a light-weight mechanism to place and configure widgets), and direct browsing or querying of the SQL database. Eventually, we envision connecting SIP to course management software that would make this *post hoc* inspection both easier and more powerful. However, as such facilities become an integral part of SIP, we will also need to address issues relating to ownership of data, such as ensuring student anonymity when appropriate.

5.4.3 SIP Widget Authoring Environment

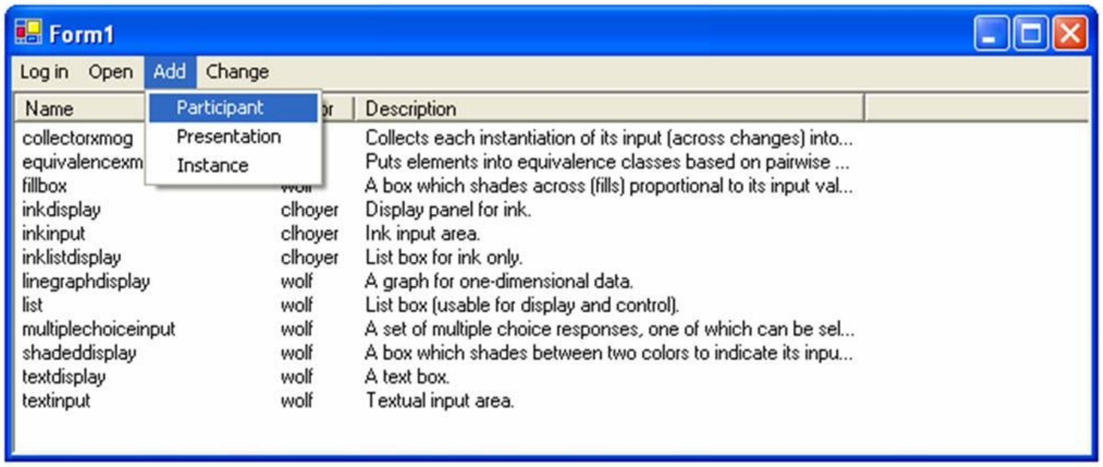
To facilitate the development of new SIP widgets, we have developed a graphical tool that allows both stand-alone — *i.e.*, database- and presentation-free — testing of SIP widgets and database-connected testing of widgets. The widget authoring environment also manages uploading completed widgets to the database for use in designing future presentations. Uploaded widgets are entered into the widget type tables described in Section 5.4.4.

The tool presents a canvas for drawing, moving, resizing, and configuring widgets. It uses .NET reflection to search a target directory for classes matching the widget API and makes any widgets found available in the widget creation window. The widget designer then specifies the width and height of the widget graphically as she would in the PowerPoint SIP design environment. She may then select any widget and observe or manually alter input or output property values of the widget. When the tool is connected to the SIP database, the widget designer can also specify queries to connect widget input and output properties, as in the PowerPoint design environment, and observe propagation of data changes across widgets through these queries.

5.4.4 SIP Database

SIP's database ties together the design, execution, and widget authoring environments described above. SIP uses the database to maintain and manipulate widget type information, completed SIP presentations, and student data. This section begins with a gestalt description of communication with the database and then describes the schemata used in SIP's database and the SIP components closely related to the database, such as the .NET remoting server used by the execution environment to access the database.

Interaction with the database begins with creating an entry for the participant who will author the presentation. A simple application adjunct to Presenter allows both browsing and control of database contents. As shown in Figure 5.25, the manager of the SIP database adds new participants using this application. Next, the presentation author designs a new SIP presentation, as described in Section 5.4.1. The presentation author then exports the presentation to the database, by first logging into the database and then specifying a title and "venue" (multicast address) for the presentation, as indicated in Figures 5.26–5.28. (Technically, the multicast venue is used for the presentation *instance*, not the presentation.



Name	or	Description
collectorxmog		Collects each instantiation of its input (across changes) into...
equivalencexm		Puts elements into equivalence classes based on pairwise ...
fillbox	wolf	A box which shades across (fills) proportional to its input val...
inkdisplay	clhoyer	Display panel for ink.
inkinput	clhoyer	Ink input area.
inklistdisplay	clhoyer	List box for ink only.
linegraphdisplay	wolf	A graph for one-dimensional data.
list	wolf	List box (usable for display and control).
multiplechoiceinput	wolf	A set of multiple choice responses, one of which can be sel...
shadeddisplay	wolf	A box which shades between two colors to indicate its inpu...
textdisplay	wolf	A text box.
textinput	wolf	Textual input area.

Figure 5.25: The SIP database manager uses the database browsing tool to add a new participant. The manager will next pick a username and password for the new participant. The window's content is a list of widget types (called up by the database manager before selecting the Add menu).

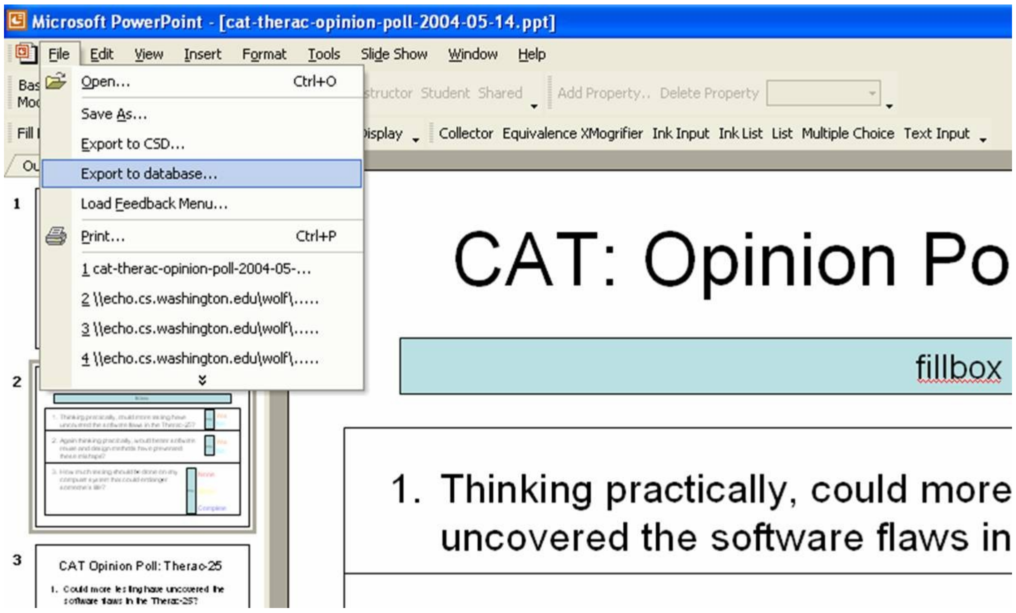
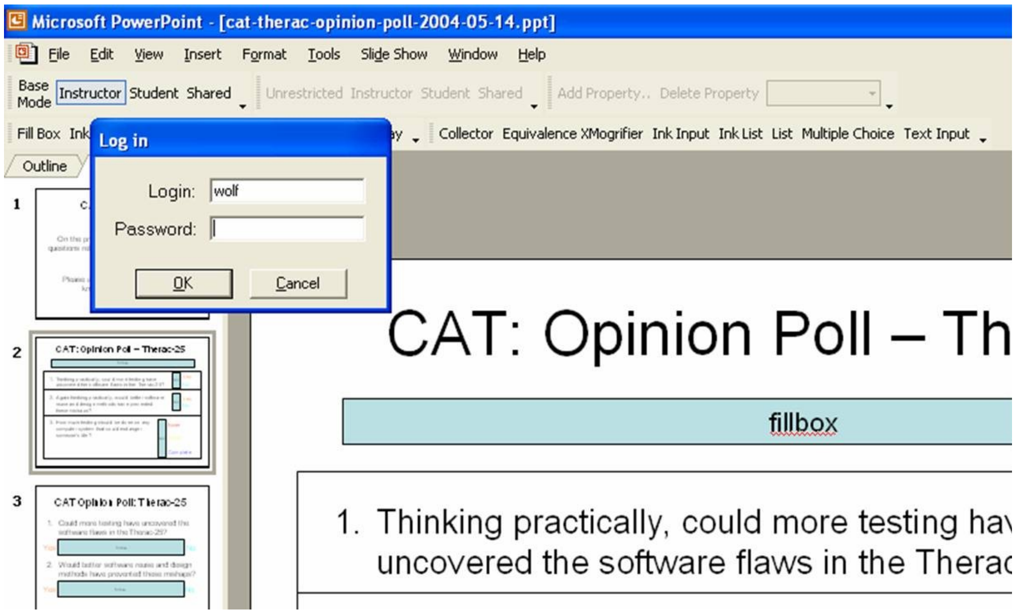

Microsoft PowerPoint - [cat-therac-opinion-poll-2004-05-14.ppt]
File
Edit
View
Insert
Format
Tools
Slide Show
Window
Help
Open... Ctrl+O
Save As...
Export to CSD...
Export to database...
Load Feedback Menu...
Print... Ctrl+P
1 cat-therac-opinion-poll-2004-05-...
2 \\echo.cs.washington.edu\wolf\.....
3 \\echo.cs.washington.edu\wolf\.....
4 \\echo.cs.washington.edu\wolf\.....
structor Student Shared
Add Property.. Delete Property
isplay
Collector Equivalence XMogrifier Ink Input Ink List List Multiple Choice Text Input
CAT: Opinion Po
fillbox
1. Thinking practically, could more
uncovered the software flaws in
CAT Opinion Poll: Therac-25

Figure 5.26: A presentation author selects "Export to database" to begin uploading a presentation to the SIP database.


File Edit View Insert Format Tools Slide Show Window Help
Base Mode Instructor Student Shared
Unrestricted Instructor Student Shared
Add Property.. Delete Property
Fill Box Ink
Collector Equivalence XMogrifier Ink Input Ink List List Multiple Choice Text Input
Outline
OK
Cancel

Microsoft PowerPoint - [cat-therac-opinion-poll-2004-05-14.ppt]

Log in

Login: wolf

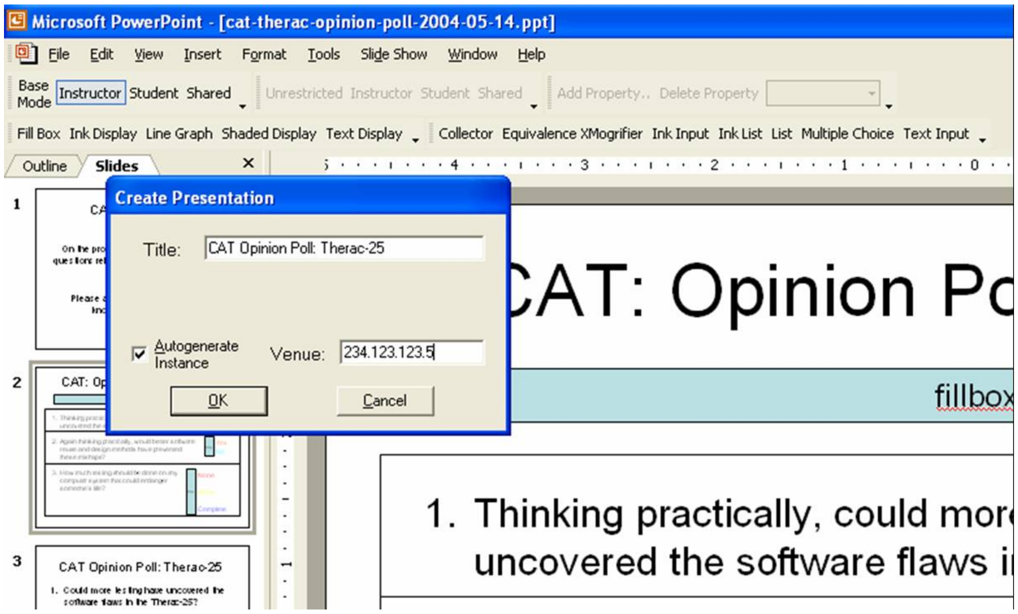
Password:

CAT: Opinion Poll – Th

fillbox

1. Thinking practically, could more testing hav
uncovered the software flaws in the Therac

Figure 5.27: A presentation author logs into the SIP database to upload the presentation from Figure 5.26.


File Edit View Insert Format Tools Slide Show Window Help
Base Mode Instructor Student Shared
Unrestricted Instructor Student Shared
Add Property.. Delete Property
Fill Box Ink Display Line Graph Shaded Display Text Display
Collector Equivalence XMogrifier Ink Input Ink List List Multiple Choice Text Input
Outline Slides
OK
Cancel

Microsoft PowerPoint - [cat-therac-opinion-poll-2004-05-14.ppt]

Create Presentation

Title: CAT Opinion Poll: Therac-25

[✓] Autogenerate Instance

Venue: 234.123.123.5

CAT Opinion Poll: Therac-25

1. Could more testing have uncovered the software flaws in the Therac-25?

CAT: Opinion Po

fillbox

1. Thinking practically, could mor
uncovered the software flaws i

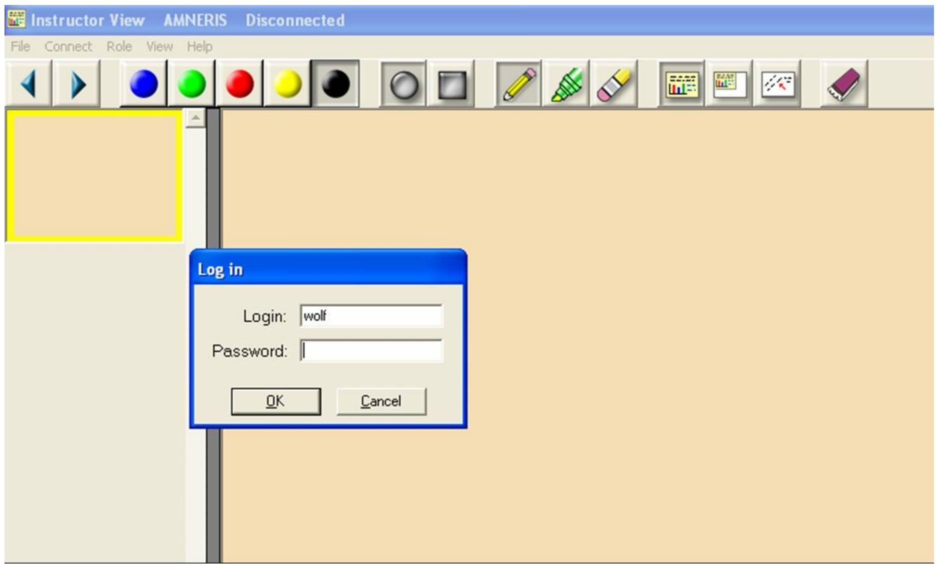
Figure 5.28: A presentation author selects the title and “venue” (multicast address) for the presentation from Figure 5.26.

Each presentation can be instantiated multiple times, each time with a clean slate of student data. The same database browsing application mentioned above allows instantiation of existing presentations into new presentation instances.) On upload of a presentation, the SIP design environment automatically generates and uploads slide images in each view mode — instructor, student, and shared — to the database and registers all the SIP widgets and their properties.

Accessing a SIP presentation begins with starting the .NET remoting application, the "SIP server", that mediates SIP's access to the database. The server usually runs on the same computer as the database and can be left running to support multiple SIP sessions. The presenter then starts SIP's execution environment and logs in, as in Figure 5.29. SIP displays a list of available presentation instances, as in Figure 5.30. Each instance includes a title, presenter, multicast venue, date the presentation instance was updated (*i.e.*, created or last run), and whether a current, locally cached copy of the presentation data exists.[5] The presentations are sorted in order from most recently added or modified to least. Therefore, instructors will often want one of the presentations high in the list, and most students will just load the first (default) presentation, *i.e.*, the one their instructor just loaded to start the class.

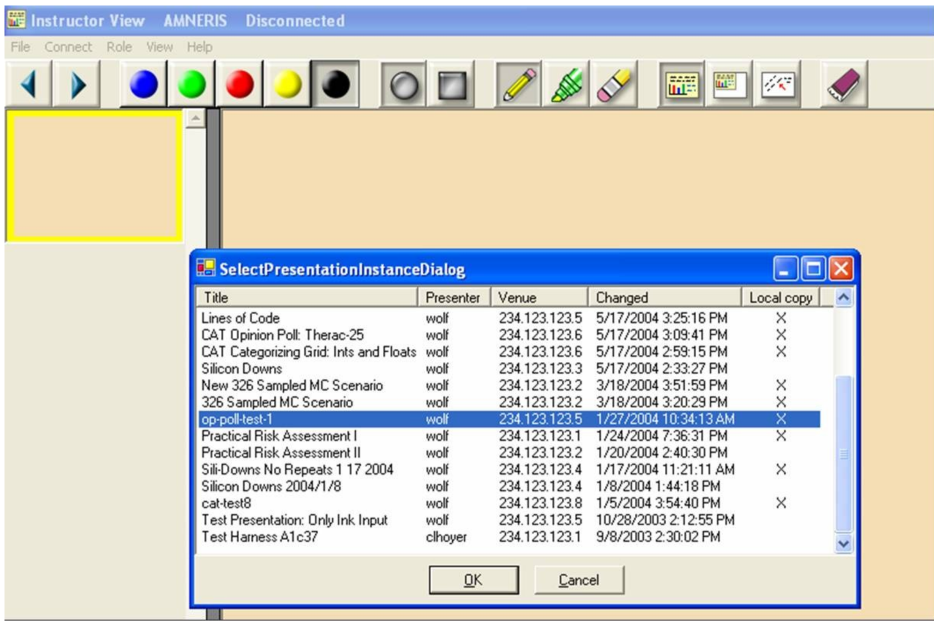
When a participant loads a presentation for which she is the instructor (as indicated in the database), SIP opens the presentation, automatically enters the instructor role, and connects to the presentation instance's multicast venue, as in Figure 5.31. When a participant loads a presentation for which she is not the instructor, SIP opens the presentation, automatically enters the student role, connects to the presentation instance's multicast venue, and selects the first available client in the instructor role on that venue as the distributed presentation's instructor, as in Figure 5.32. (If no instructor is currently available, the student client waits until one becomes available and then attaches to that instructor.) Instructors start the projector role for a two-computer SIP configuration by logging in a second time

[5]Caching speeds loading of the presentation. The visible indicator of caching also reminds a presenter which presentations she has run on her local device.


Instructor View AMNERIS Disconnected
File Connect Role View Help
Log in
OK
Cancel

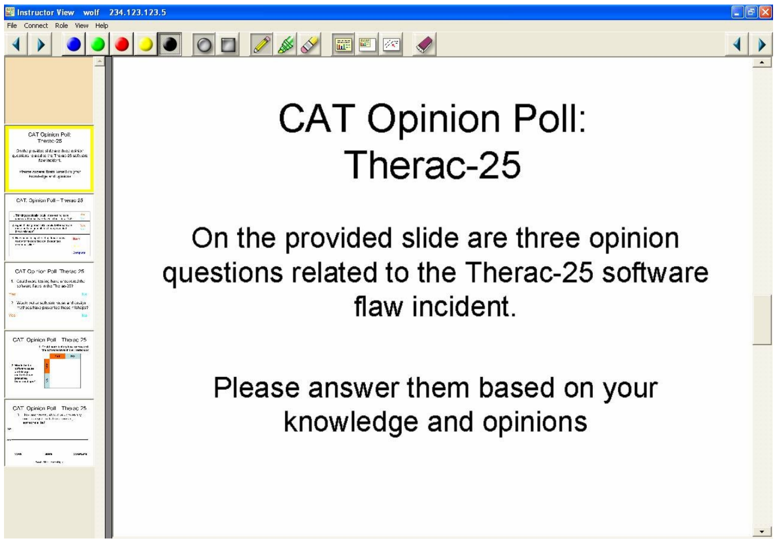
Login: wolf
Password:

Figure 5.29: A SIP user logs in at the SIP execution environment's startup screen. If the login is successful, the user will proceed to a screen like Figure 5.30.


Instructor View AMNERIS Disconnected
File Connect Role View Help
SelectPresentationInstanceDialog
OK
Cancel

Title	Presenter	Venue	Changed	Local copy
Lines of Code	wolf	234.123.123.5	5/17/2004 3:25:16 PM	X
CAT Opinion Poll: Therac-25	wolf	234.123.123.6	5/17/2004 3:09:41 PM	X
CAT Categorizing Grid: Ints and Floats	wolf	234.123.123.6	5/17/2004 2:59:15 PM	X
Silicon Downs	wolf	234.123.123.3	5/17/2004 2:33:27 PM	
New 326 Sampled MC Scenario	wolf	234.123.123.2	3/18/2004 3:51:59 PM	X
326 Sampled MC Scenario	wolf	234.123.123.2	3/18/2004 3:20:29 PM	X
op-poll-test-1	wolf	234.123.123.5	1/27/2004 10:34:13 AM	X
Practical Risk Assessment I	wolf	234.123.123.1	1/24/2004 7:36:31 PM	X
Practical Risk Assessment II	wolf	234.123.123.2	1/20/2004 2:40:30 PM	
Sili-Downs No Repeats 1 17 2004	wolf	234.123.123.4	1/17/2004 11:21:11 AM	X
Silicon Downs 2004/1/8	wolf	234.123.123.4	1/8/2004 1:44:18 PM	
cat-test8	wolf	234.123.123.8	1/5/2004 3:54:40 PM	X
Test Presentation: Only Ink Input	wolf	234.123.123.5	10/28/2003 2:12:55 PM	
Test Harness A1c37	clhoyer	234.123.123.1	9/8/2003 2:30:02 PM	

Figure 5.30: A SIP user selects among presentation instances. After selecting a presentation, the user will proceed either to a screen like Figure 5.31 — if she is the presenter for this presentation — or to a screen like Figure 5.32 — if she is not.

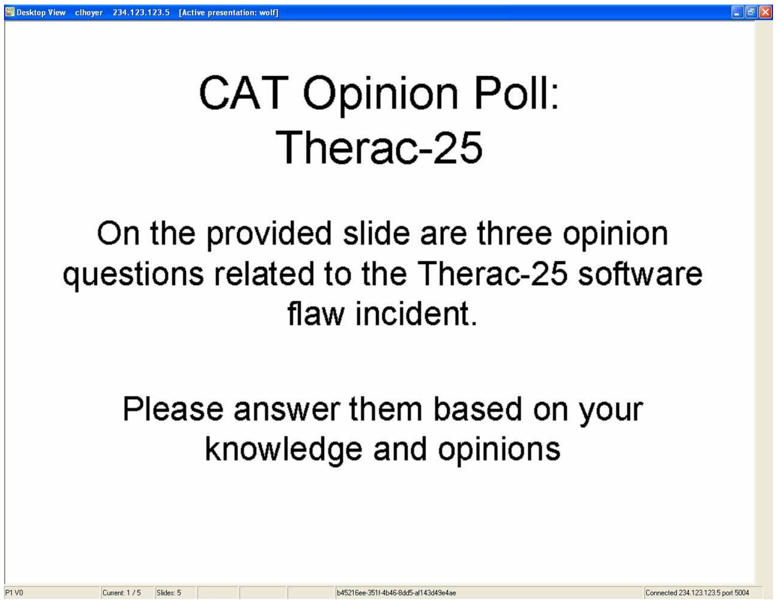

Instructor View wolf 234.123.123.5
File Connect Role View Help

CAT Opinion Poll:
Therac-25

On the provided slide are three opinion questions related to the Therac-25 software flaw incident.

Please answer them based on your knowledge and opinions

Figure 5.31: The opening screen for an instructor who has just logged in to one of her presentations. SIP automatically selected the instructor role and connected to the presentation instance's venue.


Desktop View clhoyer 234.123.123.5 [Active presentation: wolf]

CAT Opinion Poll:
Therac-25

On the provided slide are three opinion questions related to the Therac-25 software flaw incident.

Please answer them based on your knowledge and opinions

P1 V0 Current: 1 / 5 Slides: 5 b45216ee-351f-4b46-8dd5-af143d49e4ae Connected 234.123.123.5 port 5004

Figure 5.32: The opening screen for a student who has just logged in to an instructor's presentation. SIP automatically selected the student role and connected to the presentation instance's venue. The instructor "wolf" already logged in to the presentation, and the student's client selected "wolf" as the instructor (indicated by the tiny text "Active presentation: wolf" on the title bar).

with the same account and switching to projector role, as in Presenter (see Section 3.3.2). In all cases, SIP uses unicast remoting to connect clients to the SIP server.

During a presentation, communication with the database happens automatically. All SIP widgets automatically instantiate themselves, pass new output to the database, and respond to new inputs from the database. Some properties may have associated queries which update the properties' values based on other properties' values. The presentation instantiation process automatically identifies the dependencies between these queries and the property values they rely on for calculations. During execution, the SIP server uses these dependencies to rerun queries and recalculate associated properties' values whenever necessary. More generally, the SIP server mediates communication with the database: caching frequently accessed results, arbitrating conflicting data updates, and tracking changes in data that might require refreshing queries.

The underlying SIP database centers on the concepts of presentation, participant, and widget. Figure 5.33 shows the relationships among presentations, participants, presentation instances, and instance participants. Each participant has a textual name (required to be unique), an automatically assigned unique ID, a password, and a creation timestamp. Each presentation has a name, presentation data (an enhanced Presenter slide deck), an automatically assigned unique ID, and a creation timestamp. The presentation also includes an author ID that must correspond to some participant's ID and a security field that is reserved for future use. A presentation instance has a name, a multicast venue, an automatically assigned unique ID, and an update timestamp. Each presentation instance also has a presenter ID that must match some participant's ID, a presentation ID that must match some presentation's ID, and a security field reserved for future use. Unless otherwise specified, a new presentation instance uses the name of its presentation as its name and its presentation's author as its presenter. The presentation instance's update timestamp is updated each time its presenter reopens the presentation in SIP. Finally, an instance participant represents a participant's relationship to a given presentation instance. Each instance participant has a role (*e.g.*, "instructor" or "student") and a creation timestamp. Each instance participant also

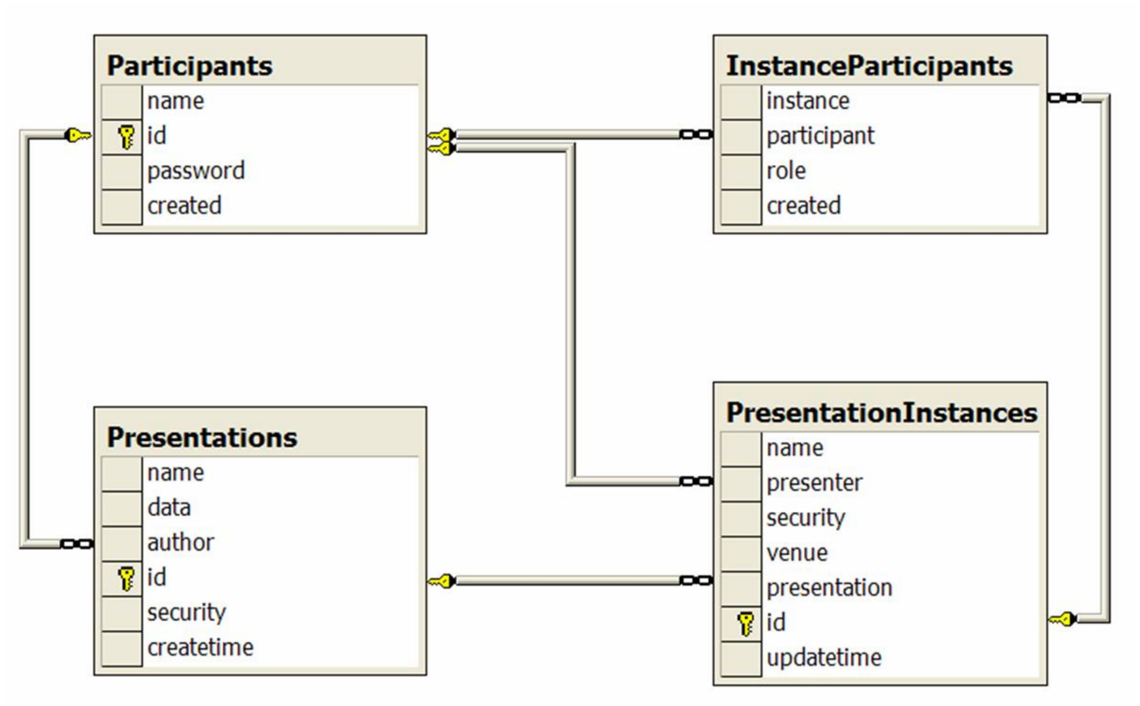


Figure 5.33: Database schemata for SIP's high level concepts: presentations, participants, and their instantiations. Each box is a table labeled with its title. Each row in the box is a field of the table. Fields marked with a key symbol are keys for their tables. (No two records may have the same set of values across their key fields.) Fields at the start of an arc (chain symbol) must match some value of the field at the end (key symbol) of the arc. In other database visualizations below, arcs may not clearly begin or end at a field. In these cases, the relationships are described in text but should be easy to deduce from the figure.

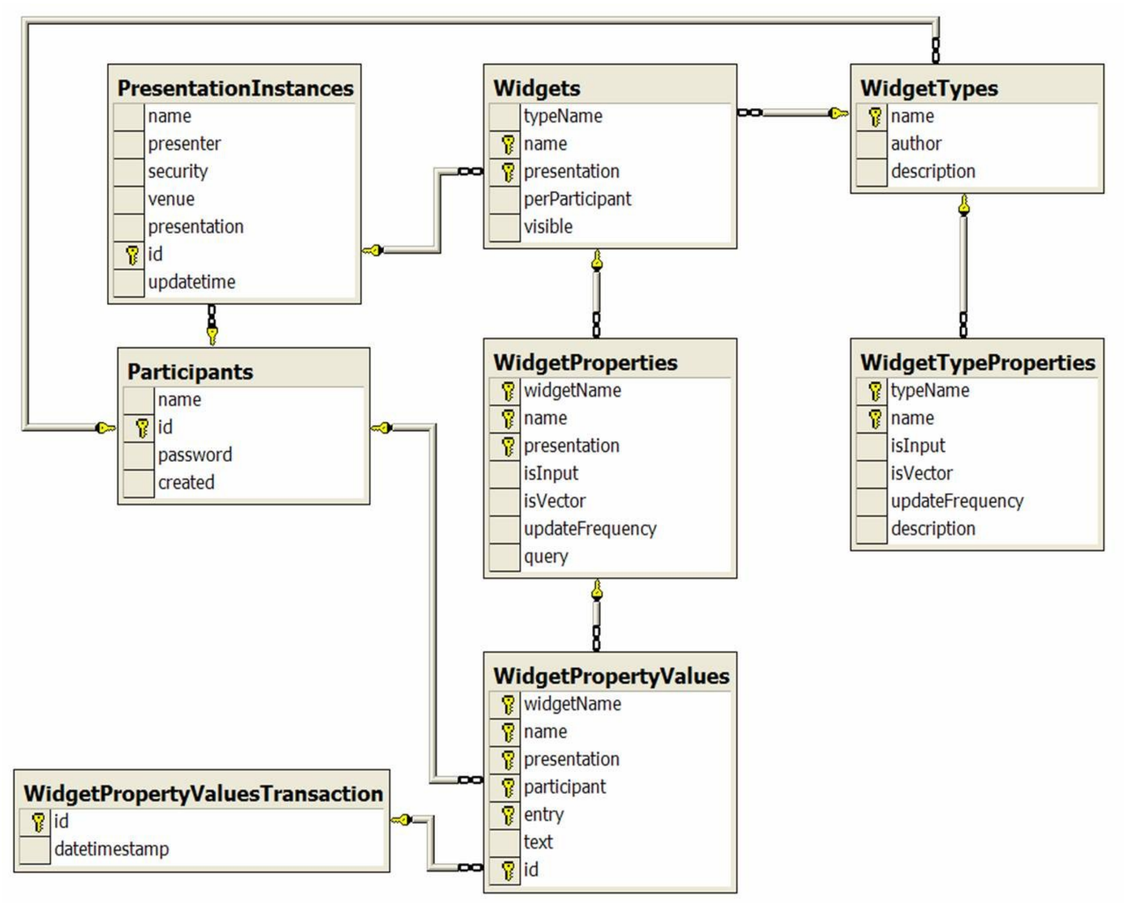


Figure 5.34: Database schemata for SIP's core components: widgets and properties. For an explanation of the visual elements, see Figure 5.33.

has a participant ID corresponding to some participant and an instance ID corresponding to some presentation instance. The instance participant table thus comprises an enrollment list for each presentation instance.

The heart of the database relates widgets, their properties, and their values to widget types, presentation instances, and participants. Figure 5.34 shows this portion of the database. Presentation instances and participants are described above. The right side of the figure shows the schemata for widget types, from which widgets configured for individual presentations are instantiated. Each widget type has a unique name, an author ID referring to an existing participant, and a textual description. Each type can have many properties. Each widget type property has a name and a textual description along with fields indicating whether it is an input (actually, input/output) property or solely an output property; whether it is a scalar (single string) or vector (list of strings); and how often SIP should run any associated query to update this type of property (only once or each time a dependency of the query changes). Finally, the widget type property includes a reference to its widget type.

Instantiated widgets are represented by the tables in the middle of Figure 5.34. Each widget includes a type name and presentation ID that associate it with its type and presentation instance. Each widget also has a name unique to its presentation instance; a field indicating whether its properties should be separately instantiated for each participant (*e.g.*, a text input widget allowing separate input from each participant) or instantiated once and shared across the presentation instance (*e.g.*, a text display widget showing the single average of numerical inputs from the whole class); and a "visible" field reserved for future use.

A widget property refers to its widget by presentation ID and name (since name alone is not necessarily unique). Each property also indicates whether it is an input or output property, whether it is vector or scalar, and its update frequency. These fields are described above for property types and are instantiated to the same value held by the associated property type. Each widget property that is an input may also have an optional query. If present, this query is run at least once to assign a value to the property and may be run multiple times depending on the update frequency. (Also, the query may be run once per participant or once across all participants depending on the associated widget's "per participant" field.) Finally, each property has a name that is unique in the context of its widget and its widget's presentation instance. Note also that not all properties are necessarily instantiated from property types. The SIP design environment allows presentation authors to generate new properties to help manipulate data, *e.g.*, to collect data from several widgets' outputs before passing it to the current widget's input. These properties must be assigned a new, unique name but may otherwise be configured freely, and thus the "isInput", "isVector", and “updateFrequency” fields are not necessarily redundant.

All student data is stored as a value of some widget property. Conceptually, there is one value (for scalars) or one list of values (for vectors) associated with each property/widget/instance/participant combination. In practice, several factors complicate this arrangement. First, the database stores every value ever assigned to any given property. Therefore, a widget property value has a unique ID that associates with a timestamp (which may be shared with other property values set in the same operation). Only the value with the most recent timestamp is actually current. Second, the “list of values” for a vector property is stored across multiple rows as text values associated with the same ID, participant, property, widget, and presentation instance. Elements in the list are numbered from one using the “entry” field of the widget property values table. Each vector also includes an empty zero entry, which distinguishes an uninitialized vector from a zero-length vector. The value of a scalar property is simply stored with the “entry” field set to zero. Finally, widgets may be instantiated either per participant or per presentation. Properties of widgets instantiated per participant include a reference to a normal participant ID to indicate which student device generated the data. Properties of widgets instantiated per presentation may refer only to a specially designated “null participant”, allowing only one value (or list, for vectors) per widget/presentation combination.

Each widget property value, therefore, includes a widget name, name (property name), presentation ID, participant ID (or null participant reference), entry number, ID, and text value. The widget property values transaction table associates widget property value IDs with timestamps. While each ID may appear only once in the transaction table, it may appear multiple times (paired with distinct widget/property/presentation/participant/entry combinations) in the widget property values table.

Finally, SIP also uses the database to associate widgets with their representations. Figure 5.35 illustrates this portion of the database. Each widget type is associated with one or more representations of various types. For now, only two types of representation exist: Classroom Presenter (“CP”) representations and PowerPoint (“PPT”) representations.

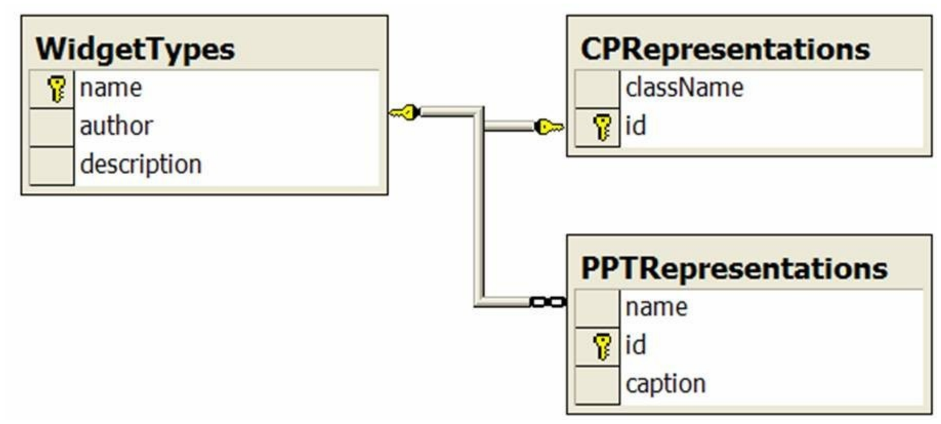


Figure 5.35: Database schemata for the external representations of SIP widgets (in the SIP execution environment and PowerPoint add-in). For an explanation of the visual elements, see Figure 5.33.

Eventually other applications could also support SIP widgets and lodge appropriate representation descriptions in this table. Eventually, adding a new table for each representation might become unwieldy, in which case a single table for all representations with a column indicating the target application might be preferable. For now, the requirements of each representation seem distinct enough to merit separate schemata for separate representations. A transition to a single table would require creating views that simulate existing representation tables for backward compatibility with applications expecting their own representation table.

Each Presenter representation indicates a fully qualified .NET classname used to implement the widget. The indicated class must extend the abstract “SIPControl” class, which requires only that subclasses implement a two-place constructor taking a widget name and an interface for data management and pass them to the superclass. The SIPControl class otherwise handles all communication with the database. Subclasses access properties through an object returned by a protected “RequestProperty” method of SIPControl. The object supports getting its value (either vector or scalar), setting its value, and attaching to an event that fires when the value changes.

The PowerPoint representation currently includes name and caption fields. The caption field is used to label the button that inserts the widget into a PowerPoint presentation. The name labels the textbox placeholder laid out in the presentation itself. The design environment initializes information about the widget's properties directly from the type signature in the database.

As mentioned above, each widget's input property can contain a query. SIP uses the query to initialize the input property's value and, if the property is marked as requiring updates, to update the property after each change to data the query depends on. In principle, the contents of a property query can be any legal SQL query (though not a SQL update). Allowing such free queries gives instructors the full power of SQL, particularly the aggregation functions SQL supports (*e.g.*, summary statistics functions like maximum and standard deviation). However, this freedom has at least three severe disadvantages. First, it demands from instructors the ability to write SQL code. This is unrealistic for instructors outside computer science. Even within computer science, those instructors with the ability to generate SQL code likely would prefer not to do so to create their presentation slides. Second, exposing raw SQL to instructors requires them to have knowledge of the structure of the underlying database. Again, this requirement is an unreasonable burden on instructors. Furthermore, exposing these details to instructors (and encouraging them to write code depending on these details) limits SIP's future flexibility. Finally, in order to update queries when the data they depend on changes, SIP must "understand" how those queries access data. Teasing out dependencies on upstream data from arbitrary SQL queries is impossible in general.

Therefore, SIP exposes a set of useful macros to instructors to simplify queries. The most important of these let instructors specify a source of data by its associated presentation, participant(s), widget(s), and property or properties. Furthermore, instructors can leave off prefixes of these data to access their "default" values (*e.g.*, the current presentation). So, for example, an instructor could write:

```
$<SIP_VIEW.%.multiple-choice-%.responsenumber>
```

to access a view summarizing results for all properties named `responsenumber`, contained in widgets with names starting `multiple-choice-`, entered by any participant, and in just the current presentation — more precisely, the current presentation instance. Note that "%" is a SQL wildcard that matches any sequence of zero or more characters. Other macros allow access to the current presentation's ID, the current participant's ID, and other contextual information about the presentation. (Prepending these macros with `SIP_VIEW` yields a SQL query representing this view. Leaving the prefix off yields the string data associated with the view.)

As another example, consider the "hypothetical vaccine" exercise illustrated in Figures 5.4 and 5.5. The top bar of the four in Figure 5.5 aggregates all the student responses to the question "Is the vaccine worth using?" from the start of class. This particular bar uses two query-based properties: one to count the total number of responses and one to count the total number of "Yes" responses. The latter property's query is:

```
SELECT COUNT(*)
FROM $<SIP_VIEW.vaccine-mc-1.responsenumber> t
WHERE sip_responsenumber LIKE '1'
```

This query accesses a view of widget `vaccine-mc-1`'s `responsenumber` data. It considers only the `responsenumbers` that equal `'1'` ("Yes"). Finally, it counts the number of such responses.

Using macros simplifies the instructor's task in designing property queries somewhat and also exposes key data that helps SIP establish dependencies among properties. SIP scans each property query for macros accessing particular data views (like the examples in the previous paragraphs) and establishes a list of dependencies that encompasses all data that could possibly be referenced by these macros. Therefore, queries that use only the macros to access data from the SIP database are guaranteed to be updated when data they are dependent on changes. Queries that directly access the SIP database receive no such guarantee. In either case, each time SIP runs a query, it extracts the contents of the first column as strings (or just the first column of the first row for scalar properties) and uses the result as the new value for the associated property.

More detailed description of these queries is outside the scope of this dissertation, primarily because the SQL queries and the macro language are both temporary measures. As Section 5.3.8 describes, replacing these daunting languages with facilities appropriate for the instructor/end-user is a crucial next step for SIP.

Database benefits and problems

Using a relational database as SIP's back-end data storage system entailed both advantages and disadvantages. The primary advantages were access to the power of relational databases: persistence, reliability, speed, and rich query capabilities. Many aggregation, filtering, and selection operations performed as part of linking together SIP widgets' properties are trivially and efficiently implemented as SQL queries. Furthermore, the relational database easily manages storage of *all* data generated by all students in a class as they interact with SIP. Even for large classes, the number of students is small enough to make a database administrator chuckle. However, factoring in the dozens of widgets in a presentation, the half-dozen properties of each widget, and the frequent changes student actions impose on those property values, the volume of data that would be challenging to manage without database support. Theoretically, any of this data is also available in the database for complex post-processing or data mining.

Unfortunately, using a relational database also caused some serious problems. Chief among these was the complexity of managing complex, structured data as if it were flat, hierarchical data. That is, each SIP widget naturally induces its own structure on data, *i.e.*, sets, lists, or named collections of string, integer, floating point, or ink data. For example, a simple multiple choice question might naturally have a list of string responses, an integer identifying the correct response, and another integer identifying the response that the user has selected. A more complex structure for multiple choice questions might instead demand a list of "text responses" each of which includes fields for the string contents, a font (including color, size, *etc.*), and a justification. Furthermore, the "correct response" field might be better replaced with another list associating responses with point values.[6] Indeed, the entire notion of a "multiple choice widget" might be folded into a larger "multiple choice quiz widget" that contains lists of multiple choice questions along with other properties.

None of these more complex structures is represented in SIP. Instead, a widget can have a named collection of properties typed within the cross product of input/output and scalar/vector. *All* the actual data is stored as text strings. These choices reflect the difficulty of representing fluid, configurable hierarchies of data in a relational database. If the hierarchies are known in advance, it is easy to generate tables that encapsulate those hierarchies. The tables can even be generated on the fly as new widget types are entered into SIP. However, the hierarchies are *not* known in advance, and adding new tables for each new widget type would rapidly render the database schema unmanageable. Furthermore, this arrangement would either need to support the additional burden of instructors adding to widgets' property structures at design time or restrict instructors' ability to customize widgets to suit their individual purposes, a violation of our guiding design principle to enable end user innovations. An alternate choice of database type — *e.g.*, object, object-relational, or semi-structured — might have alleviated some of these problems.

Finally, databases have unwieldy support for event-driven programs that respond to changes in the data. This lack of support led to the need for the intricate "SIP server" application, which reimplemented much of the functionality that should have been managed by the database, such as resolution of conflicting updates to the database, so that it could

[6]The SIP style of constructing interaction exercises from smaller parts strongly suggests that multiple choice widgets should *not* include the question prompt, as that is easily placed as a separate text object. In fact, many of the presentations included in this dissertation eschew the multiple choice question's built in support for response text as well, placing responses as separate text objects. Both of these choices do make it more difficult for downstream widgets to reason about upstream multiple choice questions, however.

also reissue property queries when the data they depend on changed.

In the end, the database solved more problems than it caused. However, a better design would have disentangled SIP's property specifications from the underlying database, allowing easy transition from "true" database support and, *e.g.*, text-based XML storage. Indeed, SIP widgets benefit from just such buffering, managed by standard widget base class. As a result, SIP's widget authoring environment can test widgets either with their properties attached to the database back-end or using trivial stubs to supply property values. (If property queries were disentangled from the database, the widget authoring environment would also be able to simulate property updates without a connection to the database.)

5.5 Experimental SIP Class

As a first step toward understanding how SIP addresses the challenge of providing an interactive "score" for the classroom, we conducted an experimental, proof-of-concept class session with SIP. The session's goal was to test the design and execution process for a SIP presentation and begin to understand how instructors will use SIP to create interactive presentations and how students engage in the resulting presentations.

I acted as the class instructor. 24 faculty, staff, and students from my department (a computer science and engineering department) acted as the class's students.[7] To foster a realistic class environment, the class topic was risk assessment in a public health context, a topic that was of interest to but not a common area of expertise of members of the department.

The experimental SIP class had four primary goals: (1) to demonstrate that SIP could effectively support an interactive style of instructional design, (2) to demonstrate that the SIP prototype implementation was technologically sound enough to be used in a realis-

[7] 6 of the 24 were members of or affiliated with the Classroom Presenter project, and 2 of these 6 were members of the SIP research project. All results in this dissertation are reported both including and excluding the Presenter project team members. Also, note that volunteers were rewarded with cookies, though they were *tasty* cookies.

tic classroom scenario, (3) to study the classroom environment that use of SIP engenders, and (4) to begin the “intervention” side of SIP’s iterative design process. Section 5.5.1 describes the design process used to create SIP slides/exercises for a full hour-long class, demonstrating SIP’s effectiveness for design. Section 5.5.2 first establishes that SIP’s execution environment (mostly) worked during the class session and that the errors that occurred had no critical negative impacts. Then, Section 5.5.2 weaves together the remaining two goals, using data and experiences from the hour-long course to illustrate SIP’s effect on the classroom environment and discussing how these data reflect on our design decisions in SIP.

5.5.1 Instructor’s Reconstruction of Design Process

Throughout the instructor’s preparation process, he made reflective notes, recording the mechanical and cognitive process as it occurred. The indented text below is the instructor’s reconstruction of the design process based on design notes, presentation artifacts, and memory. This reconstruction shows how SIP’s facilities played an integral role in the design process, helping the instructor to create a presentation and exercises that addressed his learning goals for the students.

The design notes used for the reconstruction are reflective notes made by the instructor after each episode of working on the presentation or significant steps during a period of work. There are 12 notes taken over the 10 days leading up to the class with a total of 288 non-blank lines of text and 2761 words. Only the first 7 days were used for presentation design, and only a fraction of the instructor’s time during these days was devoted to design of the presentation. However, exact timing of effort on the presentation is available only for the process of configuring and debugging the SIP properties (as described below). The final three days before the presentation were used for setup of the computers involved: physical arrangement, installation, networking, and testing. The presentation artifacts used for the reconstruction are five drafts of the SIP slides eventually used in the class.

The reconstruction follows between vertical bars:

I began the design process for the experimental SIP class by selecting a topic, admittedly a less constrained task than an instructor in a regularly scheduled course would face. Because I wanted a topic outside the normal expertise of computer science department members, I scanned references from unrelated fields for inspiration. An article in an organic chemistry text [71, pages 29–30] suggested the topic of risk assessment in a public health context. The article contrasts people's, perhaps unwarranted, comfort with dangerous natural substances against their fear of less dangerous artificial substances (such as aflatoxin *versus* chloroform). Risk assessment would be an area of interest but not expertise to department members. Furthermore, I felt I could develop enough expertise on the topic in the ten days available to arrange a valuable one-hour class on the topic. Finally, risk assessment does have a tie-in to computer science in the artificial intelligence topics of rationality and probabilistic reasoning.

I next gathered and studied a set of materials to use as the basis of the class. I used the books [71], a preprint of [66], [49], and [89] as well as the medical information website WebMD [114].

As I studied these materials, I also began constructing a list of learning goals for the class. I targeted goals that students could perform/achieve rather than abstract "coverage" of the topic. This was partly because of my unfamiliarity with the topic, partly because of my recent reading of Walvoord and Anderson's text on grading (which advocates centering course structure and content around performance learning goals) [110], and partly because SIP's affordance of student interaction naturally led me to a model of active student involvement.

After several revisions and consultation with another SIP project team member, I established a list of learning goals for the session. By the end of class, students will: (1) believe that some of their perceptions of chance and risk are inaccurate, (2) predict how technological changes will affect societal risk perception, and (3) consciously elaborate their own utility curve for risk (*i.e.*, a qualitative sense of how their tolerance for risk changes with the magnitude of the risk and their current circumstances).

Over the course of designing the class, I eventually decided that two learning goals were enough for one fifty-minute session and dropped the goal #3 and the "chance" aspect of goal #1.

As I developed learning goals, I also began to arrange my thoughts and materials for the class into a SIP presentation. Designing interactive exercises as I designed structure and flow of static lecture material was natural in SIP. By incorporating interactivity into my design tool, SIP pulled interactive exercises into the foreground, making them "first-class citizens" of my design process, just like textual descriptions of material. Even though proper configuration of interactive exercises' properties and connections was a ways off, the placeholders for an interactive exercise in the design environment are artifacts that mediate my design, extending my cognition by their persistent arrangement, availability, and display.

The first exercise I designed focused on the danger of influenza and was based on the "misconception/preconception check" exercise from Classroom Assessment Techniques [13, pages 132–137]. Appendix Figures A.2–A.9 show the full exercise. Students estimate the number of people who die each year in the United States from influenza, and the next slide aggregates students' responses into a graphical plot of the estimates and summary statistics. I planned to use this plot to encourage discussion of how we judge risks in our lives and why we misjudge them, beginning to address learning goal #1. I also planned to use this exercise early in the lecture to "disequilibrate" students [20, 83] — violate their expectations in a way that would motivate them to learn.

As a disequilibrating activity, I realized that this exercise demanded a paired activity, both as an opportunity for students to reestablish equilibrium after the initial influenza exercise and as an opportunity for students to *succeed* on an exercise. It seemed like that activity could either go next or at the end and should explore how students would evaluate the risk of influenza. This would probably be a distributed human computation exercise (see Section 5.2.3) to give students the opportunity to think about other students' responses. Eventually, this developed into the exercise described in Section 5.2.3 in which students imagine and evaluate questions to assess the risk of a hypothetical perfect vaccine for influenza.

I then laid out the middle of the talk, dividing it into one section on identifying misapprehensions of risk on important modern topics (other than influenza) and another on assessing risk using a social-historical perspective. Initially, the plan also included a section on assessing risk from a mathematical-psychological perspective, but this section was dropped with learning goal #3.

At this point, five days into the design process, I had fully planned the structure of the class and drafted a presentation that embodied that structure. The presentation included an initial, disequilibrating influenza exercise, a two-part exercise evaluating the perception of risk of an influenza vaccine (one part near the beginning and another near the end), an exercise evaluating the risk of common substances and activities, an exercise evaluating the change in perception of risk of fire and asbestos over time, and a closing exercise imagining questions for evaluating an influenza vaccine. The first exercise used numerical input and the last used a combination of free text and multiple choice input, but all the other exercises used multiple choice input. I eventually changed the asbestos and fire exercise into a drawing exercise because of the monotony of so many multiple choice exercises in a row and because of my bias toward including an inking exercise to evaluate SIP's support for ink input and display.

I had not yet performed any configuration of SIP properties to actually make the interactive exercises work. Instead, I used the layout and textual contents of SIP widget placeholders to identify the role and purpose of interactive elements.

Throughout this initial process, I used both the SIP presentation and my own separate notes to support my design, referring to them to remind myself of my current design, and editing them to reflect new ideas and directions. I made extensive use of Presenter's "instructor notes" feature to record "todo" items and other private thoughts that were not to appear in the final, public presentation. I also used SIP's widget toolbars as reminders of the interactive functionality at my disposal.

In order to gauge the time necessary to configure SIP properties for a full class presentation, I finished all the property editing in one timed session. Before the session, I performed none of the configuration of SIP properties necessary to make the interactive exercises work. Instead, I used the layout and textual contents of SIP widget placeholders to identify the role and purpose of interactive elements. During the session, I configured all the SIP properties and also made small modifications to presentation layout where it seemed to improve the presentation, *e.g.*, transposing rows and columns in the display of common substance and activity risk results so that the layout of the results slide matched the layout of the prompt slide (as in Appendix Figures A.21–A.24).

Overall, the editing session took just under an hour: 59 minutes and 30 seconds. Multiple selection and editing of widgets' properties (described in Section 5.4.1) worked particularly well during the editing process. Copying, pasting, and editing similar exercises from previous presentations also worked well and reduced the time necessary to configure SIP widgets. In fact, none of the exercises needed to be built from scratch since similar exercises existed in other presentations. For example, the lead-off exercise about the number of influenza deaths per year (see Appendix Figures A.2–A.9) closely resembles the "America Before Columbus" exercise about the number of people in North America before 1492 (see Appendix Figures A.62–A.65). Constructing the influenza exercise involved copying the America Before Columbus widgets, renaming them and editing properties for a few desired effects (such as omitting responses larger than 100,000), and adding extra widgets for some new functionality (such as displaying the maximum, minimum, and average responses). Data validation (*e.g.*, checking that a numeric input only provides numbers) was difficult, requiring significant SQL code and could have been improved by more support from existing widgets, widgets customized to numerical and other data types, or separate validation widgets that could serve both to filter data and visually indicate to SIP users whether their input was valid.

The property editing session completed all major editing on the SIP presentation. However, I did then spend three further sessions totalling just under one hour and fifteen minutes debugging problems in the presentation. The most time consuming problems were minor mistakes in SQL queries linking widget properties together. Such problems would be mitigated by a visual design process that automatically generated the simple underlying SQL queries. Other issues included minor aesthetic fixes to the presentation such as realigning misaligned widgets and static slide objects. Finally, this time includes locating and fixing one bug in the SIP code exposed earlier in the design process.

Appendix Section A.1 shows six versions of the completed presentation folded into one sequence — the cross product of two stages (during design and during execution with data from the class) and three views (instructor, student, and public projection display). During the actual class, a bug in one widget property caused some of the data for the initial influenza exercise to be dropped from display. For these screenshots, the bug has been fixed, and all the original data is displayed. Otherwise, the screenshots differ only in not including the instructor's ink.

5.5.2 Experimental Class Execution

Appendix Section A.1 details the presentation used in the experimental SIP class. The class met its primary goal: demonstrating that a classroom full of students could interactively engage in a SIP lecture for a full course period of one hour. It also illustrated some strengths and weaknesses of SIP. This section discusses SIP's use in the class based on the instructor's notes from the day of the class; notes from two trained observers who focused on student interaction; video, audio, ink, and SIP data logs of the class[8]; and survey results from the students (see Appendix Section B.2 for the survey form). The survey was distributed immediately after the end of class, and although it was voluntary, all but one student completed the closed response section of the survey and at least one of the open response questions.

SIP's technical performance

From a technical standpoint, SIP was mostly successful during the experimental class. The technical problems that did occur are discussed in detail below. Although discussion of "everything going fine" provides few lessons or insights, it is important to stress that the during vast majority of classtime SIP operated perfectly: slides advanced when they should, exercises worked as they should, and response time was real-time or near enough to avoid notice.

Two major technical problems arose, although only one was obvious to students. The more obvious and severe problem required restarting SIP on students' computers. At

[8]The video captured only the instructor. The audio primarily captured the instructor's voice although "crowd noise" of the students and louder individual remarks are audible.

24:15[9], the instructor noticed that a transition on his display was not reflected on the public display. Shortly thereafter, he determined that the students' devices had also failed to transition. The instructor told students to "discuss amongst yourselves about asbestos and fire for a minute or two" and began diagnosing the problem. The instructor and two assistants took about two minutes to diagnose the problem — a failure of network connectivity stemming from SIP's known use of a deprecated network communication library — and restart the instructor's SIP client. At 25:57, the instructor began instructing students to close, restart, and reconnect their SIP clients. Once students had reconnected, the instructor tested the connection (by writing some ink and checking that students could see it). The test was successful and at 28:18, four minutes after noticing the problem, the instructor resumed normal progress of class.

Several factors mitigated the severity of this problem. First, SIP had been designed to support a quick and easy restart (as described in Section 5.3.6); so, the delay caused by the failure was brief. Second, the delay fortuitously coincided with the midpoint of the class's 56 minute length and a transition between major topics (from commonly held perceptions of risk to historical shifts in risk perception). The students had just completed an introductory exercise for the "historical shifts" section. Students took well to the opportunity to discuss the first section of the class, their initial thoughts on historical shifts in risk perception, and the SIP System itself. Indeed, students did not generally think of this system restart as an important problem. Only 21% of respondents (28% of those uninvolved with the Presenter project) mentioned the system restart in response to the survey prompt "Please briefly describe any problems (e.g., technical or user interface issues) you had with the SIP system." Because this is an open response question, we should expect that more students than those who mentioned the system restart might have thought of it. However, by comparison, 33% of respondents (22% of those uninvolved with the Presenter project) stated that they had *no* technical problems. Table 5.1 summarizes responses to this survey

[9]All presentation times will be given as *mm*:*ss* where *mm* is the number of minutes and *ss* the number of seconds from the start of the presentation.

prompt.

The second major technical problem was a bug in the SIP property specifications for the first exercise of the session — the one shown in Appendix Figures A.2–A.9 — that occurred between 01:05 and 05:53. A typographical error in one of the properties' SQL code caused SIP to discard some of the student data from the exercise.[10] Again, several factors mitigated the severity of the problem. First, enough student data was processed to be representative of most students' responses. Thus, most students did not notice their missing data. In fact, only two students (8% of the survey respondents and 6% of those uninvolved with the Presenter project) reported missing numbers in the initial exercise. One student also asked about the problem during class, but the class as a whole did not discuss the problem. Some students likely forgot about this problem before the survey simply because it occurred near the start of the class; however, many others likely did not notice their data missing. Indeed, it is telling that the missing responses referenced by all three students who mentioned the problem (one during class and two on the survey) were outside the range of displayed responses, either unusually small or unusually large, and therefore particularly noticeable.

Besides these major problems, all the technical problems manifested during the class were minor. The only one mentioned by more than 2 students was a distracting display of handwriting recognition "results" on an inking exercise (the exercise in Appendix Figures A.25–A.30). The presence of the display was unintentional since the exercise did not use the recognition results and the inking requested was a graph, not writing. Three students mentioned this problem. Of the remaining issues, the most important are minor problems students had understanding how to use input areas (*e.g.*, "It was sometimes hard to see if [my input was] sent.") and read graphical displays (*e.g.*, "horizontal bar graphs took a bit to understand").

The instructor also suffered two technical problems during the presentation, besides the system restart described above. First, the "ink list" widget (shown in Appendix Fig-

[10]This bug was fixed for the screenshots included in this dissertation, both in this chapter and in Appendix Section A.1.

Table 5.1: Summary of responses to the SIP survey prompt “Please briefly describe any problems (e.g., technical or user interface issues) you had with the SIP system.” Note that this was the first of the open answer survey questions, and all but one student answered at least one of the open answer survey questions. (See Appendix Section B.2 for the entire survey text.) Percentages in the second column are for all students in the class (24). Those in the third column are for those students uninvolved with the Presenter project (18). The “DNR” category includes all blank responses (which are *not* included in the “none” category). Some responses were multiply coded. “*Any problem*” counts the responses with at least one code besides “none” and “DNR”. (Note that one responses said “no problems” but also listed a problem.)

Category	% of students	% uninvolved with CP
system restart	21	28
none	33	22
DNR	17	17
hand writing recognition	13	17
distracting applications	8	11
hard to use inputs	8	11
lack of student control	8	11
hard to understand display	8	11
unreported flu numbers	8	6
slow network	4	6
my computer crashed	4	6
dropped ink strokes	4	0
any problem	54	67

ures A.29 and A.30) had a serious interface problem. As new student results arrived from the database, the list reconfigured itself to include those results. This caused three problems: delayed response to instructor input when the widget was busy reconfiguring, accidental selection of unintended items when they appeared under the instructor's pen as he clicked, and changes to the selected item without action on the instructor's part (when a new item replaced an existing, selected item). The instructor handled these problems by asking students to stop entering new responses when he was ready to review the responses they had already provided. However, this problem clearly indicates the need for SIP widgets to gracefully manage exogenous events and user input simultaneously. Second, slide transitions on the public display occasionally noticeably lagged the instructor's transitions on his device. This lag was likely due to dropped network data and caused no serious problems in class.

Student engagement with SIP

The experimental SIP class was highly interactive, and students were deeply engaged in the class. Unfortunately, attributing student engagement to SIP is difficult. Potential alternative explanations abound, including the (intentional and SIP-related) active learning style of the class, a predisposition of the students, the unusual nature of this "mock" class, a "Hawthorne effect", the provocative nature of course content, and the instructor's skill at facilitating interaction. The data from this one class cannot possibly refute all of these alternate explanations. Indeed, several of these surely did contribute to students' engagement and interaction. Therefore, this section instead focuses on establishing that the class was highly interactive and that most students engaged in the class discussion.

Compared to a standard lecture, the experimental SIP class was highly interactive. To establish this, I coded class time second-by-second using video and audio logs into "teacher-talk" (any time the instructor was speaking to the class at large or otherwise commanding the class's attention), "student-talk" (any time students were speaking to the class at large), "student-discussion" (any time students were speaking with each other, but *only* if

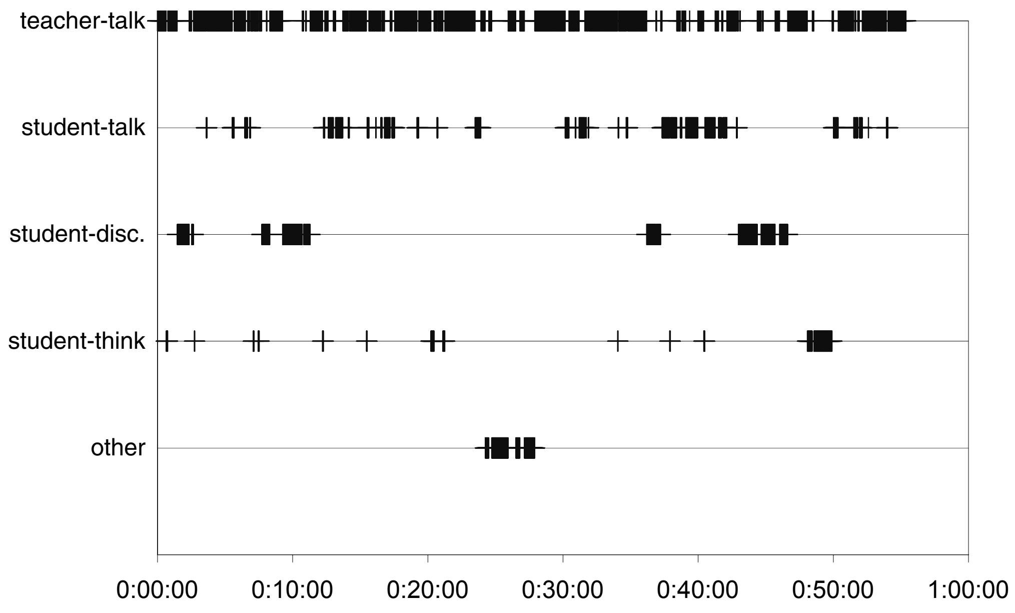


Figure 5.36: Timeline of activity during the experimental SIP class. Time is counted from the start of class, and each vertical line is one second of class time. More than one activity often happened simultaneously, but coding favored higher lines on the chart. Overall, 62% of class time was “teacher-talk”, 15% was “student-talk”, 13% was “student-discussion”, 5% was “student-thinking”, and 5% was “other”.

they were working on an exercise), and “student-thinking” (any time students worked quietly on exercises), and “other” (any other time, mostly time spent responding to technical problems). If a segment of class fit more than one of these descriptions — *e.g.*, the instructor spoke to the class but many students also carried on discussions about an exercise — the time was coded with a single category selected in the order given here. That is, the coding bias was toward counting more time as “teacher-talk” and less time as student-oriented activities.

Only 62% of class time was “teacher-talk”. This compares favorably with the normative expectation of about 68% teacher talk suggested by Flanders,[11] especially consid-

[11]Flanders’ work used the eponymous Flanders Interaction Analysis System (FIAS). Although we did not use FIAS, our system resembles FIAS. FIAS also codes (at its most coarse level) into teacher talk, student

ering that Flanders's norms are for K-12 classrooms, which tend toward decreased interaction at higher grade levels [44]. Furthermore, the instructor of the experimental class spent a substantial portion of the "teacher-talk" time asking questions of students, responding to students' questions and comments, or discussing student input from SIP exercises. Roughly one third (33.4%) of class time was entirely student-focused: "student-talk", "student-discussion", or "student-thinking". Furthermore, as Figure 5.36 shows, this student time — shown in the middle three rows — was spread throughout the course of the class. "Student-talk" happened frequently throughout the class. "Student-discussion" and "student-thinking" was mostly restricted to the six SIP exercises used during the class but also happened briefly during sporadic raised-hand polls and other instructor questions throughout the class. Of course, discussion seminars may be more dominated by "student-talk", but SIP scaffolds an effective intermediate between lecture and discussion.

Students themselves also perceived the SIP class to be highly interactive. The end-of-class survey asked students to rate how often they participated in class through various means, with the option to respond "never", "once or twice", "3–5 times", or "6 or more times". 23 of the 24 students in the class completed all closed answer (*i.e.*, multiple choice) questions on the survey, including these participation questions (see Appendix Section B.2 for question ordering and formatting). The median amount of interaction through the SIP System reported was "6 or more times" ("3–5 times", excluding students involved in the Presenter project). Even more telling, only one student reported interacting fewer than three times through the SIP System. Students also reported substantial participation outside the SIP System. The median amount of interaction with other students reported was "3–5 times" (unchanged by excluding students involved in the Presenter project). Only one student reported never interacting with other students during the class while eight reported interacting with other students six or more times. Most students even reported interacting directly with the instructor, with the median reporting interacting "once or twice"

talk (corresponding to both "student-talk" and "student-discussion"), and quiet time. FIAS also codes on a similar time-scale to ours (3 second intervals, rather than 1 second intervals) [44].

(unchanged by excluding students involved in the Presenter project). Of the students uninvolved with the Presenter project, only two reported never interacting with the instructor, while seven reported interacting with the instructor three or more times.

The directly measurable data from the class reinforce these reports. In the 456 distinct opportunities for student input through the SIP System (19 inputs $*$ 24 students), students responded 451 times, over 98% of the time. No more than one student skipped any single input or even exercise (which might encompass multiple input opportunities). Finally, only one student skipped more than one input opportunity, and that student skipped only two. It is difficult to tell whether answers were sincere and carefully considered. However, only three responses of 24 in the first exercise seem implausible, and one is from a Presenter project group member.[12] A more challenging exercise asked students to work in pairs to craft two questions to assess the danger of a hypothetical influenza vaccine (assuming the vaccine has been in use for 20 years). Although one student did not respond and one repeated another's question, the other 22 responses exhibited both sincerity and careful thought. The following is a list of all 23 responses as they were entered in class (including typographical oddities).[13] The top-level list is the list of "top-scoring" (according to students' ratings) members of each group. Sublists are the other members of a group. Both the grouping and scoring is based on student data provided in class, and this list corresponds to the view the instructor saw during class (*e.g.*, Figure 5.8).

1. What are the chances/how likely is it that influenza will return if we stop administering the vaccine?

2. - What unforseen ailments can be traced back to the use of the vaccine and how many deaths did they cause ?

[12]The exercise asked how many people die in the United States each year from influenza. One implausible response exceeded 10 million. Another was "42", a likely reference to The Hitchhiker's Guide to the Galaxy [2]. The third, "43", is suspiciously close to 42. The other responses, ranging between 150 and 500,000, are all plausible.

[13]One student's name in a response has been replaced with "<name>".

   (a) Are there long-term side effects of using the vaccine (e.g. developmental disorders)?
   (b) Of those who died from receiving the vaccine, what was the percentage of those who had compromised immune systems?
   (c) Are there any dangerous side-effects of the vaccine?
   (d) What are the rates of death or other serious side effects for specific groups (e.g., by death, by pre-existing health conditions) as a result of using this vaccine?

3. Why vaccinate a disease that doesn't exist?

4. What is the distribution of flu deaths versus the distribution of vaccine deaths?

5. Is there now a super flu bug that we have created by using the vaccine?

6. # fatalities as direct consequence of vaccine use?

   (a) Have subsequent studies during the last 20 years shown an increase in mortality rates in otherwise healthy individuals who were vaccinated (ie, $n > 5$)?

7. Is it more likely to die from the flu or from the vaccine?

8. what is the growth rate in deaths if we stop vaccinating (or some portion stops being vaccinated)?

   (a) How thorough is the eradication now?

9. 1. What other factors were involved in deaths due to the vaccine?

10. Did the 100 deaths caused by the flu vaccine occur in high-flu-risk groups?

   (a) Did th 100 deaths caused by the vaccine occur in high-risk groups? (<name> stole my question!!! This is totally unfair :-)

(b) Which demographic groups die from the vaccine?

11. Is it possible to reduce the risk of the vaccine through further technology advancements?

12. HAS THERE BEEN A RISE IN INCIDENTS OF OTHER DISEASES SINCE ITS FIRST USE?

13. - Were there as many deaths as predicted (over the past 20 years) due to the vaccine?

14. Is there a danger of influenza appearing again?

15. Positing that we've irradicated the flu from the human population, can it be transmitted though some other vector?

Our measurements of spoken student participation in class also reinforce students' self-reported participation numbers. Review of audio logs identified 72 discernible, non-overlapping student contributions directed to the class at large.[14] Of these only 12 were "incidental" comments: single-word interjections, jokes, or other comments "in passing". Unfortunately, students do not appear in the video log and are difficult to distinguish in the audio log; therefore, it is impossible to review the exact speakers and contents of the student contributions. However, the two observers of the class recorded 31 of the student contributions directed to the class as a whole. Of these, 25 were clearly content-related, 5 were clarification questions on exercises — such as this question regarding the 5 deaths per year attributed to a hypothetical influenza vaccine in an exercise: "Is that US or worldwide?" — and one was directly related to the SIP System itself. These 31 comments and questions came from 11 distinct students of the 24 student class. These data underrepresent the number of student contributions by a factor of two and almost certainly also underrepresent

[14]Comments made during "teacher-talk" were discarded as inaudible to the class as a whole.

student participation. Nonetheless, they do indicate that about half the students contributed publicly to the class, already an excellent participation rate.

Students' survey responses also supported the notion that SIP encouraged engagement and indicated some specific aspects of SIP that affected student participation levels. The end-of-class survey asked students two germane questions: "In what ways did the SIP system support or encourage your participation in the class?" and "In what ways did the SIP system hinder your participation in the class?" (See Appendix Section B.2 for question ordering and formatting.) I coded these responses using emergent categories named, where possible, using students' own response text and allowing multiple codings of a single response. 20 of the 24 students, 83%, cited some aspect of SIP that encouraged their participation *besides* its novelty (14 of 18, 78%, for those uninvolved with the Presenter project). Only 11 of 24 students, 46%, cited any aspect of SIP that hindered their participation (9 of 18, 50%, for those uninvolved with the Presenter project). Only one of these students cited no encouraging aspect of SIP. (Just over half of students in the class — 14 of the 24 students and 10 of the 18 uninvolved with the Presenter project — specifically reported that no aspect of SIP hindered their participation, yet 4 of these including one Presenter project member also mentioned some aspect of SIP that *did* hinder their participation but which they had discounted, such as the presence of distracting applications on the student devices.)

Design lessons from the survey

Students' responses to the experimental class survey validated some of SIP's design decisions and highlighted some of its strengths and weaknesses. The responses confirmed that SIP has the potential to create a highly interactive classroom atmosphere and, in particular, that embedding interactive exercises into the slides helped promote interaction in the class. Students' perceptions of anonymity in SIP demonstrated that easy specification and clear explanation of anonymity policies is critical to full and honest student participation, as suggested in Section 5.3.8. To our surprise, students' responses also suggested that including interactive exercises in the slides created a social obligation for students to participate; students felt “forced” to participate. Three primary lessons emerged from hindrances to interaction reported by students: (1) SIP should avoid presenting distractions to students or demanding their attention whenever possible, (2) SIP should provide obvious, direct benefit to students (*e.g.*, notetaking support) to encourage them to “buy in” to the system, and (3) the integrated support for interactions in SIP does not absolve instructors of justifying their choice of interactive pedagogy and explaining their expectations to students. Finally, the survey results verified that students can *learn* from a SIP class and not just enjoy or engage in it.

Analysis in this section focuses on two questions from the survey:

1. “In what ways did the SIP system support or encourage your participation in the class?” — summarized in Table 5.3

2. “In what ways did the SIP system hinder your participation in the class?” — summarized in Table 5.2

About one third of students reported that the inclusion of their data in the whole class’s result encouraged their participation. This validates our goal of stimulating participation by folding student contributions into the mediating slides. Including a student’s response in the public artifact and exposing the responses of the class as a whole encourage participation. Here is how several of the students put it: “I was more interested in the results knowing my answers were included”, “seeing what others thought was interesting.”, and “Voting online kept my attention; was interested in classwide results.”

Nearly as many students — 6 of 24, none involved with the Presenter project — cited the “participatory feel” of the class. One student simply said “I felt the class was to be participatory.” The others mentioned how the large number of activities and displays of student input encouraged participation. In particular, several of these students cite the interactive artifacts present on the slides (*e.g.*, “the questions, graphs, and questions certainly

Table 5.2: Summary of responses to the SIP survey prompt “In what ways did the SIP system hinder your participation in the class?” Note that this question followed the question about technical problems summarized in Table 5.1, and all but one student answered at least one of the open answer survey questions. (See Appendix Section B.2 for the entire survey text.) Percentages in the second column are for all students in the class (24). Those in the third column are for those students uninvolved with the Presenter project (18). The “DNR” category includes all blank responses (which are *not* included in the “none” category). Some responses were multiply coded. “*Any hindrance*” counts the responses with at least one code besides “none” and “DNR”. (Note that several responses said “no hindrances” but also listed minor hindrances.)

Category	% of students	% uninvolved with CP
none	58	56
distracting applications	17	22
distracted looking at my slides	17	17
DNR	13	11
reduced coverage	8	11
lack of student control	8	11
pace too fast	4	6
pair discussions	4	6
anti-climactic exercise	4	6
writing rather than talking	4	0
any hindrance	46	50

Table 5.3: Summary of responses to the SIP survey prompt “In what ways did the SIP system support or encourage your participation in the class?” Note that this question followed the question about hindrances summarized in Table 5.2, and all but one student answered at least one of the open answer survey questions. (See Appendix Section B.2 for the entire survey text.) Percentages in the second column are for all students in the class (24). Those in the third column are for those students uninvolved with the Presenter project (18). The two starred responses set off by double-lines list *hindrances* that were mentioned in response to this prompt. The “DNR” category includes all blank responses (which are *not* included in the “none” category). Some responses were multiply coded. “*Any encouragement*” counts the responses with at least one code besides “none”, “DNR”, “curiosity/playfulness”, and the two starred responses.

Category	% of students	% uninvolved with CP
sharing responses w/whole class	33	39
participatory feel	25	33
curiosity/playfulness	17	22
anonymity	25	17
forced to participate	17	17
helps follow instructor	8	11
neighbor discussion	8	6
none	4	6
DNR	4	6
*** “getting it wrong”	4	6
*** lack of student control	4	6
“Reflection/Analysis”	4	0
any encouragement	83	78

encouraged participation.”). These responses suggest that SIP transformed lecture slides from an artifact that encourages passivity (as described in Chapter 1 and documented by Gustafson and Kors [53]) into one that encourages interactivity and engagement.

Anonymity in SIP played an important and complex role, as it did in the Classroom Feedback System (see Section 4.4.4). 6 of the 24 students cited anonymity as an aspect of SIP that encouraged them to participate, *e.g.*, “The anonymity factor makes it easier — more comfortable — to give answers when you have no idea of the correct answer.” However, 3 of the 6 responses came from students involved in the Presenter project. One other student uninvolved with the Presenter project responded “the questions, graphs, and questions certainly encouraged participation. More importantly, they encouraged honest participation.” (The emphasis is from the original response.) Although this student does not explicitly mention anonymity, the response suggests a problem with public, non-anonymous participation.

Anonymity is both a strength and a weakness in SIP’s design. It is a strength in the sense that the instructor was able to design and students to perceive formative assessments as anonymous and, therefore, unthreatening. It is a weakness in the sense that the policy of anonymity on any given exercise is not visible to or verifiable by students. Furthermore, students were *not* truly anonymous in that the instructor could have (and did) later peruse responses with their associated login IDs. In this case, the association between ID and student identity was destroyed for privacy purposes associated with the research project. However, it seems unlikely that this level of “anonymity” is what the students meant when they cited anonymity as a comforting factor. For example, the 3 of 6 Presenter project members who cited anonymity as encouraging participation would surely be dismayed to realize that their “anonymous” answers would only be anonymous within the group of six IDs known to be associated with Presenter project members! Therefore, a critical design goal in the future must be to support clear establishment by instructors of policies with respect to important “design values” [46] such as anonymity *and* mechanisms to expose these design choices to students in a comprehensible and verifiable manner, as discussed in

Section 5.3.8.

A surprising encouraging factor cited by 4 of 24 students (3 of 18 uninvolved in the Presenter project) was a feeling that *not* participating was not an option. One of the students summarized this aspect this way: "well, I felt as though I really didn't have much choice but to participate when the lecture reached an activity. (I think this is a good thing.)" (The emphasis is from the original response.) Another student put this less forcefully: "there was no reason not to participate." Bearing in mind the research showing that students see prepared slides as an organizing influence in the class [53], these students' responses may validate the embedding of exercises into the class slides as discussed in Section 5.3.1. That is, when the slides dictate participation in an exercise, especially when students' own devices reinforce that dictum, the students feel participation is the only socially allowable response.

Only a few students cited the novelty of the SIP System as an encouraging factor. 4 of 24 students mentioned curiosity or playfulness as a factor that encouraged them to participate — *e.g.*, "I got to draw pretty pictures" and "playing w/the tablet PC" — although two of these students also cited other positive aspects of SIP. Unsurprisingly, none of the Presenter project team members, who had either heard about or seen demonstrations of SIP before, cited curiosity or playfulness as encouraging factors. Although curiosity and playfulness were the third most cited encouraging aspect of SIP for students uninvolved with the Presenter project, *many more* cited other encouraging aspects of SIP (14 of the 18 students uninvolved with SIP). This relatively small number of responses citing the novelty of the system suggests that novelty, and perhaps the related "Hawthorne effect", were relatively unimportant.

Three other encouraging aspects were cited infrequently. Two students each mentioned that SIP helped them follow the instructor's discussion (*i.e.*, by presenting a synchronized slide view on their own devices) and that SIP's exercises fostered participation by encouraging discussions with neighbors. One Presenter project member also answered only "Reflection, Analysis". Although the encouragement of neighborly discussion was at least partly due to the instructor’s prompting, a problem with anonymity in SIP may also help encourage discussion among neighbors. The problem is the potential for students to see each others’ screens and responses (as happened in the Classroom Feedback System, see Section 4.4.4). Although this “spying” might increase apprehension when responding and create cheating concerns for graded assessments, it also affords opportunities for valuable discussion.

By far the most cited aspect of the SIP System that *hindered* participation was the introduction of distractions. (Table 5.2 summarizes these results.) The two most cited hindering aspects were both distractions. 4 of 24 students (all uninvolved with the Presenter project) cited the availability of distracting applications, *e.g.*, “[SIP] helped (though Solitare [*sic*] was admittedly distracting)”. 4 of 24 students (3 of 18 uninvolved with the Presenter project) cited the temptation to focus on the student device rather than the instructor, public display, or other students, *e.g.*, “sometimes had my eyes glued to tablet screen instead of watching the instructor”. [15] One student uninvolved with the Presenter project also mentioned that the appearance of handwriting recognition results on the student device was “annoying” in response to an early survey question about technical problems in SIP. (Both the handwriting recognition problem and the survey question are described above.) Together, 7 distinct students (6 uninvolved with the Presenter project) cited distractions as problematic in SIP. The prevalence of this response emphasizes the importance of designing classroom interventions to be “quiet” technologies, at least when they are not intentionally the focus of attention. With Tablet PCs, a first step toward “quieting” might be to remove the keyboards and lay the devices flat on the tables so that they are less visible. Note however, that even instructors who use no instructional technology will face the problem of distracting technology as it enters the classroom in students’ pockets, just as instructors must now deal with crosswords, newspapers, and books.

The lack of student control over their own devices (*e.g.*, over slide transitions) was

[15]This may be related to the eye gaze problem discussed in Section 3.4.1 (page 61).

cited only twice in response to the survey question targeting hindrances but came up in response to the question about technical problems with SIP, a catch-all question asking for “other comments”. This issue even came up (as a hindrance) in response to the question about factors encouraging participation. In all, 5 distinct students (none involved with the Presenter project) mentioned frustration at the lack of student control over slide transitions, sharing of exercise responses, and note-taking.

These responses emphasize the importance of securing adoption not just from the instructor but from students as well. SIP, like all groupware [52], must ensure that all its users see direct benefit from the effort they invest in the system. In SIP, students invest effort into responding to exercises and suffer the opportunity cost of distraction from the main purpose of class (as described above). Natural benefits for students to expect in return are independent navigation (to review discussion points), control over their creations (their exercise responses), and note-taking. On the Tablet PC, note-taking is an especially natural expectation since the input device’s similarity to a normal pen clearly affords writing and since it usurps the desk space students would normally use for a notebook.

The latest versions of Presenter have begun to address student note-taking and student navigation [4], but these features are non-trivial. Difficult UI problems arise in allowing easy student control over navigation and inking and simultaneously resolving conflicts with the instructor’s navigation and inking. Also, both features lessen the slides’ power as mediating artifacts by allowing individual participants’ views of the slides to diverge.

A total of 6 distinct students (5 uninvolved with the Presenter project) cited five other hindrances to participation in SIP. These included concerns that SIP reduced “coverage” of course content, that pair discussions detracted from learning the course content, that one exercise was “anti-climactic” because it was too easy, that the pace of class was too fast, and that writing responses discouraged *also* speaking them aloud. Of these, two students cited the first, one cited the remainder, and the last was mentioned only by a Presenter project member. The first three concerns are common objections to the introduction of active learning into any classroom, and standard (though not universally effective) stratagems

Table 5.4: Summary of responses to the SIP survey prompt "What was the most important thing you learned from the class?" (See Appendix Section B.2 for question ordering and layout.) Note that all but one student answered at least one of the open answer survey questions. Percentages in the second column are for all students in the class (24). Those in the third column are for those students uninvolved with the Presenter project (18). The "DNR" category includes all blank responses (which are *not* included in the "nothing" category). Some responses were multiply coded. "*specific, non-system outcome*" counts the responses with at least one of the codes "learning goal", "lecture idea", and "discussion idea".

Category	% of students	% uninvolved with CP
DNR	25	28
learning goal	25	17
lecture idea	17	17
discussion idea	17	17
tool/interactivity	17	17
nothing	8	11
other	4	6
specific, non-system outcome	54	56

exist to address or allay these concerns [16, 68]. The fourth concern was paired with a suggestion of allowing students to navigate slides independently to cope with a fast-paced class, a suggestion discussed in detail above. The last concern echoes somewhat the more general issue of distractions presented by the SIP System. SIP should not become an alternative to other healthy and effective means of classroom participation. Ratto *et al.* discuss cultural mechanisms for establishing the primacy of spoken participation over an interactive instructional technology [85].

Finally, most students did believe they learned something from the SIP class. The survey included an adaptation of the standard "Minute Paper" Classroom Assessment Technique [13]: "What was the most important thing you learned from the class?" This question was coded using predetermined, rather than emergent, categories to focus on a few specific outcomes. The categories included: “learning goal”, indicating that a response directly addressed one of the learning goals described in the presentation and mentioned in Section 5.5.1; “lecture idea”, indicating an idea primarily addressed by lecture — *i.e.*, “teacher-talk” — during the class; “discussion idea”, indicating an idea primarily addressed by discussion — *i.e.*, “student-discussion” and “student-talk” — during the class; “tools/interactivity”, indicating lessons about active learning or the SIP tool itself; “nothing”; “did not respond”; and “other”, indicating a response that did not fit well into the other categories.[16] Table 5.4 summarizes the results. 6 students did not respond, 2 said they had learned nothing (both wrote “—”), and 3 others mentioned only lessons about the SIP System or vague learning outcomes. However, 13 of the 24 students cited something specific learned in the “lecture idea”, “learning goal”, or “discussion idea” categories (9 of 18 students uninvolved with the Presenter project). 6 of the 24 students (3 of the 18 uninvolved with the Presenter project) directly mentioned the learning goals established for the class. So, at least a few of the students recognized the critical, overarching lessons the instructor tried to convey. Within the lecture and discussion categories, responses were split evenly, indicating that both parts of the class were cogent enough to reach some students. Overall, SIP supported a successful learning environment, *i.e.*, one in which students *learned*.

Design lessons from the class

Experiences during the class also highlight some of SIP’s strengths. During the experimental class, SIP demonstrated its value as a tool for formative assessment by dispelling one of the instructor’s tightly-held misconceptions. As with the survey, the classroom experience also confirmed SIP’s potential to create a highly interactive class environment. Finally, experience with the “distributed human computation” (DHC) exercise (described in Section 5.2.3) demonstrated the high pedagogical value of the comparison, or “computation”,

[16]Note that for a normal class, categorizing responses by important threads and misconceptions that arose would be a more appropriate coding scheme.

phase of that exercise.

Using SIP, the instructor was able to disabuse himself of a tightly-held misconception and refocus his presentation to account for the change. As indicated in the reconstruction of the instructor's design process above (see Section 5.5.1), the instructor believed that students would misperceive many common modern risks and yet be unaware of their own misperception of risk. Several of the SIP exercises were therefore specifically designed to explore students' misperceptions. In point of fact, while individual students lacked knowledge about the risks of the various modern activities, substances, and diseases discussed during class (*e.g.*, influenza, peanut butter, and airplane and automobile travel), the class as a whole was quite knowledgeable about and accurate in its assessment of these risks. For example, by far the most common response to the question of how many people in the United States die from influenza each year was 35,000. This response is remarkably close to the Center for Disease Control's estimate of 36,000 [28] and stems from a recent NPR report several students heard. Because students demonstrated a clearer than expected grasp of modern risk, the instructor felt free to place less emphasis on modern risks and more emphasis on historical risk.

Besides this high-level contradiction of an instructor preconception, students also contributed substantively and frequently to the class, both through SIP and out loud. Page 168 shows the list of insightful questions students contributed through the SIP System for an exercise evaluating a hypothetical vaccine twenty years in the future. Substantive spoken contributions occurred throughout the class time. One example was a pithy and evocative name contributed by a student — the "killing innocent babies" factor — for how people's risk perception changes based on the endangered populations. During a different discussion, a student pointed out that comparing bus and airplane fatality rates would be fairer than comparing automobile and airplane fatality rates because of the training pilots receive and the commercial nature of most flights. Students also frequently contributed opinions, facts, and even corrections (*e.g.*, telling the instructor how to pronounce "mesothelioma", an asbestos-related illness).

Another surprising validation for SIP came in the form of students' satisfaction with the "distributed human computation" (DHC) exercise described in Section 5.2.3. One of the instructor's concerns before the class was that students would not see the rating stage of that exercise as pedagogically valuable, instead perceiving it as students doing the software's grunt work. However, when the instructor was wrapping up the DHC exercise and asked what else he should do to discuss it, one of the students responded that the most useful part of the exercise had been to look at and talk about other students' questions during the rating stage. The class as a whole nodded their assent to this comment. While instructors might still feel uncomfortable using students to perform computation in a DHC, this experience indicates that principled use of this style of exercise can be beneficial to students.

5.6 Contributions

The key contributions of this chapter are the prototype Structured Interaction Presentation System, a presentation system that integrates interactive exercises with slides and integrates exercise design into the slide design process; a conceptual framework for how this system fits into the collaborative classroom, as the class "score" mediating design and execution of interactive exercises; a set of design principles underpinning the system's design; and a detailed description of an experimental class run using SIP, which demonstrates SIP's potential and informs its design principles. This chapter also contributes examples of interactive presentations using SIP, including the innovative "distributed human computation" exercise pattern. Appendix A presents several more SIP exercises, including a full transcript of the experimental class's slide deck.

Chapter 6

RELATED WORK

The context of the work reported in this dissertation breaks down roughly into four categories: pedagogical and philosophical foundations of the dissertation as a whole and research related to each of the three systems developed — Classroom Presenter (Chapter 3), the Classroom Feedback System (Chapter 4), and the Structured Interaction Presentation System (Chapter 5). As both CFS and SIP extend Presenter, work related to Presenter is usually also related to the other two systems. Also, the work related to CFS and SIP, both of which support student interaction through computer-mediated communication, overlaps significantly. This chapter presents the pedagogical and philosophical background in Section 6.1, the related work for Presenter in Section 6.2, for CFS in Section 6.3, and for SIP in Section 6.4.

6.1 Pedagogical and Philosophical Background

Three key strands of pedagogical and philosophical thought shaped our work. First, Ann Brown's design experiments [24] provided our methodology. Second, the notion of course slides as "mediating artifacts" that participate in the cognitive process of learning and teaching provided a leverage point for intervention (as described in Chapter 2). Finally, "active learning" [16] as a style of instruction provided a goal and model for our interventions.

Ann Brown's design experiment methodology [24] provides a framework for crafting interventions that participate in a cognitive task. Design experiments recognize the integral contribution of students and instructors (*i.e.*, users) to the form and practice of an intervention and suggest a synthesis of quantitative and qualitative methods appropriate for comprehending such complex interactions. Previous work with this methodology, like Cole's "Fifth Dimension" [31], Stevens's "Video Traces" [101], and Brown's own reciprocal teaching [25] show how design experiments transition with facility between (often technical) development of the intervention and exploration of the social and cognitive nature of the engineered environment.

The notion of a "mediating artifact", introduced by Vygotsky [108], is central to all three systems described in this dissertation. In designing these systems, we consciously drew on the power of slides as an artifact mediating the learning and teaching process to enhance that process. More specifically, we view learning in the classroom as a distributed cognitive process engaged in collaboratively by students, teachers, and artifacts, in the spirit of Distributed Cognition [58]. Distributed Cognition rejects the notion that cognitive processes must be "bounded by skin and skull". The classic example of a distributed cognitive process is the piloting of a large naval ship, a cognitive task achieved through the coordinated action not only of multiple crew members but also through the mediating influence of navigation and communication tools and, through these, in collaboration with the tools' designers [60].

Similarly, we see classroom education as a cognitive process that encompasses students, instructor, and tools. In the interactive classrooms that we envision, these parties are engaged together in co-constructing knowledge. From this perspective, changes to critical tools clearly have the potential to change the nature and quality of the cognitive task. The strong effects of design decisions in previous presentation software — documented in [53] and polemicized frequently elsewhere [33, 104] — are evidence of the participation of one classroom artifact in the cognitive process of teaching and learning.

The third strand of pedagogical thought that influenced our work relates to "active learning". The goal of our interventions was to create a more flexible and interactive classroom environment and shift classes from a teacher-dominated, lecture-oriented style of instruction toward active learning. Bonwell and Eison introduced active learning [16] and a variety of teaching guides help instructors operationalize the concept [19, 30, 68, 70]. Various learning theories acknowledge interaction and active engagement as critical to the learning process. For example, in social learning theories (*e.g.*, Lave and Wenger's situated learning [64] and Vygotsky's zones of proximal development and artifact-mediated cognition [108]), interaction is an absolute prerequisite for meaningful learning; and in constructivist learning, students who engage and interact are more likely to build the personal knowledge structures necessary for their learning [73].

At a practical level, Angelo and Cross's Classroom Assessment Techniques [13] were a direct inspiration for CFS and especially for SIP. Mazur's notion of lecture thoughtfully admixed with interactive exercises [67] and other similar instructional designs [56] pointed the way toward the "bridge" design of Presenter, CFS, and SIP that enables instructors familiar with, and often expert at, the lecture paradigm to smoothly combine their strength at lecturing with more interactive strategies. Finally wireless networks offered the capacity to fold out-of-class technology-based active learning exercises (*e.g.*, Just-in-Time Teaching [80] and Carver's web quizzes and student-guided lectures [27]) and distance education's polls and quizzes (*e.g.*, in Centra [29] and WebEx [113]) into synchronous, collocated classrooms in the CFS and SIP systems.

6.2 Classroom Presenter

Digital ink technologies have evolved over time and include cameras, touch sensitive whiteboards, PDAs, Tablet PCs, and digital pens. Systems using these technologies support note taking and sharing [35, 57], real-time distributed conversation [62] and meetings [82, 100], and classroom presentation and capture [1, 59]. Other systems have pioneered alternative navigation techniques for presentation including zooming [51], physical [78], and intelligent vision-based navigation [45].

Several recent systems parallel Classroom Presenter's functionality, integrating ink with prepared slides for lecturing. Some commercial systems integrate ink and projected material on a modified whiteboard [41, 97]. Others support presentation to remote audiences [29, 113]. Lecturer's Assistant is one early research system that integrated slides with student and instructor writing in the classroom [26]. Similar systems exist for PDAs [76], tablets, and whiteboards [14, 23]. The Digital Lecture Hall project, like Presenter, focuses on creating a practically adoptable classroom presentation system [74, 88]. PowerPoint and Windows Journal can also be used to project and annotate material from the Tablet PC. Müller and Ottmann [75] and Brotherton [23] provide overviews of many such lecture presentation systems, though both focus on archiving.

Our project differs in technology and focus from these systems. The form factor of PDAs makes the presentation experience very different from a Tablet PC. Tablet and wireless technology has changed radically since some of the earlier projects, which allows Presenter to provide a very different experience for both instructors and students. Our focus on flexibility, interaction, and the slides as a mediating artifact has resulted in novel affordances, including embedded instructor notes and architectural support for student interaction. Finally, Classroom Presenter is designed to be deployable without a dedicated room or substantial financial investment, in contrast to higher-end systems like Classroom 2000/eClass that can require substantial commitment to deploy [117].

6.3 Classroom Feedback System

CFS builds on prior efforts in supporting classroom interaction but is unique in accommodating student-initiated feedback within a rich, shared context.

Many interaction systems — *e.g.*, ClassTalk [37, 38] — differ distinctly from our approach in CFS in that they target instructor-initiated input opportunities like quizzes. Section 6.4 focuses on these systems.

The ParcTab project's "tabarbitron" was perhaps the first wireless system for audience feedback during presentations [111]. Brittain's work with mobile phones [22] was inspirational for our work on CFS in that Brittain also aimed to create a simple, low-cognitive-load channel for communication. However, both these systems and other student-initiated feedback systems (*e.g.*, ClassInHand [109], TELEP [63], and Flatland [115]) provide only minimal context for feedback, such as time or slide number.

ActiveClass [85, 103] supports both instructor- and student-initiated interaction. Although ActiveClass lacks the rich context for feedback that CFS provides, it offers an innovative solution to the tradeoff between a small number of feedback categories (survey responses, in their case) and flexibility by allowing students to enter new categories which can then be used by other students. The ActiveClass research team has also performed thorough ethnographic investigations of classes of students with wireless, handheld devices which promise to inform our own future work with smaller form factors, wider adoption, and student devices [85, 103].

Non-technological approaches to increasing interaction in large classes range from advice on establishing a conducive classroom climate to larger scale changes in teaching method and class design (*e.g.*, active learning [16], mentioned above). While valuable in their own right, success of these methods relies heavily on the instructor's personality, talent, and time commitment, both for preparation outside class and for novel activities during class. CFS was designed provide an adoption path for instructors using a lecture style that begins to introduce active learning with minimal instructor and student training.

6.4 Structure Interaction Presentation System

A vast literature discusses systems for interactive exercises in class, including descriptions of computer- and non-computer-based systems and experiments with and anecdotes about use of such systems. Of key relevance to SIP are a few systems that inspired its development and guided its design principles and a few others that offer similar functionality. I will first discuss these systems and their use and then provide a sampling of relevant literature on other interaction systems.

The ClassTalk System [37, 38] directly inspired SIP's development. ClassTalk has evolved from a system based on wired networking of graphing calculators to support for simple wireless remotes and handheld computers. Throughout this process, its goal has been to support cycles of discovery, discussion, and lecture in synchronous collocated classrooms, embodying the active learning goals of progressive physics education championed, *e.g.,* by Mazur [67]. Unlike most other classroom interaction systems, ClassTalk supports not just closed response (*i.e.,* multiple choice) questions but also numerical response and even limited free text. ClassTalk supports automatically generated histograms of free text responses by sorting responses into preset bins according to substring matches with a keyword associated with each bin [37, 38]. Besides offering design flexibility to instructors, exercises based on free response questions may have greater pedagogical value than exercises with closed response questions [93].

ActiveClass [85, 103] informed SIP's design in several ways. First, SIP embraces integration with the instructor's presentation and design environments in part in response to ActiveClass's rejection of integration. While Truong *et al.* warn that sustainable adoption of educational interventions may preclude integration with expensive, commercial software [103], our work with SIP provided an opportunity to explore the benefits of integration that must be balanced against such concerns (see Section 5.3.1). Anecdotes of user-inspired innovations in ActiveClass (such as the instructor's explicit treatment of the system as a "virtual student" and students' co-opting of the question feature to answer other students' questions) led us to design for flexibility in SIP so that end users would have the freedom they need to innovate. Finally, ActiveClass's extensible polling mechanism — in which students in a poll may either vote for an existing option or introduce a new one — showed that carefully designed mechanisms could strike a balance between allowing open responses and supporting easy aggregation and analysis of responses, leading to our design of the Distributed Human Computation exercise pattern (see Section 5.2.3) [85, 103].

SIP adopted the metaphor of "conductor-of-orchestra" for interaction in technology-enabled classrooms espoused by Roschelle and Pea as part of the WILD project [87]. Roschelle and Pea also describe a variety of interaction systems in the same work. Pea and others later proposed the ClassSync Modeling Language (CML), a more detailed framework for WILD [21]. Although CML is as yet unimplemented, it represents a potential framework for specifying, understanding, and organizing exercises like those SIP supports.

Our experience with the Classroom Feedback System (described in Chapter 4) also informed our design of SIP. Instructors using CFS had trouble managing unexpected, student-initiated feedback in real-time during a class. While SIP certainly does not seek to "quash" student-initiated interaction, it does allow instructors to design and thoughtfully plan the flow of *instructor-initiated* interactions. The INTICE system similarly supports instructors' premeditated design of pedagogically valuable, instructor-initiated interactive exercises [93].

A full list of systems supporting interaction in the classroom is outside the scope of this dissertation. However, discussing a few representative systems is instructive.

The Xerox ParcTab [111] was an early handheld computer and included "tabarbitron", an interaction system [111, page 25]. Tabarbitron allowed "up/down" voting by the audience of a professional talk on the pace and quality of the talk. Interestingly, the results of these moving polls were displayed publicly during the talk. Many later systems support similar "up/down" moving polls [22, 85]. As student-initiated feedback, however, this is outside the scope of SIP.

A wide variety of systems support multiple choice or numeric questions and histogram displays of results in class. Academic systems include ClassInHand [109], the Digital Lecture Hall [74, 102], and ActiveClass [85]. Commercial systems include CPS [40], PRS [39], H-ITT [61], and TurningPoint [105, 106]. Many of these systems include important features, and researchers using them discuss interesting classroom practices, *e.g.*, device-swapping to support anonymity using TurningPoint [94]. However, most of the systems differ from SIP in supporting only closed response or numeric questions and simple histogram displays that are loosely, if at all, coupled to the class slides.[1]

[1]The Digital Lecture Hall supports freeform input and integration, but the results of its freeform inputs are simply presented to the instructor as a list and not aggregated or otherwise integrated into the presentation, unlike SIP's freeform inputs or the Digital Lecture Hall's multiple choice and numeric responses [74].

Of these, TurningPoint [105, 106] supports the tightest integration of slides and interactive exercises. TurningPoint allows design of interactive exercises built into PowerPoint presentations and, unlike SIP, executes within PowerPoint as well. As such, TurningPoint bears many similarities to SIP, including smooth combinations of static and interactive slide elements and use of PowerPoint navigation primitives to control timing of interactive exercises. TurningPoint also supports significantly more dynamic control of exercises than SIP (*e.g.*, keystrokes for re-running an exercise or changing the graphical summary of responses) and support for broader "learning management" concerns (*e.g.*, management of participant lists and generation of data reports). However, TurningPoint's model of an interactive exercise is more restrictive than SIP's: exercises generally must appear only on a single slide, all exercises use multiple choice input, and the range of graphical summaries is limited to pie charts, bar graphs, and numerical summaries. Furthermore, each exercise is associated with only a single graphical display [106]. SIP's support for interactive exercises constructed from simple, interactive building-blocks allows substantially more freedom in exercise design (see Section 5.3.3).

A number of instructors have devised non-computer-based systems for supporting classroom interaction. Of course, hand raising, standing, or shouting are natural voting mechanisms that require no equipment to support them. Most equipment-based strategies use public displays of some physical token which the instructor approximately aggregates by scanning the classroom (*e.g.*, Post-It Notes [55]). These non-computer-based systems are generally low-cost, and any class can probably make productive use of such techniques, including classes that use computer-based interaction systems. However, non-computer-based systems suffer from several drawbacks. The physical process of participation — instructions, movement, and paper-shuffling — can be confusing and time-consuming. Instructors can only report simple, estimated aggregates of the results (like the plurality winner of a poll), in contrast to complex aggregations like the line-graph of Appendix Figure A.7 and the correlated results of Appendix Figures A.7 and A.10. Tracking data across exercises and across classes is extremely difficult. Finally, the public displays of student “votes” in non-computer-based systems may discourage honest response, especially on sen-sitive topics.

Chapter 7

CONCLUSIONS

This dissertation shows how slides — viewed as mediating artifacts and manipulated as shared digital artifacts — can support and shape effective practices of instruction and learning. Each of the three systems described in Chapters 3–5 relies on the mediating nature of public, shared course slides to enhance the flexibility and interactivity of the lecture classroom.

In the Classroom Presenter project, we used design experiment methodology [24] to explore the potential of an old idea — putting ink in the context of slides — combined with the new affordances of digital, networked media. Presenter showed the substantial value of judiciously separating different participants' views of the mediating artifact (Section 3.2.2): instructor notes enhanced instructors' design process and eased the cognitive burden of instruction, a separate public view resolved the conflict between the desire to simplify and enlarge the public display and the need for extra control and information on students' and instructors' displays, and rapid and flexible control of ink over slides made the expression of concepts and ideas more effective and encouraged instructors to be responsive to student input. Furthermore, our iterative, user-guided design process enabled Presenter's users to contribute improvements to the system's interface. Users were responsible for the filmstrip, resizability of the filmstrip, the slide preview, the undo button, the "whiteout" pen, and for winnowing out flashy features lacking practical value. Each of these contributions improved Presenter as a system, but the stories behind them, related in this dissertation, contribute more broadly, showing researchers how to make their design processes open to users' innovations. Finally, our survey research documented Presenter's broad and successful adoption in real classes.

The Classroom Feedback System introduced a simple mechanism for student feedback that relies heavily on the mediating nature of the slides to help students articulate and instructors understand the feedback. Experiments with CFS showed its value for dramatically increasing student interaction. Furthermore, our qualitative research identified particularly effective patterns of use of CFS, including the "student-guided lecture" pattern that would not have been possible without computer support (Section 4.4.4). Both evaluations also showed important weaknesses of CFS — such as the difficulty instructors had responding on their feet to unexpected feedback — that show the limitations of this approach. Besides the CFS system itself, the challenges to interaction — especially the largely unexplored challenge of "feedback lag" (Section 4.1) — and design principles that led to CFS stand as a foundation for other computer-mediated communication systems.

The Structured Interaction Presentation System built on Roschelle and Pea's model of the instructor as "conductor-of-orchestra" [87] by incorporating interactive exercises into the class slides to create a class "score". Chapter 5 detailed our vision of class slides as score (Section 5.1) and also described how the SIP system fulfills this vision. Besides showing how relying on the mediating nature of slides can enhance both the design and execution process of interactive exercises, our work with SIP also introduced individual innovative exercises, especially the "Distributed Human Computation" style of exercise (Section 5.2.3). Although SIP has not yet been used in a real class, its use in an experimental class established its potential. SIP effectively supported the instructor's design process; students quickly learned their interface to SIP and made valuable contributions to the class and their own learning through the system; and the system proved technically sound enough for use in a real class. Appendix A presents examples of SIP presentations and exercises that further demonstrate the system's potential to enhance learning and instruction.

Of course, this dissertation also leaves open important and difficult questions about how to make even more effective use of slides as mediating artifact. Each of Chapters 3–5 leaves challenging questions for future research. The Classroom Presenter work exposed intriguing structures in the way that instructors use ink — *e.g.*, attentional marks [12] —

and raised the question of how digital manipulation can be used to support and enhance the educational efficacy of these structured uses. The successful patterns of interaction in the Classroom Feedback System — especially the surprising "student-guided lecture" pattern (Section 4.4.4) — lead to questions about how future systems could best exploit these successful patterns and how researchers can predict and understand which patterns of interaction will be successful under the new mechanisms and social norms established in computer-enhanced classrooms. The fledgling Structured Interaction Presentation System shows promise for supporting instructors' intentional design of interactive classes; however, we know little yet about how instructors and students will co-opt SIP in real classes. Experiments in real classes promise to expose exciting and unexpected aspects both of SIP and of the classroom, as with Presenter and CFS. Also, serious system design issues remain unsolved in SIP, such as providing a mechanism for specifying the connections between interactive elements on SIP slides that feels natural for computing novices yet maintains the flexibility that is among SIP's core strengths.

Technology *will* continue to change classroom education. Students will acquire new computing devices over the years that become smaller, more connected, and more capable. Computing devices will proliferate in our working spaces, filling more and more niches of information, adaptation, and control.

Using this technology to reach our pedagogical goals is our obligation as researchers. However, success in this venture is not a point but a process. Learners, teachers, and researchers together must engage in practice, reflect on theory, redevelop systems, and redesign pedagogy to chart the continuing course of educational technology. Research that ignores these participants or eschews these processes may founder in self-absorbed criticism or stray from its underlying learning goals. Research that embraces these participants and these processes will create both the future of educational technology and the future of pedagogy.

BIBLIOGRAPHY

[1] G. D. Abowd. Classroom 2000: an experiment with the instrumentation of a living educational environment. *IBM Systems Journal*, 38(4):508–530, 1999.

[2] Douglas Adams. *The Hitchhiker's Guide to the Galaxy*. Harmony Books, New York, NY, USA, 1979.

[3] Richard Anderson, Ruth Anderson, Crystal Hoyer, Beth Simon, Fred Videon, and Steve Wolfman. Lecture presentation from the Tablet PC. In *WACE'03: Workshop on Advanced Collaborative Environments*, Seattle, WA, USA, June 2003.

[4] Richard Anderson, Ruth Anderson, Beth Simon, Steven A. Wolfman, Tammy VanDeGrift, and Ken Yasuhara. Experiences with a Tablet PC based lecture presentation system in Computer Science courses. In Dan Joyce, Deborah Knox, Wanda Dann, and Tom Naps, editors, *Proceedings of SIGCSE'04: the 35^{th} SIGCSE technical symposium on Computer Science Education*, pages 56–60, Norfolk, VA, USA, March 2004. ACM Press.

[5] Richard Anderson, Ruth Anderson, Tammy VanDeGrift, Steven A. Wolfman, and Ken Yasuhara. The Presenter system for Tablet PC-based classroom presentations. Faculty poster at SIGCSE'03: the 34^{th} SIGCSE technical symposium on Computer Science Education, February 2003.

[6] Richard Anderson, Jay Beavers, Tammy VanDeGrift, and Fred Videon. Videoconferencing and presentation support for synchronous distance learning. In *Proceedings of FIE'03: the 33^{rd} ASEE/IEEE Frontiers in Education Conference*, Boulder, CO, USA, November 2003. IEEE Computer Society Press.

[7] Richard Anderson, Martin Dickey, and Hal Perkins. Experiences with Tutored Video Instruction for introductory programming courses. In Henry Walker, Renée McCauley, Judith Gersting, and Ingrid Russell, editors, *Proceedings of SIGCSE'01: the* 32^{nd} *SIGCSE technical symposium on Computer Science Education*, Charlotte, NC, USA, February 2001. ACM Press.

[8] Richard Anderson, Tammy VanDeGrift, Steven A. Wolfman, Ken Yasuhara, and Ruth Anderson. Interaction patterns with a classroom feedback system: Making time for feedback. In Gilbert Cockton and Panu Korhonen, editors, *Extended abstracts of CHI'03: the 2003 conference on Human factors in computing systems*, pages 880–881, Ft. Lauderdale, FL, USA, April 2003. ACM Press.

[9] Richard J. Anderson. A parallel algorithm for the maximal path problem. In *Proceedings of the STOC'85: the* 17^{th} *Annual ACM Symposium on the Theory of Computing*, pages 33–37, Providence, RI, USA, 1985. ACM Press.

[10] Richard J. Anderson, Ruth Anderson, Tammy VanDeGrift, Steven A. Wolfman, and Ken Yasuhara. Classroom presentation from the Tablet PC. In Vassilios Dagdilelis, Maya Satratzemi, Roger Boyle, Georgios Evangelidis, and David Finkel, editors, *Proceedings of ITiCSE'03: the* 8^{th} *annual conference on Innovation and technology in computer science education*, pages 238–238, Thessaloniki, Greece, June 2003. ACM Press.

[11] Richard J. Anderson, Ruth Anderson, Tammy VanDeGrift, Steven A. Wolfman, and Ken Yasuhara. Promoting interaction in large classes with computer-mediated feedback. In Wasson et al. [112], pages 119–123.

[12] Richard J. Anderson, Crystal Hoyer, Steven A. Wolfman, and Ruth Anderson. A study of digital ink in lecture presentation. In Elizabeth Dykstra-Erickson and Man-

fred Tscheligi, editors, *Proceedings of CHI'04: the 2004 conference on Human factors in computing systems*, pages 567–574, Vienna, Austria, April 2004. ACM Press.

[13] Thomas A. Angelo and K. Patricia Cross. *Classroom Assessment Techniques*. Jossey-Bass Publishers, San Francisco, CA, USA, 1993.

[14] Dave Berque, David K. Johnson, and Larry Jovanovic. Teaching Theory of Computation using pen-based computers and an electronic whiteboard. In Sally Fincher, Bruce Klein, Fintan Culwin, and Mike McCracken, editors, *Proceedings of ITiCSE'01: the 6^{th} annual conference on Innovation and technology in computer science education*, pages 169–172, Canterbury, UK, June 2001. ACM Press.

[15] Donald A. Bligh. *What's the use of lectures?* Jossey-Bass Publishers, San Francisco, CA, USA, 2000.

[16] C. Bonwell and J. Eison. Active Learning: Creating excitement in the classroom. ASHE ERIC Higher Education Report 1, The George Washington University School of Education & Human Development, Washington, D.C., USA, 1991.

[17] John Waite Bowers. Classroom communication apprehension: A survey. *Communication Education*, 35, October 1986.

[18] Roger Boyle, Martyn Clark, and Amruth Kumar, editors. *Proceedings of ITiCSE'04: the 9^{th} annual SIGCSE conference on Innovation and technology in computer science education*, Leeds, UK, June 2004. ACM Press.

[19] John D. Bransford, Ann L. Brown, and Rodney R. Cocking, editors. *How People Learn: Brain, Mind, Experience, and School*. National Academy Press, Washington, D.C., USA, 2000. Expanded Edition.

[20] Grant Braught and David Reed. Disequilibration for teaching the scientific method in computer science. In Gersting et al. [47], pages 106–110.

[21] John Brecht, Mark Chung, and Roy Pea. CML — The ClassSync Modeling Language. In Stahl [99].

[22] Eric Brittain. Personal Communication, August 2001.

[23] J. Brotherton. *Enriching Everyday Activities through the Automated Capture and Access of Live Experiences*. Ph.D. dissertation, Georgia Institute of Technology, Atlanta, GA, USA, September 2001.

[24] Ann L. Brown. Design experiments: Theoretical and methodological challenges in creating complex interventions in classroom settings. *The Journal of the Learning Sciences*, 2(2):141–178, 1992.

[25] Ann L. Brown and Joseph C. Campione. Students as researchers and teachers. In Herbert J. Walberg and James W. Keefe, editors, *Teaching for thinking*, pages 49–57. National Association of Secondary School Principals, Reston, VA, USA, September 1992.

[26] C. Buckalew and A. Porter. The Lecturer's Assistant. In Robert Beck and Don Goelman, editors, *Proceedings of SIGCSE'94: the* 25^{th} *SIGCSE technical symposium on Computer Science Education*, pages 193–197, Phoenix, AZ, USA, March 1994. ACM Press.

[27] MAJ Curtis A. Carver, MAJ Richard A. Howard, and COL William D. Lane. A methodology for active, student-controlled learning: Motivating our weakest students. In John Impagliazzo, Elizabeth Adams, and Karl J. Klee, editors, *Proceedings of SIGCSE'96: the* 27^{th} *SIGCSE technical symposium on Computer Science Education*, pages 195–199, Philadelphia, PA, USA, February 1996. ACM Press.

[28] The Centers for Disease Control and Prevention. CDC - Influenza (Flu) — The Disease. Online, December 2003. Retrieved June 14, 2004, from `http://www.cdc.gov/flu/about/disease.htm`.

[29] Centra Software. Centra Symposium - live interactive virtual learning, online training, and eLearning collaboration. Online, 2004. Retrieved July 17th, 2004 from `http://www.centra.com/products/symposium/info.asp`.

[30] Arthur W. Chickering and Zelda F. Gamson. Seven principles for good practice in undergraduate education. *AAHE Bulletin*, 39(7):3–7, March 1987.

[31] Michael Cole. Sustaining model systems of educational activity: Designing for the long haul. Paper presented at Symposium Honoring the Work of Ann Brown, January 2001.

[32] Alan Cooper and Robert Reimann. *About Face 2.0: The Essentials of Interaction Design*. Wiley Publishing, Inc., Indianapolis, IN, USA, 2003.

[33] Tom Creed. PowerPoint No! Cyberspace, Yes. *The National Teaching and Learning Forum*, 6(4), 1997.

[34] Allen Cypher, editor. *Watch what I do: Programming by demonstration*. MIT Press, Cambridge, MA, USA, 1993.

[35] Richard C. Davis, James A. Landay, Victor Chen, Jonathan Huang, Rebecca B. Lee, Francis Li, James Lin, Charles B. Morrey III, Ben Schleimer, Morgan N. Price, and Bill N. Schilit. Notepals: Lightweight note sharing by the group, for the group. In Williams and Altom [116], pages 338–345.

[36] Katherine Deibel. CATs for Computer Science. Online, May 2004. Retrieved

July 15, 2004, from `http://www.cs.washington.edu/homes/deibel/CATs/`.

[37] Robert J. Dufresne, William J. Gerace, William J. Leonard, Jose P. Mestre, and Laura Wenk. Classtalk: A classroom communication system for active learning. *Journal of Computing in Higher Education*, 7(2):3–47, March 1996.

[38] Robert J. Dufresne, William J. Gerace, Jose P. Mestre, and William J. Leonard. ASK-IT/A2L: Assessing student knowledge with instructional technology. UMPERG Technical Report PERG-2000#09-SEP#1-28pp, University of Massachusetts Physics Education Research Group, Amherst, MA, USA, September 2000.

[39] EduCue LLC. EduCue home page. Online, May 2004. Retrieved July 9, 2004, from `http://www.educue.com/Home.htm`.

[40] eInstruction. eInstruction — the global leader in interactive response systems. Online, 2003. Retrieved July 9, 2004, from `http://www.einstruction.com/`.

[41] S. Elrod, R. Bruce, R. Gold, D. Goldberg, F. Halasz, W. Janssen, D. Lee, K. McCall, D. Pedersen, K. Pier, J. Tang, and B. Welch. Liveboard: A large interactive display supporting group meetings, presentations and remote collaboration. In Penny Bauersfeld, John Bennett, and Gene Lynch, editors, *Proceedings of CHI'92: the 1992 conference on Human factors in computing systems*, pages 599–607, Monterey, CA, USA, May 1992. ACM Press.

[42] M. Ernst, T. Millstein, and D. Weld. Automatic SAT-compilation of planning problems. In *Proceedings of IJCAI'97: the* 15^{th} *International Joint Conference on Artificial Intelligence*, pages 1169–1176, Nagoya, Japan, August 1997. Morgan Kaufmann.

[43] Polly A. Fassinger. Understanding classroom interaction: Students' and professors' contributions to students' silence. *Journal of Higher Education*, 66(1):82–96, 1995.

[44] Ned A. Flanders. *Analyzing Teaching Behavior*. Addison-Wesley, Menlo Park, CA, USA, 1970.

[45] David Franklin. *The Intelligent Classroom: Competent Assistance in the Physical World*. Ph.D. dissertation, Northwestern University, Evanston, IL, USA, June 2001.

[46] Batya Friedman, Peter Kahn, and Alan Borning. Online, June 2003. Retrieved June 15, 2004, from `http://www.ischool.washington.edu/vsd/vsd-theory-methods-draft-june2003.pdf`. Previous version available as UW CSE TR 02-12-01.

[47] Judith Gersting, Henry M. Walker, and Scott Grissom, editors. *Proceedings of SIGCSE'02: the* 33^{rd} *SIGCSE technical symposium on Computer Science Education*, Cincinnati, KY, USA, February 2002. ACM Press.

[48] Joel Geske. Overcoming the drawbacks of the large lecture class. *College Teaching*, 40(4):151–154, 1992.

[49] Barry Glassner. *The Culture of Fear: Why Americans Are Afraid of the Wrong Things*. Basic Books, New York, NY, USA, 1999.

[50] Maryellen Gleason. Better communication in large courses. *College Teaching*, 34(1):20–24, 1986.

[51] Lance Good and Benjamin B. Bederson. CounterPoint: Creating Jazzy interactive presentations. HCIL Tech Report #2001-03, University of Maryland, College Park, MD, USA, 2001.

[52] Jonathan Grudin. Groupware and social dynamics: Eight challenges for developers. *Communications of the ACM*, 37(1):92–105, January 1994.

[53] K. K. Gustafson and K. Kors. A Vision of the Future: Remodeling the Knowledge Architecture of the University of Washington. Technical Report PETTT-03-PT-01, Program for Educational Transformation Through Technology, University of Washington, Seattle, WA, USA, January 2003.

[54] Carl Gutwin and Reagan Penner. Improving interpretation of remote gestures with telepointer traces. In Elizabeth F. Churchill, Joe McCarthy, Christine Neuwirth, and Tom Rodden, editors, *Proceedings of CSCW'02: the ACM conference on Computer supported cooperative work*, pages 49–57, New Orleans, LA, USA, November 2002. ACM Press.

[55] Chad Hanson. Silence and structure in the classroom: From seminar to town meeting via 'Post-it's. *National Teaching and Learning Forum Newsletter*, 9(6), October 2000.

[56] William L. Heward. Guided notes — improving the effectiveness of your lectures. *Tomorrow's Professor*, 495, June 2003.

[57] Loring Holden, Tim Miller, and Bob Zeleznik. ReMarkable Texts. Online, (n.d.). Retrieved July 30, 2004 from `http://ReMarkableTexts.org`.

[58] James Hollan, Edwin Hutchins, and David Kirsh. Distributed cognition: Toward a new foundation for human-computer interaction research. *ACM Transactions on Computer-Human Interaction*, 7(2):174–196, 2000.

[59] Wolfgang Hürst, Gabriela Maass, Rainer Müller, and Thomas Ottmann. The "authoring on the fly" system for automatic presentation recording. In Marilyn (Mantei)

Tremaine, editor, *Extended abstracts of CHI'01: the 2001 conference on Human factors in computing systems*, pages 5–6, Seattle, WA, USA, March 2001. ACM Press.

[60] Edwin Hutchins. *Cognition in the Wild*. The MIT Press, Cambridge, MA, USA, 1994.

[61] Hyper-Interactive Teaching Technology. H-ITT (Classroom Remote System). Online, (n.d.). Retrieved June 17, 2004 from `http://www.h-itt.com/`.

[62] Alastair Iles, Daniel Glaser, Matthew Kam, and John Canny. Learning via distributed dialogue: Livenotes and handheld wireless technology. In Stahl [99], pages 408–417.

[63] Gavin Jancke, Jonathan Grudin, and Anoop Gupta. Presenting to local and remote audiences. Technical Report MSR-TR-99-71, Microsoft Research, Redmond, WA, USA, September 1999.

[64] Jean Lave and Etienne Wenger. *Situated Learning: Legitimate Peripheral Participation*. Cambridge University Press, New York, NY, USA, 1991.

[65] Marcia C. Linn, Douglas B. Clark, and Jim D. Slotta. WISE Design for Knowledge Integration. *Science Education*, 87(4):517–538, 2003. Special Issue.

[66] Rachel Maines. *Asbestos and Fire: Technological Tradeoffs and the Body at Risk, 1870-1990*. Rutgers University Press, New Brunswick, NJ, USA, 2005 (forthcoming).

[67] Eric Mazur. *Peer Instruction: A User's Manual*. Prentice Hall, Upper Saddle River, NJ, USA, 1997.

[68] Jeffrey J. McConnell. Active learning and its use in computer science. In Boots Cassel, editor, *Proceedings of ITiCSE'96: the 1^{st} conference on Integrating technology into computer science education*, pages 52–54, Barcelona, Spain, June 1996. ACM Press.

[69] Wilbert J. McKeachie. Research on college teaching. *Journal of Educational Psychology*, 82(2):189–200, June 1990.

[70] Wilbert J. McKeachie. *Teaching Tips: Strategies, Research, and Theory for College and University Teachers*. Houghton Mifflin, Boston, MA, USA, 10th edition, 1999.

[71] John McMurry. *Fundamentals of Organic Chemistry*. Brooks/Cole Publishing Company, Pacific Grove, CA, USA, 4^{th} edition, 1998.

[72] Microsoft Research Learning Experience Project. Online, 2004. Retrieved June 11, 2004, from `http://www.conferencexp.net/community/`.

[73] Jennifer A. Moon. Reflection in learning — some fundamentals of learning, part 1. In *Reflection in Learning and Professional Development, Theory and Practice*, chapter 9, pages 103–119. Kogan Page, Sterling, VA, USA, new ed edition, January 2001.

[74] Max Mühlhäuser and Christoph Trompler. Digital Lecture Halls Keep Teachers in the Mood and Learners in the Loop. In *Proceedings of E-Learn 2002*, pages 714–721, Montreal, QC, Canada, October 2002. Association for the Advancement of Computing in Education (AACE).

[75] Rainer Müller and Thomas Ottmann. Electronic note-taking, systems, problems, and their use at universities. In *Information Technologies for Education & Training*, pages 121–138. Springer-Verlag, 2002.

[76] Brad A. Myers. Using handhelds and PCs together. *Communications of the ACM*, 44(11):34–41, 2001.

[77] Michael R. Neer. The development of an instrument to measure classroom apprehension. *Communication Education*, 36(2):154–166, April 1987.

[78] Les Nelson, Satoshi Ichimura, Elin Rønby Pedersen, and Lia Adams. Palette: a paper interface for giving presentations. In Williams and Altom [116], pages 354–361.

[79] Peter Norvig. The Gettysburg Powerpoint Presentation. PowerPoint presentation, online, (n.d.). Retrieved June 16, 2004 from `http://www.norvig.com/Gettysburg/`.

[80] Gregor Novak, Andrew Gavrin, Wolfgang Christian, and Evelyn Patterson. *Just-In-Time Teaching : Blending Active Learning with Web Technology*. Prentice Hall, Upper Saddle River, NJ, USA, 1999.

[81] Ian Parker. Absolute PowerPoint: Can a software package edit our thoughts? *The New Yorker*, pages 76–81, May 2001. Annals of Business Section.

[82] Elin Rønby Pedersen, Kim McCall, Thomas P. Moran, and Frank G. Halasz. Tivoli: an electronic whiteboard for informal workgroup meetings. In Bert Arnold, Gerrit van der Veer, and Ted White, editors, *Proceedings of CHI'93: the 1993 conference on Human factors in computing systems*, pages 391–398, Amsterdam, The Netherlands, April 1993. ACM Press.

[83] Jean Piaget. *Adaptation and intelligence: organic selection and phenocopy*. University of Chicago Press, Chicago, IL, USA, 1980. Translation by Stewart Eames from original text "Adaptation vitale et psychologie de l'intelligence".

[84] Craig Prince and Richard Anderson. Personal Communication, June 2004.

[85] Matthew Ratto, R. Ben Shapiro, Tan Minh Truong, and William G. Griswold. The activeclass project: Experiments in encouraging classroom participation. In Wasson et al. [112], pages 477–486.

[86] Everett M. Rogers. *Diffusion of Innovations*. Free Press, New York, NY, USA, 4th edition, 1995.

[87] Jeremy Roschelle and Roy Pea. A walk on the WILD side: How wireless handhelds may change CSCL. In Stahl [99], pages 51–60.

[88] Guido Rößling, Christoph Trompler, Max Mühlhäuser, Susanne Köbler, and Susanne Wolf. Enhancing classroom lectures with digital sliding blackboards. In Boyle et al. [18], pages 218–222.

[89] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1995.

[90] Roger Säljö. Representational tools and the transformation of learning. Keynote address at *CSCL'03: Computer Support for Collaborative Learning*, Bergen, Norway, June 15, 2003.

[91] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. IETF RFC 3550, July 2003. Retrieved June 11, 2004, from `http://www.ietf.org/rfc/rfc3550.txt`.

[92] Sarah Schwarm and Tammy VanDeGrift. Using classroom assessment to detect students' misunderstanding and promote metacognitive thinking. In P. Bell, R. Stevens, and T. Satwicz, editors, *Keeping Learning Complex: The Proceedings of the Fifth International Conference of the Learning Sciences (ICLS)*. Erlbaum, Mahway, NJ, USA, 2002.

[93] Jeffrey A. Siegel, Kathy J. Schmidt, and Justin Cone. INTICE — Interactive Technology to Improve the Classroom Experience. In *Proceedings of the 2004 American Society for Engineering Education Annual Conference & Exposition*, Washington, D.C., USA, June 2004. American Society for Engineering Education.

[94] Stephen E. Silliman and Leo McWilliams. Observations on benefits/limitations of an audience response system. In *Proceedings of the 2004 American Society for Engineering Education Annual Conference & Exposition*, Washington, DC, USA, June 2004. American Society for Engineering Education.

[95] Beth Simon, Richard Anderson, and Steve Wolfman. Activating computer architecture with Classroom Presenter. In *WCAE'03: Workshop on Computer Architecture Education*, San Diego, CA, USA, June 2003.

[96] Beth Simon, Ruth Anderson, Crystal Hoyer, and Jonathan Su. Preliminary experiences with a Tablet PC based system to support active learning in computer science courses. In Boyle et al. [18], pages 213–217.

[97] SMART Technologies Inc. The SMART Board. Online, 2004. Retrieved July 14, 2004 from `http://www.smarttech.com/`.

[98] T. Speakman, J. Crowcroft, J. Gemmell, D. Farinacci, S. Lin, D. Leshchiner, M. Luby, T. Montgomery, L. Rizzo, A. Tweedly, N. Bhaskar, R. Edmonstone, R. Sumanasekera, and L. Vicisano. PGM Reliable Transport Protocol Specification. IETF RFC 3208, December 2001. Retrieved June 11, 2004, from `http://www.ietf.org/rfc/rfc3208.txt`.

[99] Gerry Stahl, editor. *Proceedings of CSCL'02: the International Conference on Computer Support for Collaborative Learning*, Boulder, CO, USA, January 2002. Lawrence Erlbaum.

[100] M. Stefik, G. Foster, D. Bobrow, K. Kahn, S. Lanning, and L. Suchman. Beyond the chalkboard: Computer support for collaboration and problem solving in meetings. *Communications of the ACM*, 30(1):32–47, 1987.

[101] Reed Stevens and Rogers Hall. Seeing Tornado: How Video Traces mediate visitor understandings of (natural?) spectacles in a science museum. *Science Education*, 18(6):735–748, 1997.

[102] C. Trompler, M. Mühlhäuser, and W. Wegner. Open Client Lecture Interaction: An Approach to Wireless Learners-in-the-Loop. In *Proceedings of ICNEE'02: the* 4^{th} *International Conference on New Educational Environments*, pages 43–46, Lugano, Switzerland, May 2002.

[103] Tan Minh Truong, William G. Griswold, Matthew Ratto, and Susan Leigh Star. The ActiveClass Project: Experiments in encouraging classroom participation. Technical Report CS2002-0715, University of California, San Diego, Department of Computer Science and Engineering, La Jolla, CA, USA, 2002.

[104] Edward Tufte. The cognitive style of powerpoint, 2003.

[105] Turning Technologies, LLC. Audience response system, group response system, used in interactive meetings and interactive powerpoint presentations — turning technologies. Online, 2002. Retrieved July 9, 2004, from `http://www.turningtechnologies.com/`.

[106] Turning Technologies, LLC. *TurningPoint 2003 User Manual*, December 2003.

[107] Fred Videon. WebViewer. Online, (n.d.). Retrieved June 11, 2004, from `http://www.cs.washington.edu/education/dl/confxp/webviewer.html`.

[108] Lev S. Vygotsky. *Mind in society: The development of higher psychological processes*. Harvard University Press, Cambridge, MA, USA, 1978.

[109] Wake Forest University Information Systems Research and Development Team. ClassInHand. Online, May 2004. Retrieved July 17th, 2004 from `http://classinhand.wfu.edu/`.

[110] Barbara E. Walvoord and Virginia Johnson Anderson. *Effective Grading*. Jossey-Bass, San Francisco, CA, USA, 1998.

[111] Roy Want, Bill N. Schilit, Norman I. Adams, Rich Gold, and Karin Petersen. The ParcTab ubiquitous computing experiment. Technical Report CSL-95-1, Xerox Palo Alto Research Center, March 1995.

[112] Barbara Wasson, Sten Ludvigsen, and Ulrich Hoppe, editors. *Proceedings of CSCL'03: the International Conference on Computer Support for Collaborative Learning*, Bergen, Norway, June 2003. Kluwer.

[113] WebEx Communications. E-learning, traning center, virtual classroom: WebEx. Online, (n.d.). Retrieved July 17, 2004 from `http://www.webex.com/services/online-training-features.html`.

[114] WebMD, Inc. WebMD — Trustworthy, Credible, and Timely Health Information. Online, (n.d.). Retrieved June 14, 2004, from `http://www.webmd.com/`.

[115] Stephen A. White, Anoop Gupta, Jonathan Grudin, Harry Chesley, Gregory Kimberly, and Elizabeth Sanocki. Evolving use of a system for education at a distance. Technical Report MSR-TR-98-61 (Revised), Microsoft Research, April 1999.

[116] Marian G. Williams and Mark W. Altom, editors. *Proceedings of CHI'99: the*

1999 conference on Human factors in computing systems, Pittsburgh, PA, USA, May 1999. ACM Press.

[117] Laura R. Winer and Jeremy Cooperstock. The “Intelligent Classroom”: changing teaching and learning with an evolving technological environment. *Computers & Education*, 38(1-3):253–266, 2002.

[118] Wireless Projector Adapter. Online, (n.d.). Homepage for Projector People’s WiJET Plug-and-Play Wireless Projector Adapter, retrieved June 11, 2004, from `http://www.projectorpeople.com/screens/wijet.asp`.

[119] Steven A. Wolfman. Making lemonade: Exploring the bright side of large lecture classes. In Gersting et al. [47], pages 257–261.

[120] Douglas Zongker. *Creating Animation for Presentations*. Ph.D. dissertation, University of Washignton, Seattle, WA, USA, 2003.

Appendix A

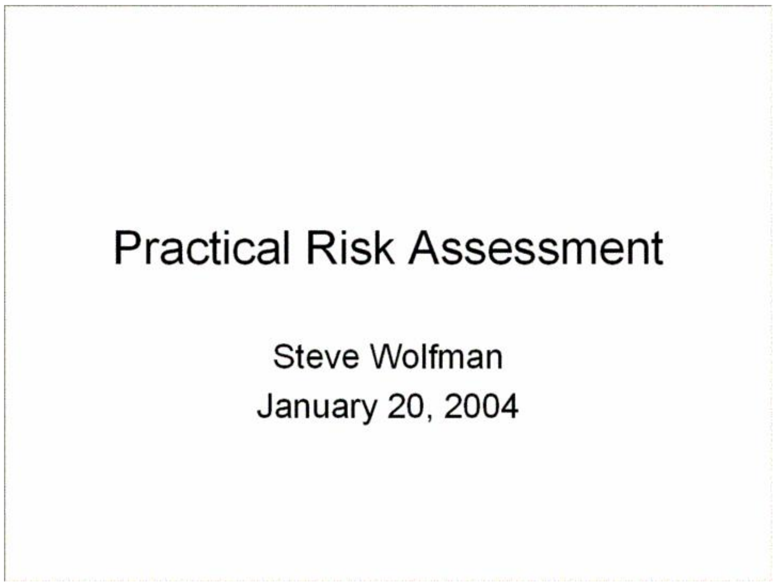
SAMPLE STRUCTURED INTERACTION PRESENTATIONS

This appendix provides a set of sample presentations generated with the Structured Interaction Presentation (SIP) System. Each presentation appears in its own section. The section begins with a summary of the presentation's content and a description of how, why, and by whom the presentation was designed. Next, the section includes slide-by-slide screenshots of the presentation. However, multiple views may be relevant for each slide — different views for the different roles and for the process of designing the presentation and executing (delivering) it in class. In general, there may be six ($3 * 2$) such views, the cross-product of instructor/student/public roles and design/execution phases. For consistency, each slide's screenshot set will begin with the public, execution view, and no two sets will share a single page. However, a set will include only as many of the six possible slides as needed to convey both its design and execution. General notes about a slide will appear in the caption of the first screenshot, although notes specific to other views may appear on later screenshots. In rare cases, a slide may even require more than six screenshots in order to demonstrate dynamic behavior. Student slides often have data filled in to make the slide's function clearer; while this data often matches data actually entered by students, it is artificial and for demonstration purposes only.

The first caption of each set will also begin with a set of codes indicating which slides are present of: public execution view (PE), public design view (PD), student execution view (SE), student design view (SD), instructor execution view (IE), and instructor design view (ID). Slides will appear in the order listed in the code, which will always be the same as the order they were introduced here (except that some slides may not be included). For example, the code "Slides: PE/SE/SD/IE/ID" indicates that the execution views for all three roles (public, student, and instructor) will be shown as will the design views for the student and instructor roles. (Note that “PE” will always be included.) Each screenshot will also indicate its slide number in the presentation, and each one besides the first in a set (which is always “PE”) will indicate which view it is (*e.g.*, “PD”, “SE”, *etc.*).

A.1 Practical Risk Assessment

I designed the “Practical Risk Assessment” presentation for the experimental SIP class described in Section 5.5. This is a full-class presentation of 20 slides. It touches on both modern and historical aspects of risk assessment in a public health context. It is the only one of the presentations in this appendix that has been used in a realistic classroom scenario (though several have been tried in group meetings). See Section 5.5 for details on the learning goals motivating this presentation and the process of its design. The execution screenshots included below use the actual student data from the experimental class. Note, however, that one bug has been fixed since the class which affected the data displayed on Slides 2 and 3 somewhat (Figures A.2–A.9). The bug is described in Section 5.5.2 (page 161).

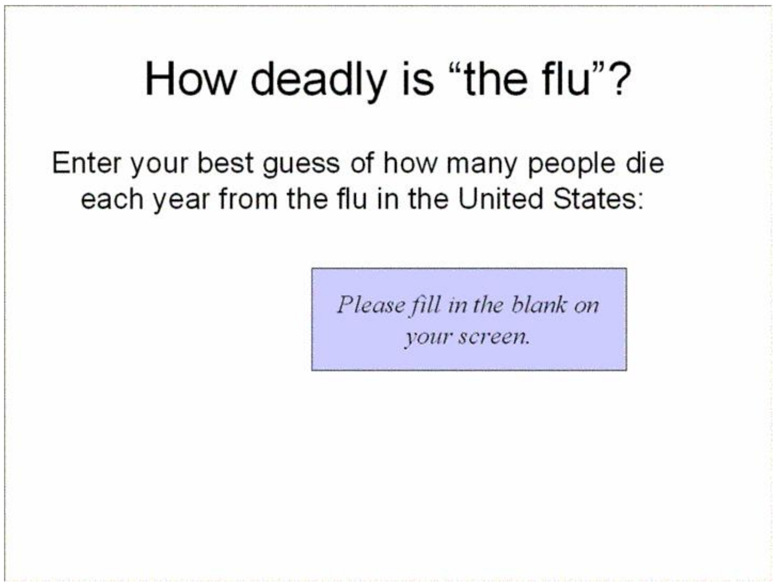


Practical Risk Assessment

Steve Wolfman

January 20, 2004

Figure A.1: Slide 1. Slides: PE. Title slide. (Refer to the beginning of this appendix for an explanation of the “Slides” codes.)

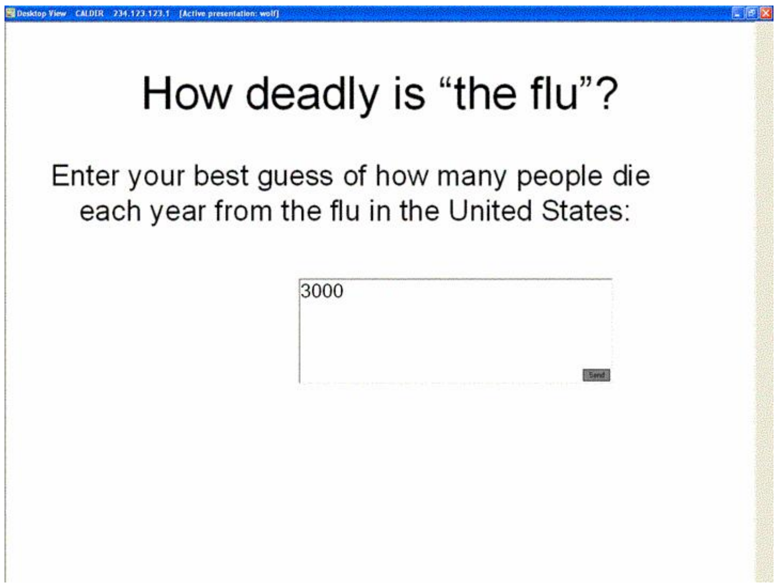


How deadly is "the flu"?

Enter your best guess of how many people die each year from the flu in the United States:

Please fill in the blank on your screen.

Figure A.2: Slide 2. Slides: PE/SE/SD/IE/ID. This public display prompts students to participate in their first exercise, which is intended to get students interested in risk assessment and show them that they have much to learn on the topic. As Figure A.3 shows, students' devices show a text input blank where the public display shows the shaded message prompt. As with most exercises in this class, the instructor suggested that students work together with their neighbors.

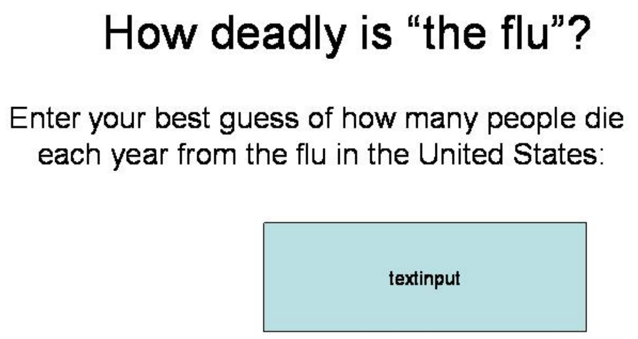

Desktop View CALDIR 234.123.123.1 [Active presentation: wolf]
Send

How deadly is “the flu”?

Enter your best guess of how many people die each year from the flu in the United States:

3000

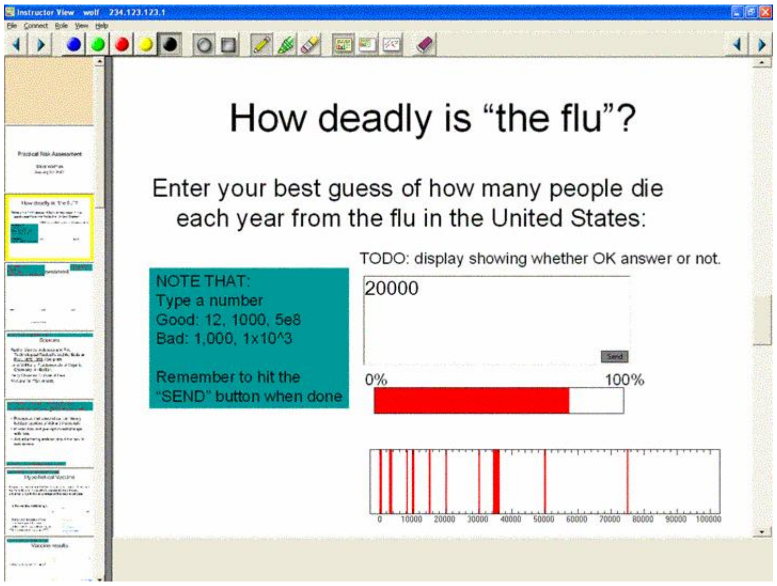
Figure A.3: Slide 2, SE. This student has typed “3000” as her answer but has not yet sent it, as indicated by the dark gray color of the “Send” button in the lower right. The button pales and becomes inactive when the text has been sent. Allowing students to send responses (rather than taking them as students type) improves performance and gives students control of their “draft” responses, hopefully making them more comfortable participating.



How deadly is “the flu”?

Enter your best guess of how many people die each year from the flu in the United States:

Figure A.4: Slide 2, SD. The shaded textbox “textinput” is the placeholder for the text input widget of Figure A.3. Henceforth, design screenshots this simple will not be shown.



How deadly is “the flu”?

Enter your best guess of how many people die each year from the flu in the United States:

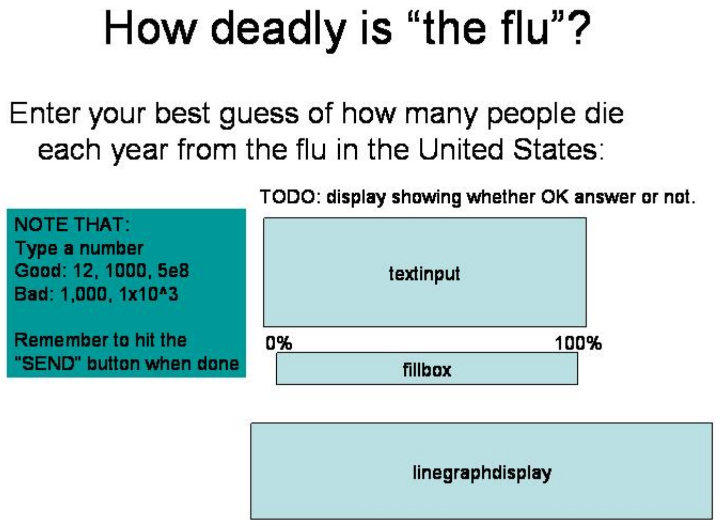
TODO: display showing whether OK answer or not.

NOTE THAT:
Type a number
Good: 12, 1000, 5e8
Bad: 1,000, 1x10^3

Remember to hit the “SEND” button when done

20000

Figure A.5: Slide 2, IE. The instructor responded “20000”. The two red displays in the lower right are SIP widgets. The filling bar shows the percentage of students who have responded, somewhere around 80%. (Because of the presence of a few spare SIP clients in case of failure of a student’s computer, the bar will never reach 100%, but almost all students responded to most exercises.) Similar “progress bars” are used on many slides. The red graph is a preview for the instructor of the graph in Figure A.7. The shaded box on the left is an instructor note. The instructor chose ugly colors for these so he would notice them. (I apologize to the color-blind reader for the particularly hideous color choice of red-on-green for several notes.) The note reminds of instructions to give the students. Finally, the text starting with “TODO” is a note the instructor left during development of the presentation but forgot to delete. Fortunately, he restricted all such notes to instructor visibility; so, students never saw the mistake.



How deadly is “the flu”?

Enter your best guess of how many people die each year from the flu in the United States:

TODO: display showing whether OK answer or not.

NOTE THAT:
Type a number
Good: 12, 1000, 5e8
Bad: 1,000, 1x10^3

Remember to hit the "SEND" button when done

Figure A.6: Slide 2, ID. As with Figure A.4, SIP widgets in this screenshot are represented by shaded textbox placeholders. In practice, these placeholders required some configuration to make them work. For example, the “linegraphdisplay” included several queries to collect data from the “textinput” widget and sift out non-numeric responses. Note also that the “textinput” widget here is the *same* one as in Figure A.4. The widget is simply restricted to both student *and* instructor visibility. Again, relatively simple design screenshots like these will not be included in the future.

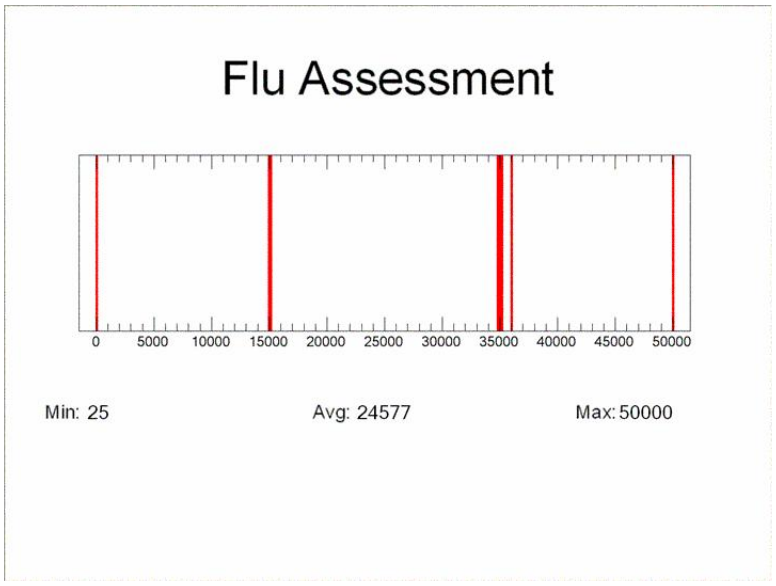
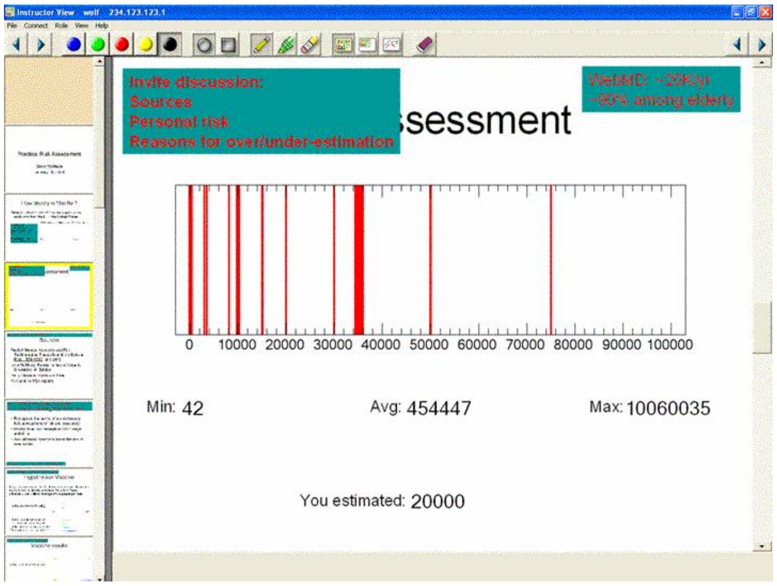


Figure A.7: Slide 3. Slides: PE/IE/ID. The aggregated results from the exercises of Slide 2. The graph shows how all students in the class responded, with wider bars for clusters of answers (*e.g.*, at 35,000). The minimum, average, and maximum of the responses is shown below. (The average is thrown off by one extremely large response, the maximum shown.) The student's display looks the same as the public display except that it includes the "You estimated" widget described in Figure A.8. During class, this slide prompted much discussion about why there was such a broad range of responses, why so many people selected 35,000 (because of a National Public Radio report), and what it meant to clearly assess personal risk with respect to influenza.


Instructor View wolf 234.123.123.1
File Connect Role View Help
0
10000
20000
30000
40000
50000
60000
70000
80000
90000
100000

Invite discussion:
Sources
Personal risk
Reasons for over/under-estimation

WebMD: ~20K/yr
~90% among elderly

ssessment

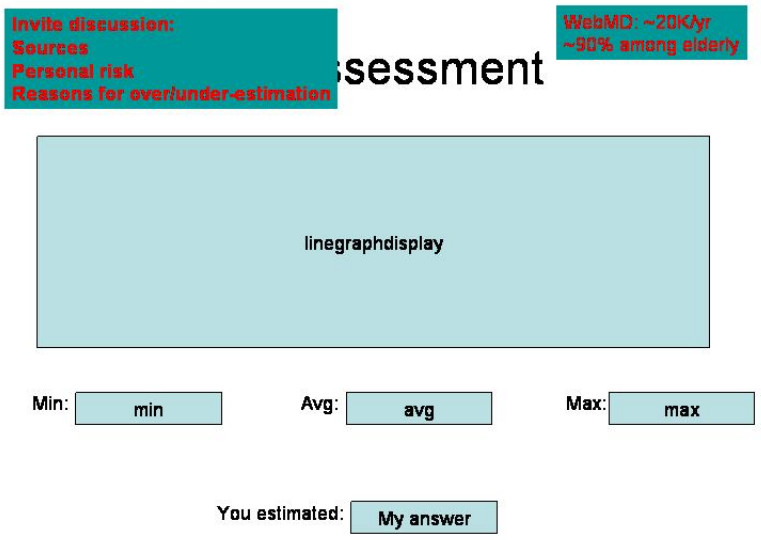
Min: 42

Avg: 454447

Max: 10060035

You estimated: 20000

Figure A.8: Slide 3, IE. The instructor has two more instructor notes near the top of the slide. The one on the left reminds of discussion points to bring up while the one on the right reminds of two facts the instructor wanted to introduce. Also, the instructor's display shows his own response at the bottom, the same as the one he entered in Figure A.5.

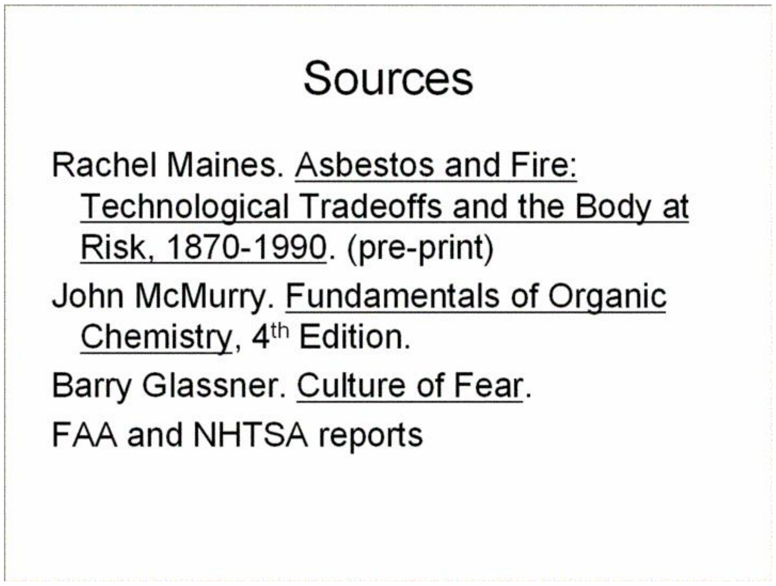


Invite discussion:
Sources
Personal risk
Reasons for over/under-estimation

ssessment

WebMD: ~20K/yr
~90% among elderly

Figure A.9: Slide 3, ID. The instructor changed four of the five widget placeholders' text to remind him of the widgets' functions, *e.g.*, "avg" for the widget that shows an average. Note that the four widgets toward the bottom are *all* text display widgets. Their placeholder text does not affect their type. The instructor used SQL queries to calculate the text he wanted, *e.g.*, the SQL "AVG" function for the "avg" widget.



Sources

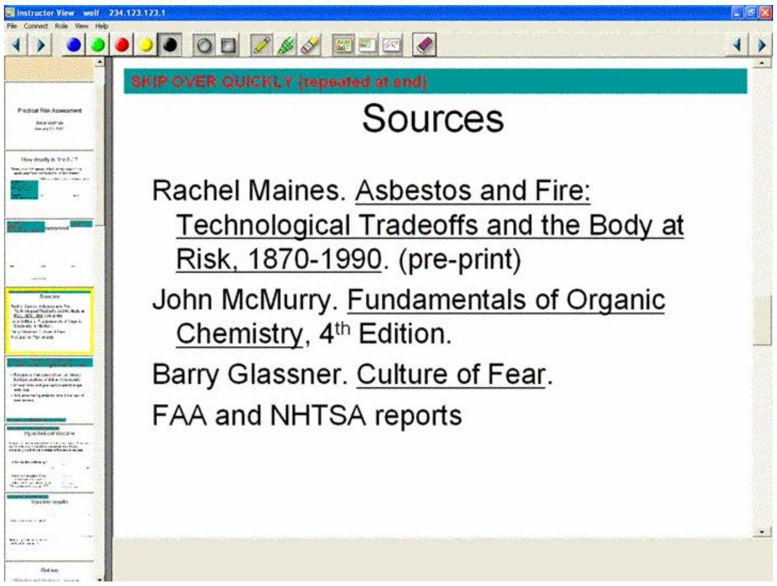
Rachel Maines. Asbestos and Fire: Technological Tradeoffs and the Body at Risk, 1870-1990. (pre-print)

John McMurry. Fundamentals of Organic Chemistry, 4th Edition.

Barry Glassner. Culture of Fear.

FAA and NHTSA reports

Figure A.10: Slide 4. Slides: PE/IE. The sources used to plan the class. The student view looks identical to the public view.



Instructor View - wolf - 234.123.123.1
File Connect Role View Help

SKIP OVER QUICKLY (repeated at end)

Sources

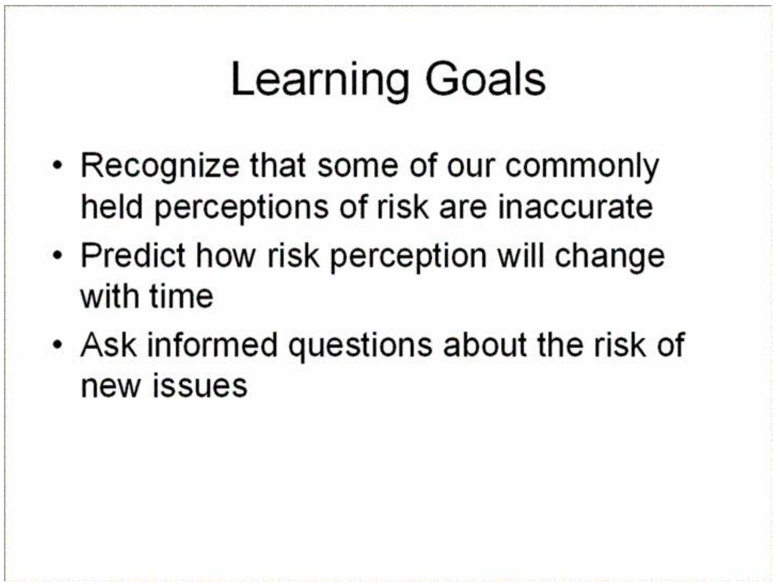
Rachel Maines. Asbestos and Fire: Technological Tradeoffs and the Body at Risk, 1870-1990. (pre-print)

John McMurry. Fundamentals of Organic Chemistry, 4th Edition.

Barry Glassner. Culture of Fear.

FAA and NHTSA reports

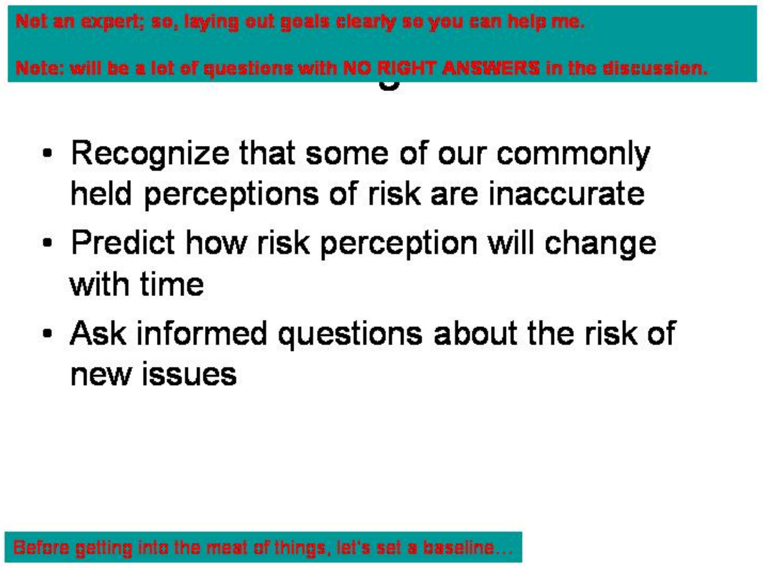
Figure A.11: Slide 4, IE. The instructor wanted to make clear to the students that authorities (and not he!) had provided the information used during the class. He also wanted to wait on longer discussion of sources until the end of the talk.



Learning Goals

- Recognize that some of our commonly held perceptions of risk are inaccurate
- Predict how risk perception will change with time
- Ask informed questions about the risk of new issues

Figure A.12: Slide 5. Slides: PE/ID. The learning goals for the class. The student view is identical to the public view.



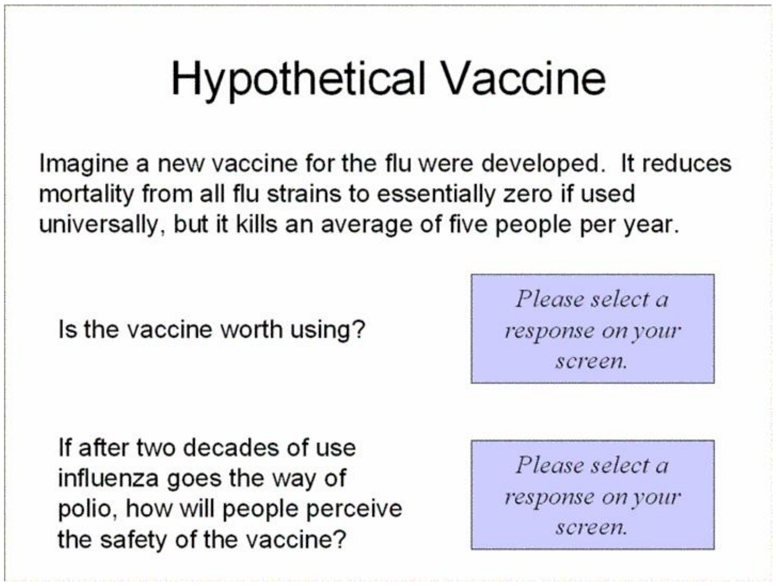
Not an expert; so, laying out goals clearly so you can help me.

Note: will be a lot of questions with NO RIGHT ANSWERS in the discussion.

- Recognize that some of our commonly held perceptions of risk are inaccurate
- Predict how risk perception will change with time
- Ask informed questions about the risk of new issues

Before getting into the meat of things, let's set a baseline…

Figure A.13: Slide 5, ID. The instructor included notes to remind him of discussion points and to prompt him for the transition to the next slide (at the bottom of the slide).



Hypothetical Vaccine

Imagine a new vaccine for the flu were developed. It reduces mortality from all flu strains to essentially zero if used universally, but it kills an average of five people per year.

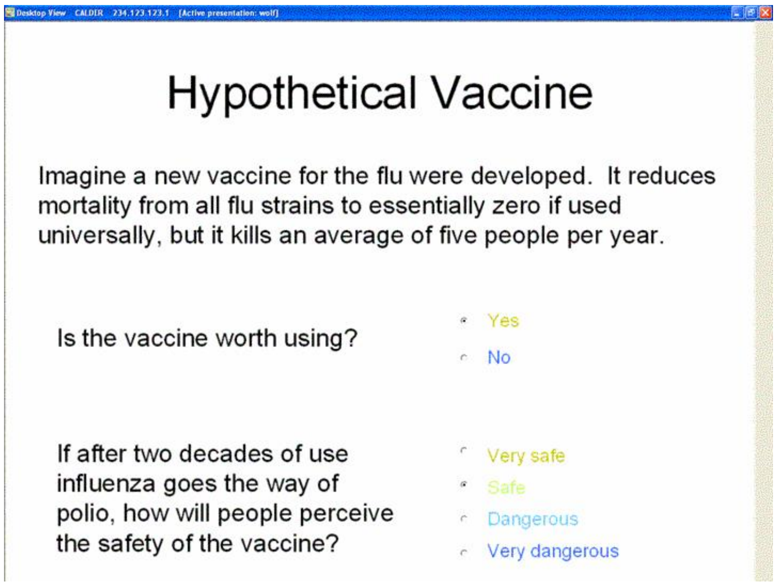
Is the vaccine worth using?

Please select a response on your screen.

If after two decades of use influenza goes the way of polio, how will people perceive the safety of the vaccine?

Please select a response on your screen.

Figure A.14: Slide 6. Slides: PE/SE/IE/ID. The public prompt for the class's second exercise, with two multiple choice questions about a flu vaccine. Students perform this exercise at the start of class and again at the end (see Slide 15, Figures A.12–A.13) to explore changes in beliefs about risk assessment during class. The shaded boxes appear only on the public display.



Desktop View CALDIR 234.123.123.1 [Active presentation: wolf]

Hypothetical Vaccine

Imagine a new vaccine for the flu were developed. It reduces mortality from all flu strains to essentially zero if used universally, but it kills an average of five people per year.

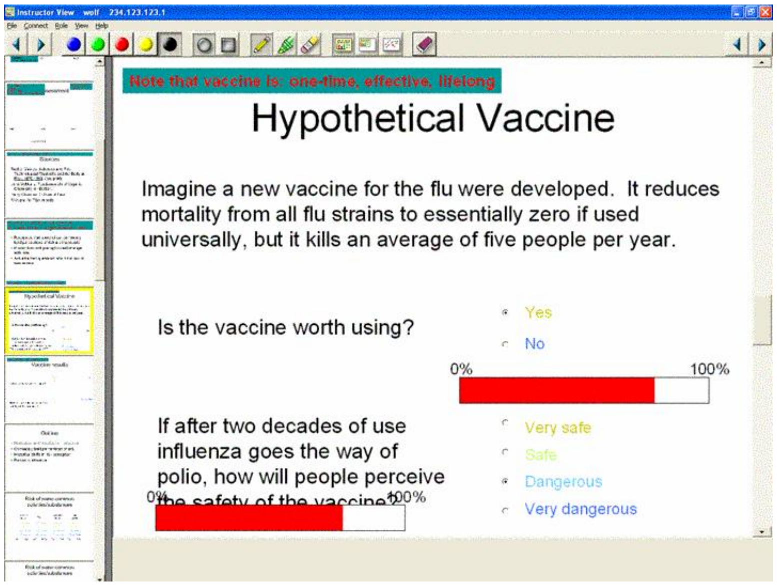
Is the vaccine worth using?

- [✓ Yes]
- No

If after two decades of use influenza goes the way of polio, how will people perceive the safety of the vaccine?

- Very safe
- [✓ Safe]
- Dangerous
- Very dangerous

Figure A.15: Slide 6, SE. Students respond to two color-coded multiple choice questions. This student responded "Yes" and "Safe".


Note that vaccine is: one-time, effective, lifelong

Hypothetical Vaccine

Imagine a new vaccine for the flu were developed. It reduces mortality from all flu strains to essentially zero if used universally, but it kills an average of five people per year.

Is the vaccine worth using?

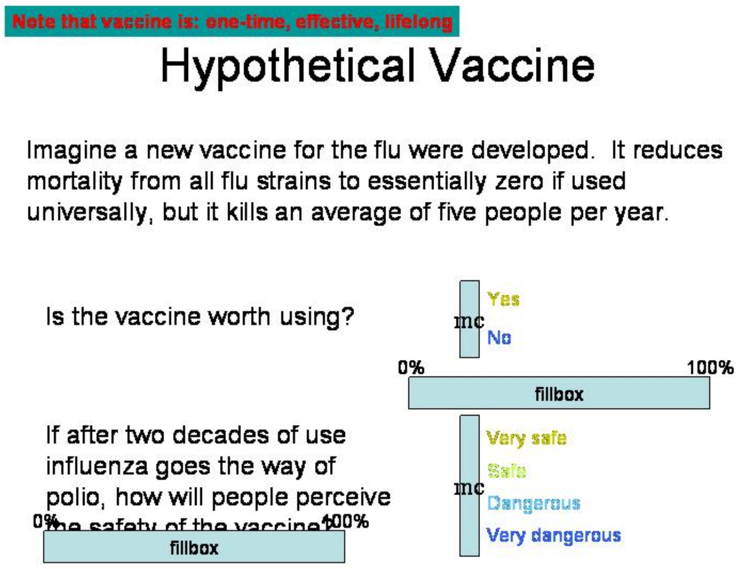
[✓ Yes] No

If after two decades of use influenza goes the way of polio, how will people perceive the safety of the vaccine?

Very safe Safe [✓ Dangerous] Very dangerous

Instructor View wolf 234.123.123.1
File Connect Role View Help
0%
100%
0%
100%

Figure A.16: Slide 6, IE. The instructor's view shows progress bars for both questions (as in Figure A.5). An instructor note also reminds of a clarification for the exercise.


Note that vaccine is: one-time, effective, lifelong

Hypothetical Vaccine

Imagine a new vaccine for the flu were developed. It reduces mortality from all flu strains to essentially zero if used universally, but it kills an average of five people per year.

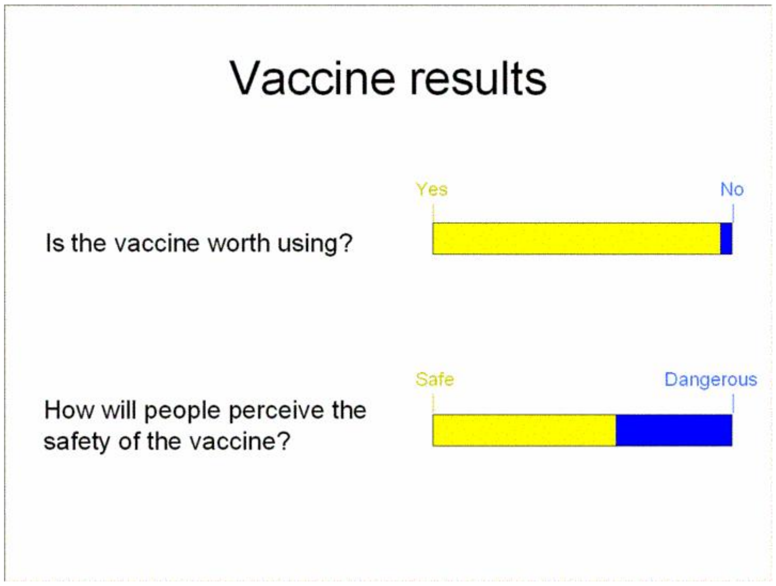
Is the vaccine worth using?

Yes No

If after two decades of use influenza goes the way of polio, how will people perceive the safety of the vaccine?

Very safe Safe Dangerous Very dangerous

Figure A.17: Slide 6, ID. This slide's design shows interwoven static elements — PowerPoint shapes like the textboxes with colored response text — and interactive slide elements — widgets like multiple choice questions and fillboxes. The fillboxes were easily constructed by copy-and-paste (with minor edits) from the fillbox of Figure A.6.

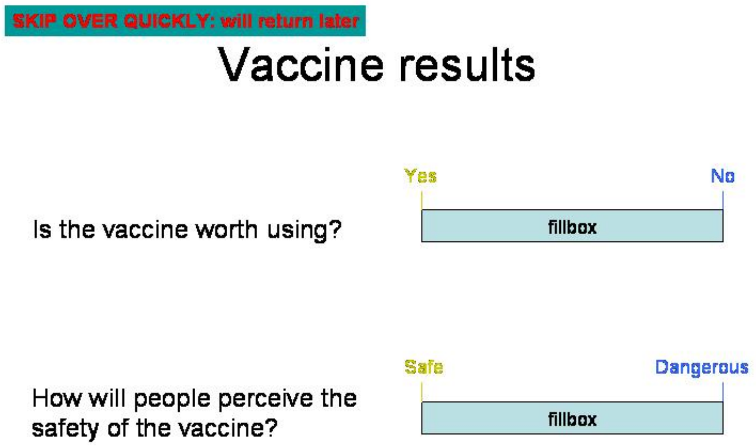


Vaccine results

Is the vaccine worth using?

How will people perceive the safety of the vaccine?

Figure A.18: Slide 7. Slides: PE/ID. Results from the exercise of Slide 6 (Figures A.14–A.17). Students overwhelmingly think the vaccine is worth using but are more conflicted about its safety. (More yellow indicates a heavier lean toward the yellow response. The Safe/Dangerous scale displays the mean of responses to the second question of Figure A.15, assuming a four-point scale.) The student view is identical to the public view.



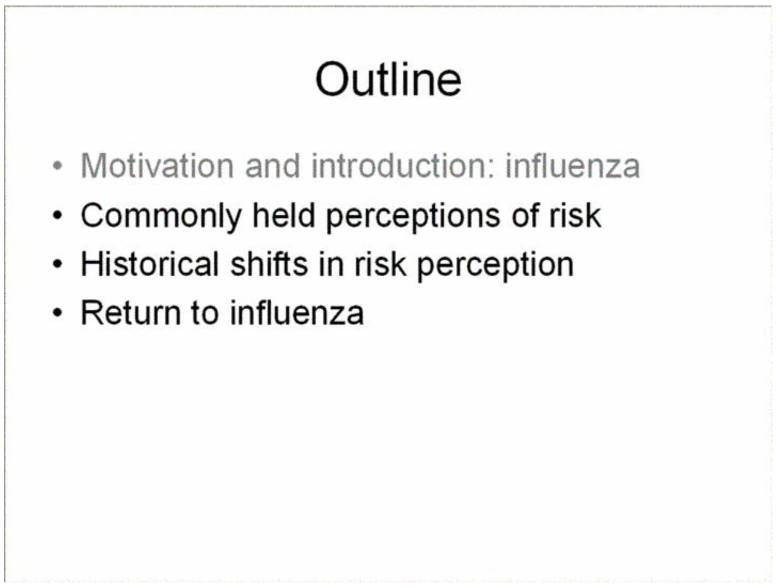
SKIP OVER QUICKLY: will return later

Vaccine results

Is the vaccine worth using?

How will people perceive the safety of the vaccine?

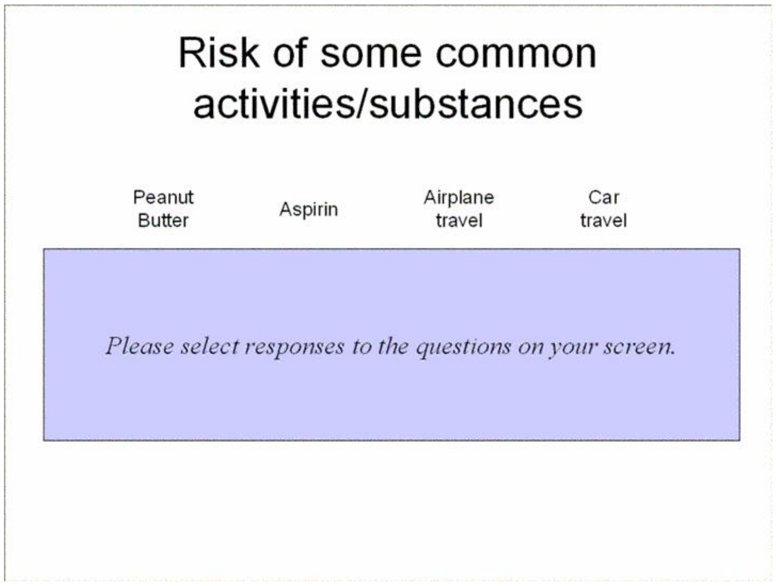
Figure A.19: Slide 7, ID. The instructor briefly discusses results from this slide but then moves on, intending to discuss the results in more detail when the exercise is repeated at the end of class (Slide 16, Figures A.38–A.39). Note again the combination of static elements (colored lines and text) and interactive widgets (fillboxes).



Outline

- Motivation and introduction: influenza
- Commonly held perceptions of risk
- Historical shifts in risk perception
- Return to influenza

Figure A.20: Slide 8. Slides: PE. The outline slide shows the three remaining sections of the class.

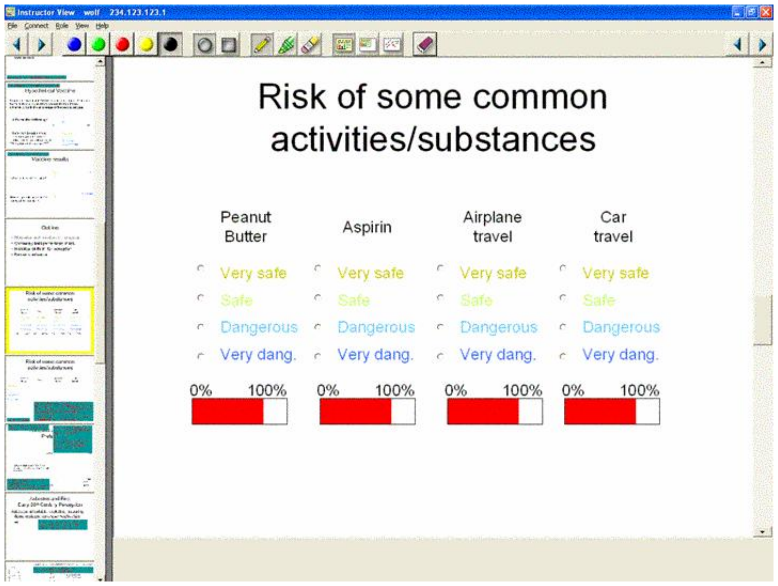


Risk of some common activities/substances

Peanut Butter | Aspirin | Airplane travel | Car travel

Please select responses to the questions on your screen.

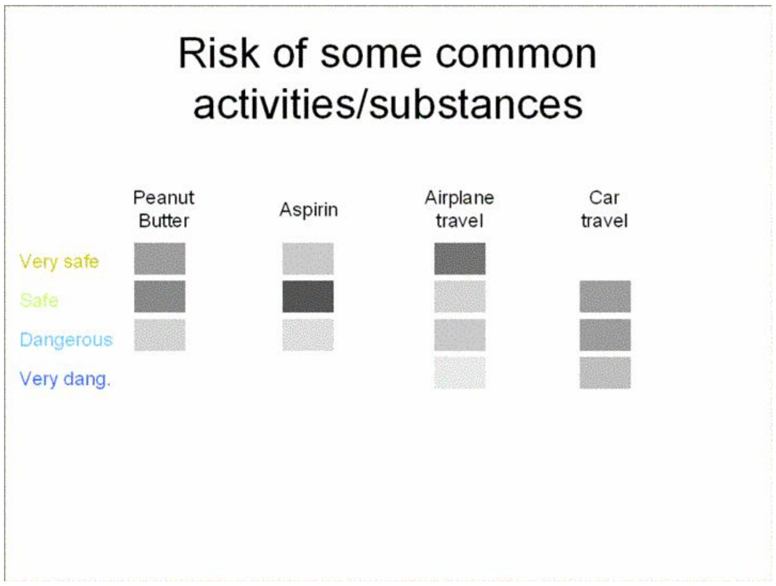
Figure A.21: Slide 9. Slides: PE/IE. The third exercise of the class introduces the four topics that will be used to discuss commonly held perceptions of risk. The instructor expected that some students would be unaware of significant dangers posed to certain populations by peanuts (*i.e.*, those with allergies) and aspirin (*i.e.*, some children with influenza) as well as broader potential dangers (*i.e.*, potential for aflatoxin in peanut butter and for injury to the stomach lining from aspirin). Also, many people misperceive the relative danger of airplane and car travel: car travel is much more dangerous by almost any measure.


Instructor View wolf 234.123.123.1
File Connect Role View Help

Risk of some common activities/substances

Peanut Butter	Aspirin	Airplane travel	Car travel
Very safe	Very safe	Very safe	Very safe
Safe	Safe	Safe	Safe
Dangerous	Dangerous	Dangerous	Dangerous
Very dang.	Very dang.	Very dang.	Very dang.
0% 100%	0% 100%	0% 100%	0% 100%

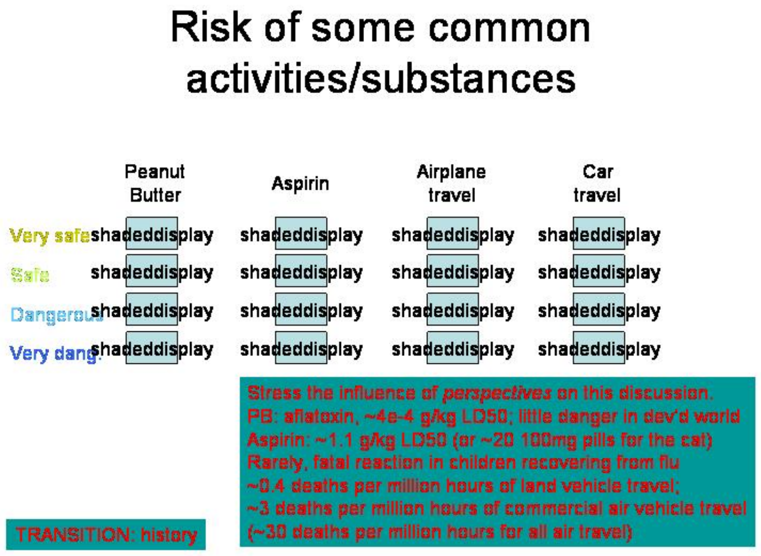
Figure A.22: Slide 9, IE. The instructor's view shows the response choices for the questions, the same as for the second question in Figure A.15. The instructor has not responded to the questions although the progress bars show that most students have. The student view of this slide is identical to the instructor's except that it lacks the progress bars.



Risk of some common activities/substances

	Peanut Butter	Aspirin	Airplane travel	Car travel
Very safe	▨	▨	▨	
Safe	▨	▨	▨	▨
Dangerous	▨	▨	▨	▨
Very dang.			▨	▨

Figure A.23: Slide 10. Slides: PE/ID. Results from the exercise of Slide 9 (Figures A.21–A.22). Each box's darkness on the display correlates to the proportion of students that chose its row's response for its column's prompt. The student view of this slide is identical to the public view. The data in this view shows a broad spread on the safety of three of the topics but general agreement on the relative safety of aspirin. Quite a few students also saw airplane travel as very safe. This slide led to extended discussion in class (more than six minutes long) about the dangers of each topic, why the class's perceptions of risk differed across each one, and what factors need to be considered besides objective measures of danger to assess risk (like the individual's feeling of control or lack thereof in cars *versus* airplanes). The instructor realized during discussion that he should have displayed the *differences* in responses to the airplane and car questions to emphasize perception of their relative risks. This change would be straightforward in SIP's design environment but could not be made dynamically.



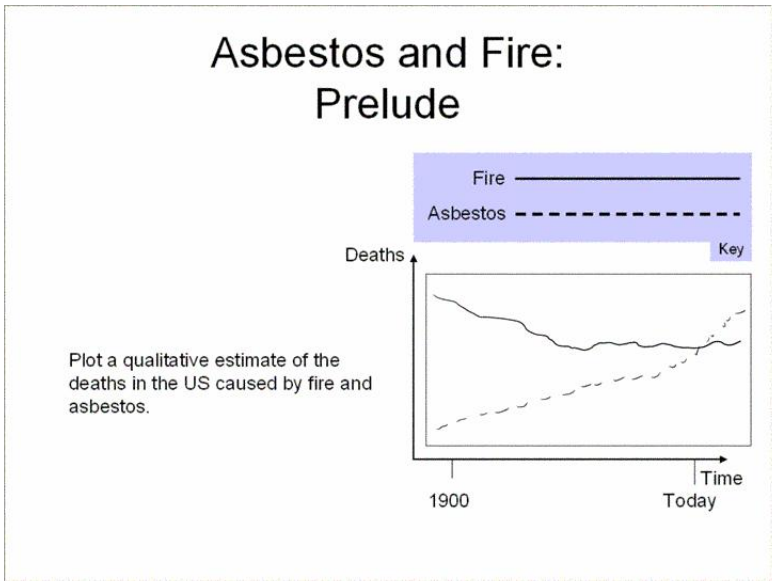
Risk of some common activities/substances

	Peanut Butter	Aspirin	Airplane travel	Car travel
Very safe	shadeddisplay	shadeddisplay	shadeddisplay	shadeddisplay
Safe	shadeddisplay	shadeddisplay	shadeddisplay	shadeddisplay
Dangerous	shadeddisplay	shadeddisplay	shadeddisplay	shadeddisplay
Very dang.	shadeddisplay	shadeddisplay	shadeddisplay	shadeddisplay

Stress the influence of *perspectives* on this discussion.
PB: aflatoxin, ~4e-4 g/kg LD50; little danger in dev'd world
Aspirin: ~1.1 g/kg LD50 (or ~20 100mg pills for the cat)
Rarely, fatal reaction in children recovering from flu
~0.4 deaths per million hours of land vehicle travel;
~3 deaths per million hours of commercial air vehicle travel
(~30 deaths per million hours for all air travel)

TRANSITION: history

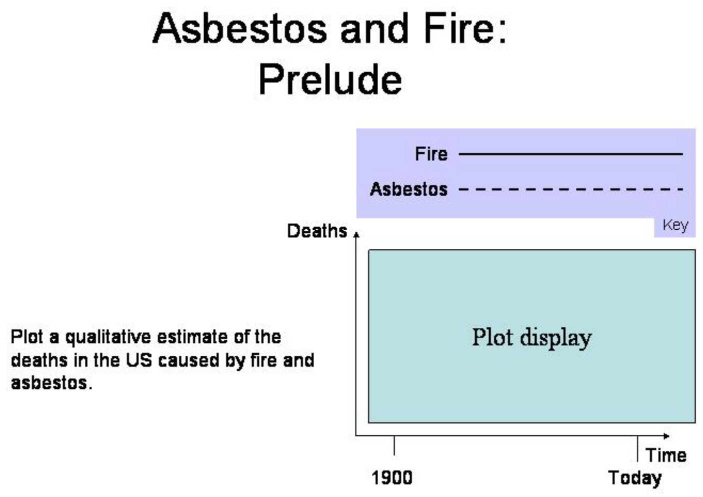
Figure A.24: Slide 10, ID. The display from Figure A.23 is built from an array of simple, identical widgets. Customized properties, copy-and-paste, and multiple selection-editing makes creating this display easy, as described in Section 5.4.1. As usual, the instructor notes on the slide remind of discussion points, facts, and an intended transition for the next slide.



Asbestos and Fire: Prelude

Plot a qualitative estimate of the deaths in the US caused by fire and asbestos.

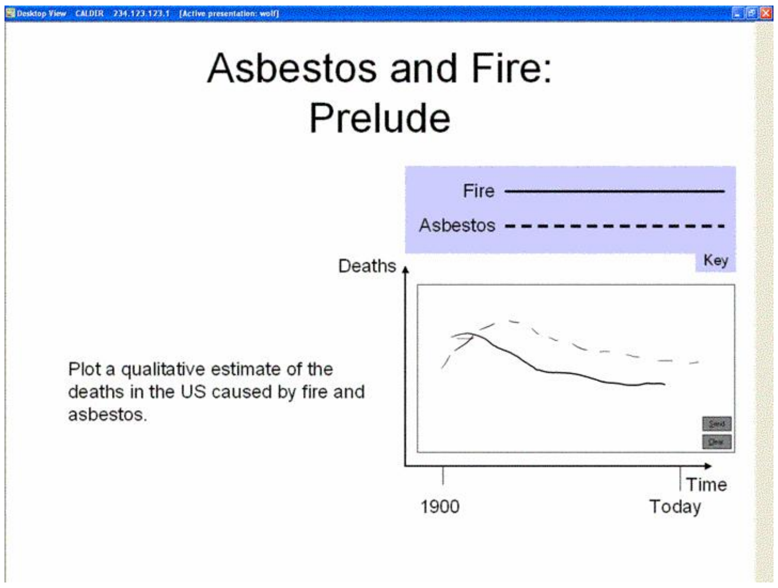
Figure A.25: Slide 11. Slides: PE/PD/SE/SD/IE/ID. This slide includes both the input and display for the fourth exercise. The exercise introduces the next main discussion topic: historical perception of risk. The exercise is meant to help students clarify and construct their own opinions about how risk perception of public health technologies changes over time *before* a lecture-style discussion of the topic. The instructor gives a few facts about the history of asbestos and fire, students draw qualitative plots of the deaths from asbestos and fire over time, and then the class discusses several of the plots and what assumptions informed their shapes. The system restart discussed in Section 5.5.2 (page 161) happened after the public view (and students' views) failed to follow the transition from this slide to the next.



Asbestos and Fire: Prelude

Plot a qualitative estimate of the deaths in the US caused by fire and asbestos.

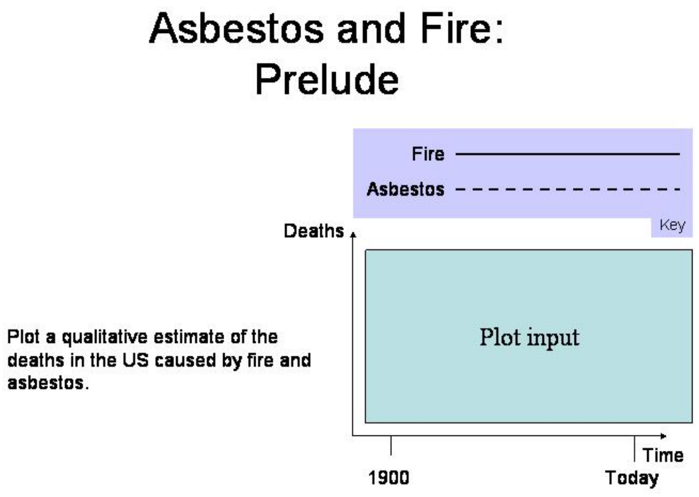
Figure A.26: Slide 11, PD. Note that the public view has an ink display (here labeled “Plot display”) where the other views have ink inputs (see Figures A.28 and A.30). This display shows whichever ink the instructor selects on his display (see Figure A.29).


Desktop View CALDIR 234.123.123.1 [Active presentation: wolf]
Fire
Asbestos
Key
Deaths
Time
1900
Today
Send
Clear

Asbestos and Fire:
Prelude

Plot a qualitative estimate of the deaths in the US caused by fire and asbestos.

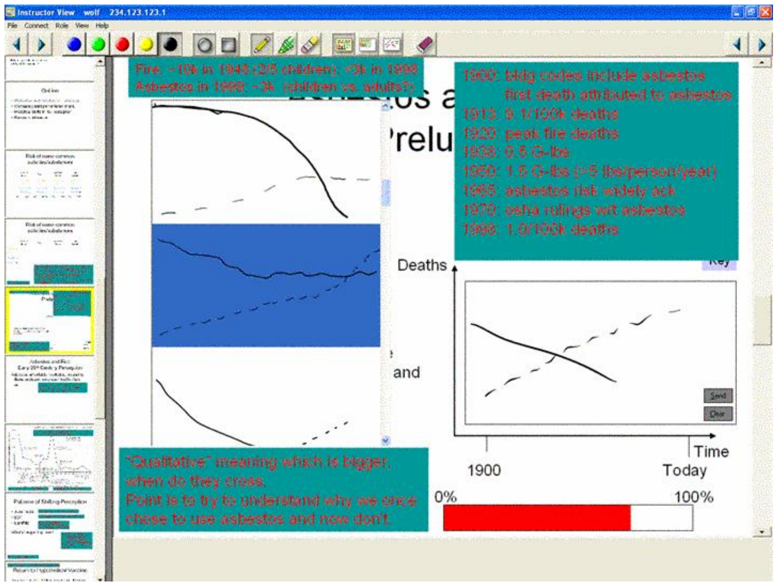
Figure A.27: Slide 11, SE. The student has drawn a plot (not the one selected by the instructor in Figure A.29 and displayed in Figure A.25).



Asbestos and Fire:
Prelude

Plot a qualitative estimate of the deaths in the US caused by fire and asbestos.

Figure A.28: Slide 11, SD. The student's design view has an ink input (labeled "Plot input") rather than a display like the public view in Figure A.26. This widget is the same as the one in the instructor's view in Figure A.30.

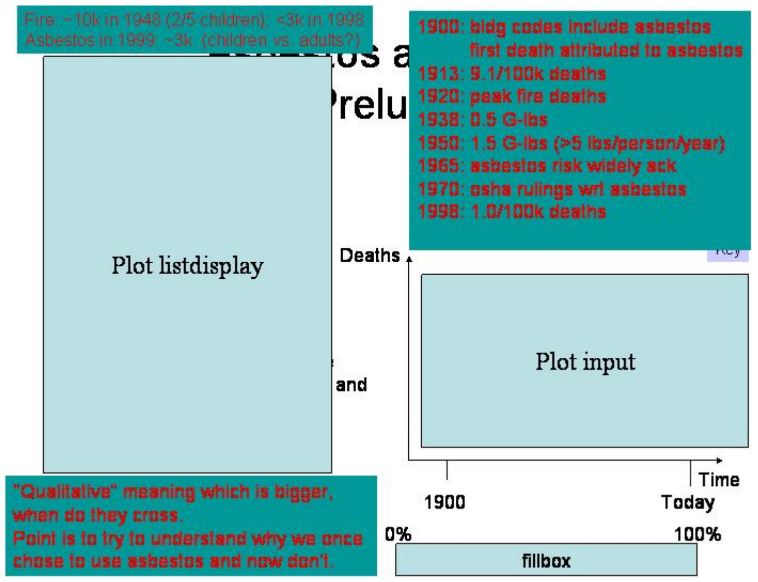

Fire: ~10k in 1943 (2/5 children); <3k in 1998
Asbestos in 1999: ~3k (children vs. adults?)

1900: bldg codes include asbestos
first death attributed to asbestos
1913: 9.1/100k deaths
1920: peak fire deaths
1938: 0.5 G-lbs
1950: 1.5 G-lbs (>5 lbs/person/year)
1965: asbestos risk widely ack
1970: osha rulings wrt asbestos
1998: 1.0/100k deaths

"Qualitative" meaning which is bigger,
when do they cross.
Point is to try to understand why we once
chose to use asbestos and now don't.

Instructor View wolf 234.123.123.1
File Connect Role View Help
Deaths
Key
Time
1900
Today
0%
100%
Send
Clear

Figure A.29: Slide 11, IE. The instructors view includes instructor notes, a list of submitted ink from which the instructor chooses the ink to display on the public view (Figure A.25), an ink input for the instructor to give his own answer, and a progress bar. The lower left instructor note reminds him to clarify what's meant by "qualitative": "which is bigger" and "when do they cross". Notes at the top give facts (including a timeline) that students might be interested in during the exercise. He offered a few of these immediately and others in response to student questions. The instructor also ended up drawing a sample diagram (like the one shown) to illustrate what he meant for students to do in input box.

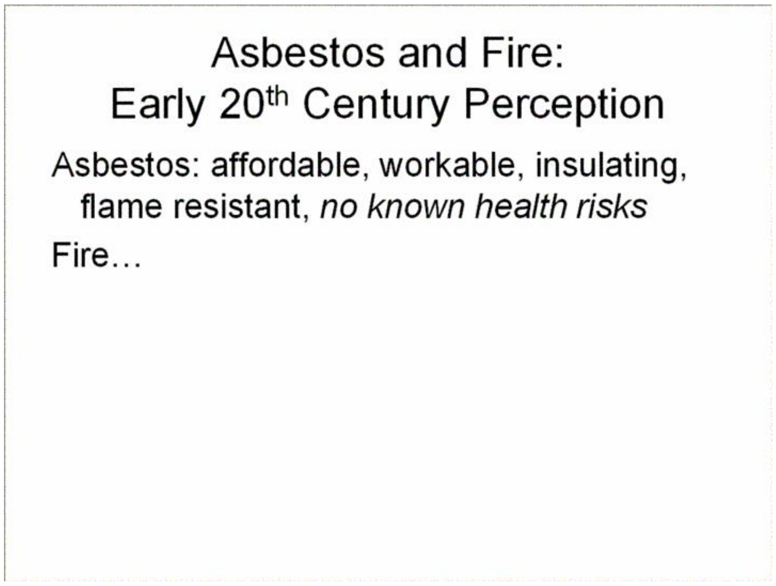


Fire: ~10k in 1948 (2/5 children); <3k in 1998
Asbestos in 1999: ~3k (children vs. adults?)

1900: bldg codes include asbestos
first death attributed to asbestos
1913: 9.1/100k deaths
1920: peak fire deaths
1938: 0.5 G-lbs
1950: 1.5 G-lbs (>5 lbs/person/year)
1965: asbestos risk widely ack
1970: osha rulings wrt asbestos
1998: 1.0/100k deaths

"Qualitative" meaning which is bigger, when do they cross.
Point is to try to understand why we once chose to use asbestos and now don't.

Figure A.30: Slide 11, ID. The instructor's design view includes three widgets. The progress bar is configured as in previous slides. The list display has a property that takes a list of ink inputs. That property is set to pull in completed submissions from all participants (including the instructor) on the ink input. Finally, the ink display in Figure A.26 simply shows whichever ink object is selected (an output property of the list display).

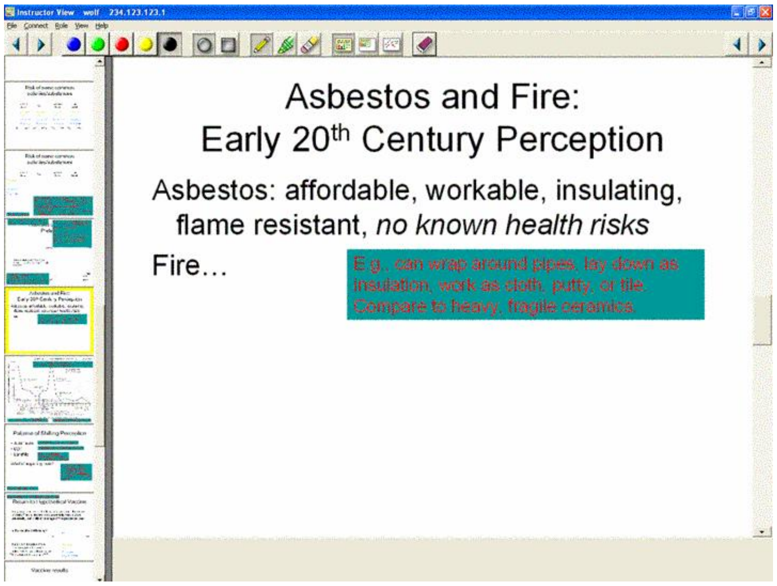


Asbestos and Fire: Early 20th Century Perception

Asbestos: affordable, workable, insulating, flame resistant, *no known health risks*

Fire…

Figure A.31: Slide 12. Slides: PE/IE. This slide introduces some facts about why people might have chosen to use asbestos.


Instructor View - wolf - 234.123.123.1
File Connect Role View Help

Asbestos and Fire: Early 20th Century Perception

Asbestos: affordable, workable, insulating, flame resistant, *no known health risks*

Fire…

E.g., can wrap around pipes, lay down as insulation, work as cloth, putty, or tile. Compare to heavy, fragile ceramics.

Figure A.32: Slide 12, IE. An instructor note lists specific aspects of asbestos that made it attractive.

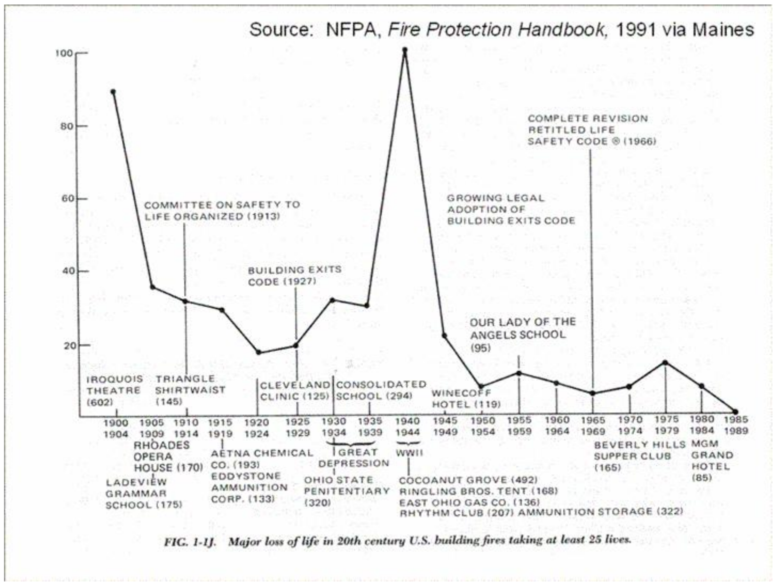
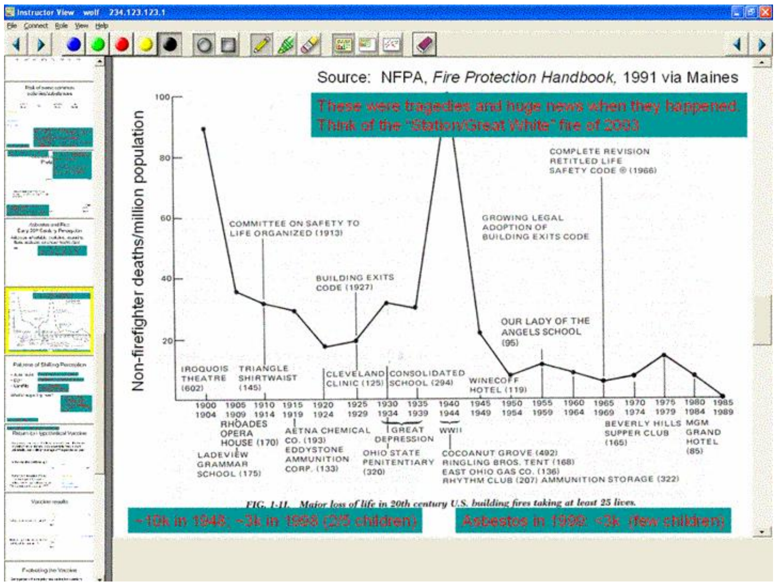


Figure A.33: Slide 13. Slides: PE/IE. Plot of historical deaths from fire. This slide partly explained why fire seemed more dangerous at the start of the 20^{th} century than at its end. The class spent four minutes discussing this slide, much as they discussed the results of the various exercises: assimilating dense information on the slide into the broader class themes.

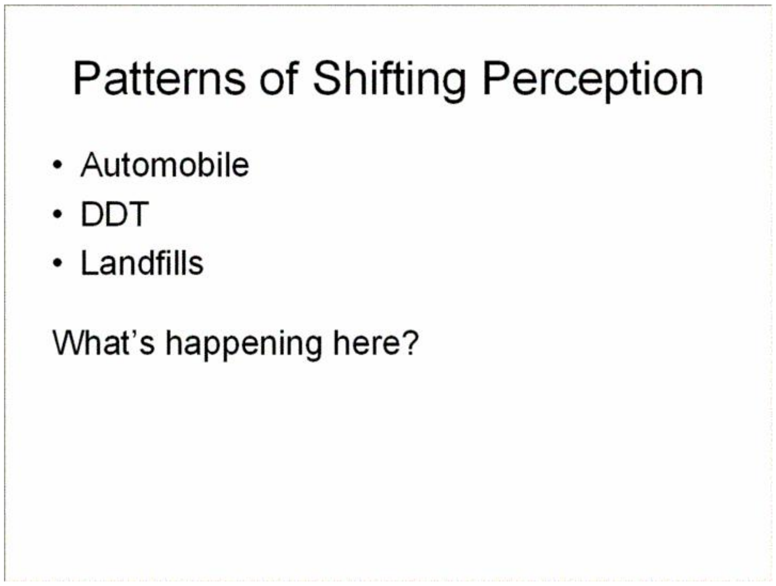

These were tragedies and huge news when they happened. Think of the "Station/Great White" fire of 2003

~10k in 1948, ~3k in 1998 (2/5 children)

Asbestos in 1999: <3k (few children)

Instructor View wolf 234.123.123.1
File Connect Role View Help
Source: NFPA, *Fire Protection Handbook*, 1991 via Maines
Non-firefighter deaths/million population
100
80
60
40
20
COMMITTEE ON SAFETY TO LIFE ORGANIZED (1913)
BUILDING EXITS CODE (1927)
GROWING LEGAL ADOPTION OF BUILDING EXITS CODE
COMPLETE REVISION RETITLED LIFE SAFETY CODE ® (1966)
OUR LADY OF THE ANGELS SCHOOL (95)
IROQUOIS THEATRE (602)
TRIANGLE SHIRTWAIST (145)
CLEVELAND CLINIC (125)
CONSOLIDATED SCHOOL (294)
WINECOFF HOTEL (119)
RHOADES OPERA HOUSE (170)
LADEVIEW GRAMMAR SCHOOL (175)
AETNA CHEMICAL CO. (193)
EDDYSTONE AMMUNITION CORP. (133)
GREAT DEPRESSION
OHIO STATE PENITENTIARY (320)
WWII
COCOANUT GROVE (492)
RINGLING BROS. TENT (168)
EAST OHIO GAS CO. (136)
RHYTHM CLUB (207) AMMUNITION STORAGE (322)
BEVERLY HILLS SUPPER CLUB (165)
MGM GRAND HOTEL (85)
FIG. 1-1J. Major loss of life in 20th century U.S. building fires taking at least 25 lives.

Figure A.34: Slide 13, IE. The instructor's view includes notes with more details about deaths from fire and asbestos.

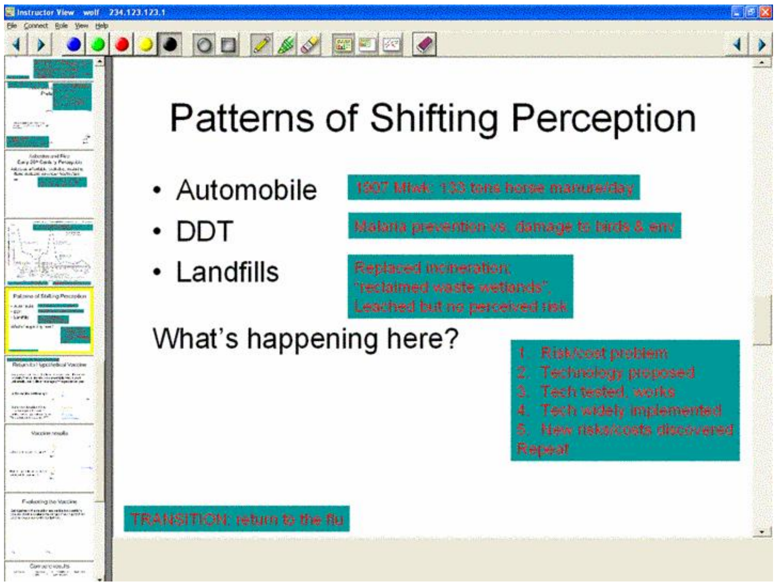


Patterns of Shifting Perception

- Automobile
- DDT
- Landfills

What's happening here?

Figure A.35: Slide 14. Slides: PE/IE. A slide relating the discussion of fire and asbestos to other historical shifts in risk perception of technologies that impacted public health.



Instructor View wolf 234.123.123.1
File Connect Role View Help

Patterns of Shifting Perception

- Automobile — 1907 Mlwk: 133 tons horse manure/day
- DDT — Malaria prevention vs. damage to birds & env
- Landfills — Replaced incineration; "reclaimed waste wetlands", Leached but no perceived risk

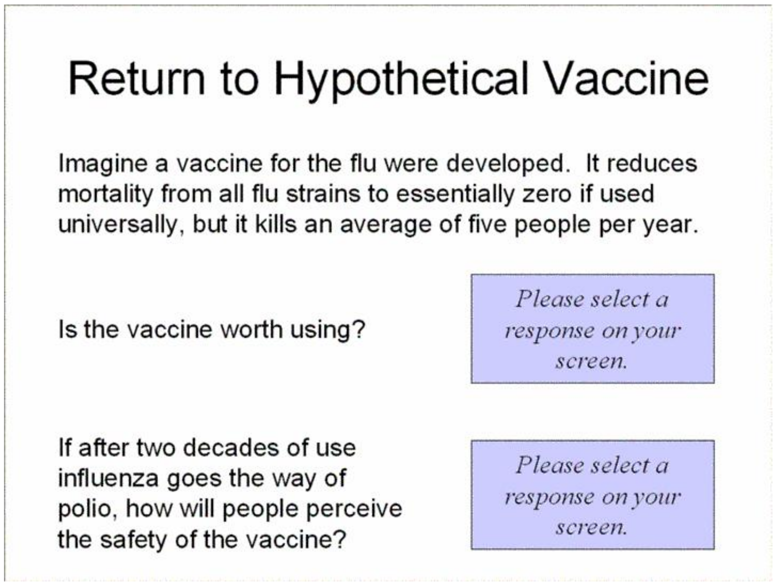
What's happening here?

1. Risk/cost problem
2. Technology proposed
3. Tech tested, works
4. Tech widely implemented
5. New risks/costs discovered

Repeat

TRANSITION: return to the flu

Figure A.36: Slide 14, IE. The instructor's view includes notes with discussion points, a transition to the next slide, and some facts about changes in historical perception of technology (*e.g.*, in 1907 Milwaukee cleaned 133 tons of horse manure per day from the streets, which *does* make the automobile sound like a "clean" technology).



Return to Hypothetical Vaccine

Imagine a vaccine for the flu were developed. It reduces mortality from all flu strains to essentially zero if used universally, but it kills an average of five people per year.

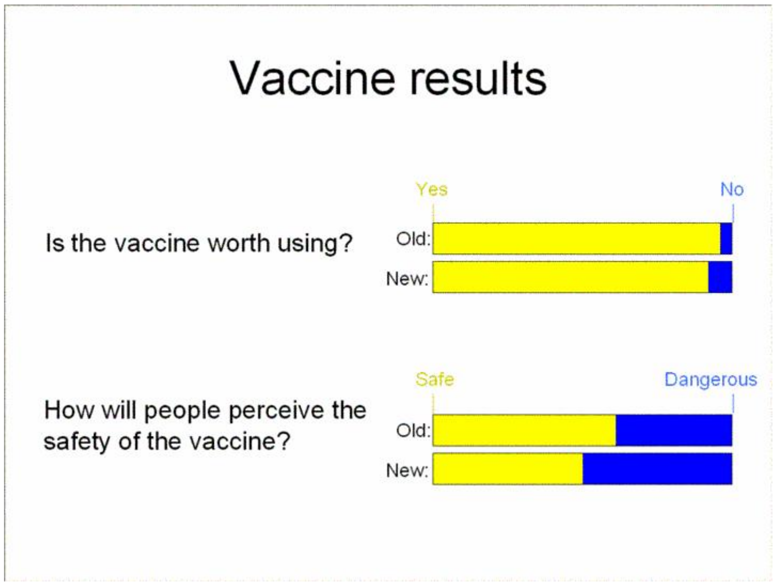
Is the vaccine worth using?

Please select a response on your screen.

If after two decades of use influenza goes the way of polio, how will people perceive the safety of the vaccine?

Please select a response on your screen.

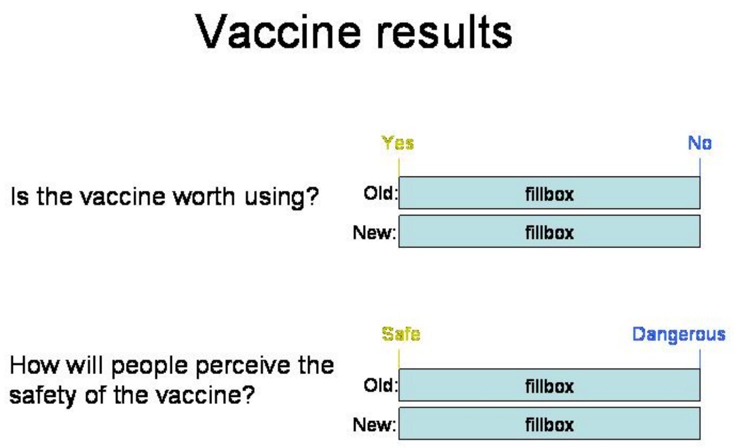
Figure A.37: Slide 15. Slides: PE. The return to the hypothetical vaccine exercise promised back on Slide 6, Figures A.14–A.17. Only the public view of the slide's execution is shown because the slide is identical to Slide 6 besides a tweak to the title and changes to the names of the widgets used.


Vaccine results

Is the vaccine worth using?

How will people perceive the safety of the vaccine?

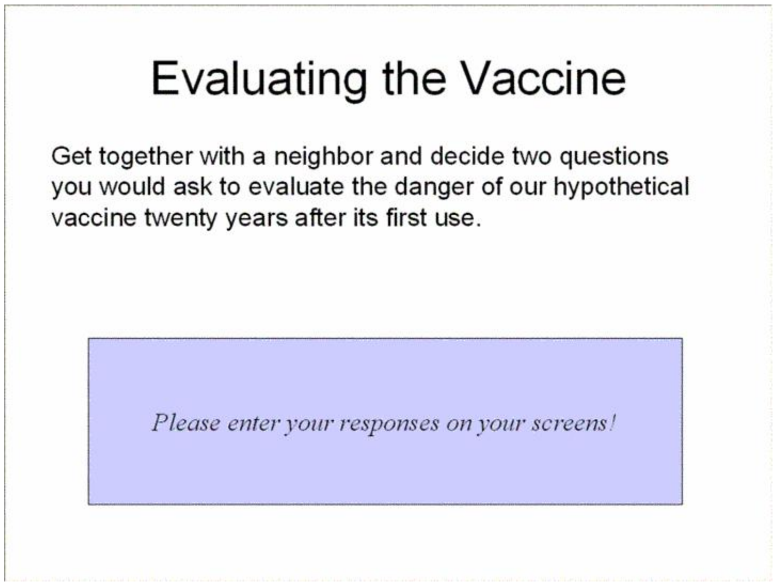
Figure A.38: Slide 16. Slides: PE/PD. Comparative results of the exercises from Slide 15 (Figure A.37; the “New” results on this slide) and Slide 6 (Figures A.14–A.17; the “Old” results on this slide). This slide sparked five minutes of discussion about why students’ responses changed. The student and instructor views are identical to the public view.


Vaccine results

Is the vaccine worth using?

How will people perceive the safety of the vaccine?

Figure A.39: Slide 16, PD. To design this slide, the instructor copied Slide 7 (Figure A.19) and then re-copied each fillbox, changing only the new fillboxes’ data sources.

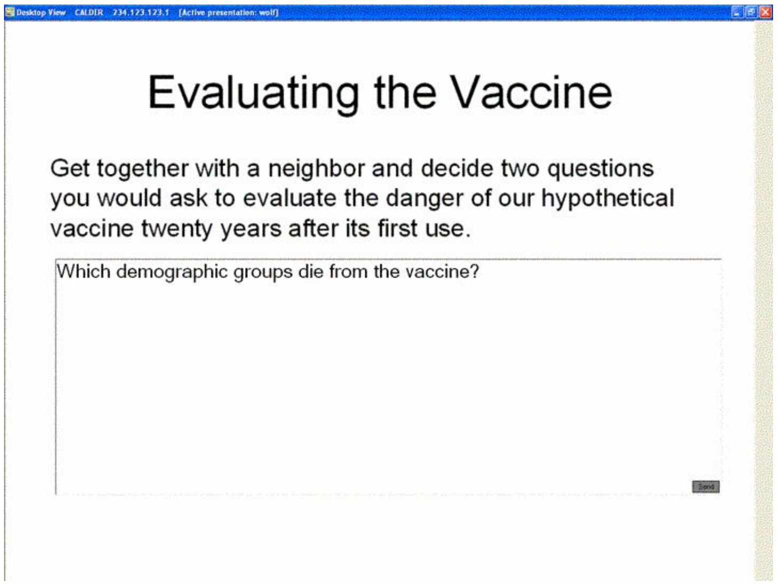


Evaluating the Vaccine

Get together with a neighbor and decide two questions you would ask to evaluate the danger of our hypothetical vaccine twenty years after its first use.

Please enter your responses on your screens!

Figure A.40: Slide 17. Slides: PE/SE. The prompt for the first phase of the final exercise. The exercise (detailed in Section 5.2.3) stretches across Slides 17–19. The exercise's purpose was to prepare students to apply their new knowledge of risk assessment by thinking of questions they could ask about public health technologies.

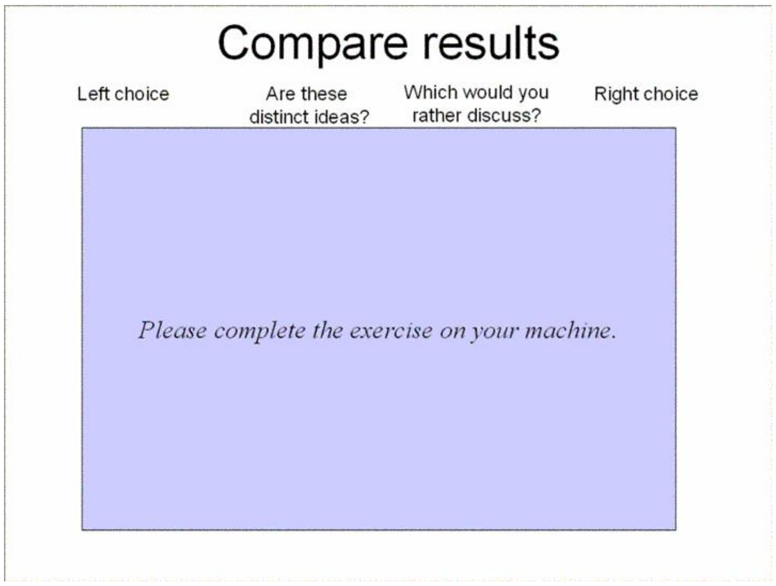

Desktop View CALDIR 234.123.123.1 [Active presentation: wolf]
Send

Evaluating the Vaccine

Get together with a neighbor and decide two questions you would ask to evaluate the danger of our hypothetical vaccine twenty years after its first use.

Which demographic groups die from the vaccine?

Figure A.41: Slide 17, SE. This student has responded with the question "Which demographic groups die from the vaccine?" The instructor's view looks like the student's view except that it includes a progress bar.



Compare results

Left choice

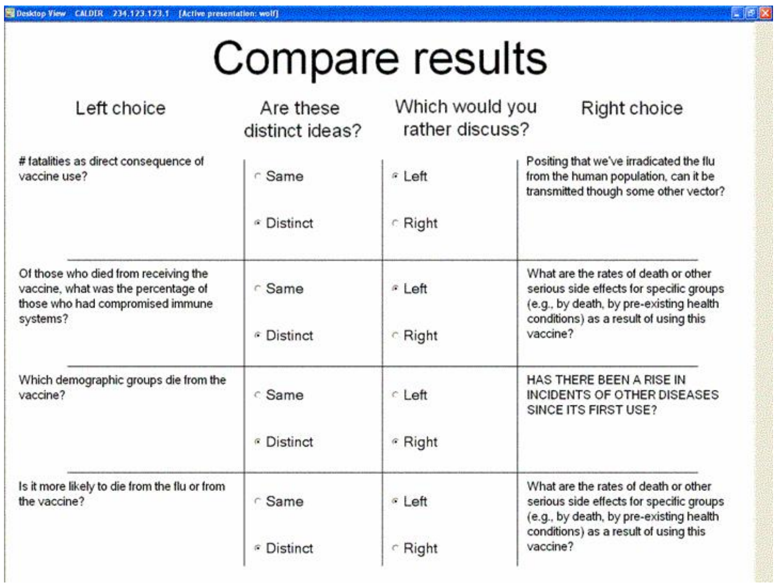
Are these distinct ideas?

Which would you rather discuss?

Right choice

Please complete the exercise on your machine.

Figure A.42: Slide 18. Slides: PE/SE/SD/ID. In the next phase of the exercise from Slide 17, students investigated each others' responses.

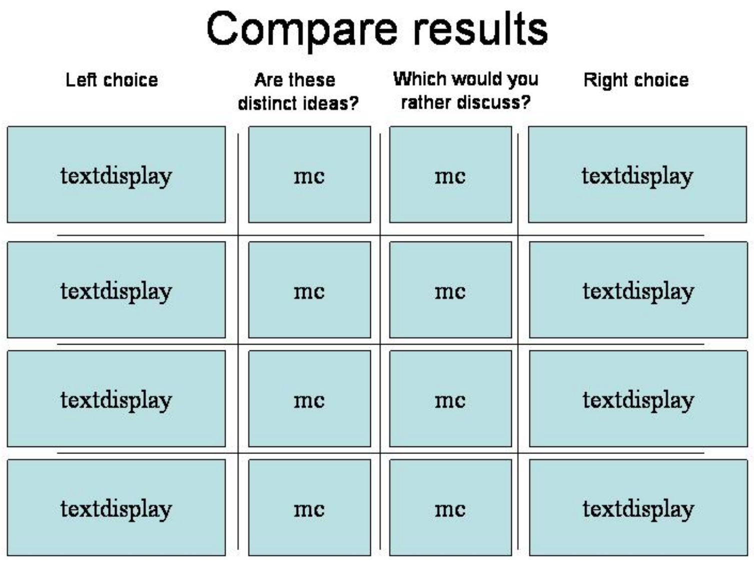


Desktop View CALDIR 234.123.123.1 [Active presentation: wolf]

Compare results

Left choice	Are these distinct ideas?	Which would you rather discuss?	Right choice
# fatalities as direct consequence of vaccine use?	Same [✓ Distinct]	[✓ Left] Right	Positing that we've irradicated the flu from the human population, can it be transmitted though some other vector?
Of those who died from receiving the vaccine, what was the percentage of those who had compromised immune systems?	Same [✓ Distinct]	[✓ Left] Right	What are the rates of death or other serious side effects for specific groups (e.g., by death, by pre-existing health conditions) as a result of using this vaccine?
Which demographic groups die from the vaccine?	Same [✓ Distinct]	Left [✓ Right]	HAS THERE BEEN A RISE IN INCIDENTS OF OTHER DISEASES SINCE ITS FIRST USE?
Is it more likely to die from the flu or from the vaccine?	Same [✓ Distinct]	[✓ Left] Right	What are the rates of death or other serious side effects for specific groups (e.g., by death, by pre-existing health conditions) as a result of using this vaccine?

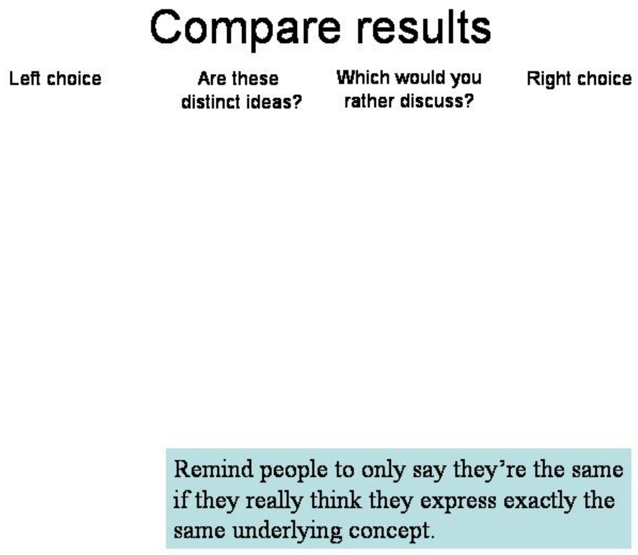
Figure A.43: Slide 18, SE. A student rates other students' responses. For example, the first row's left response is "# fatalities as direct consequence of vaccine use?", and the right is "Positing that we've irradicated the flu from the human population, can it be transmitted through some other vector?" The student felt the questions address distinct ideas and would rather discuss the left one.



Compare results

Left choice	Are these distinct ideas?	Which would you rather discuss?	Right choice
textdisplay	mc	mc	textdisplay
textdisplay	mc	mc	textdisplay
textdisplay	mc	mc	textdisplay
textdisplay	mc	mc	textdisplay

Figure A.44: Slide 18, SD. Like Slide 10 (Figure A.24), designing this slide involved configuring one row of widgets with custom properties and then copying-and-pasting the result, with minor edits, to the remaining rows.

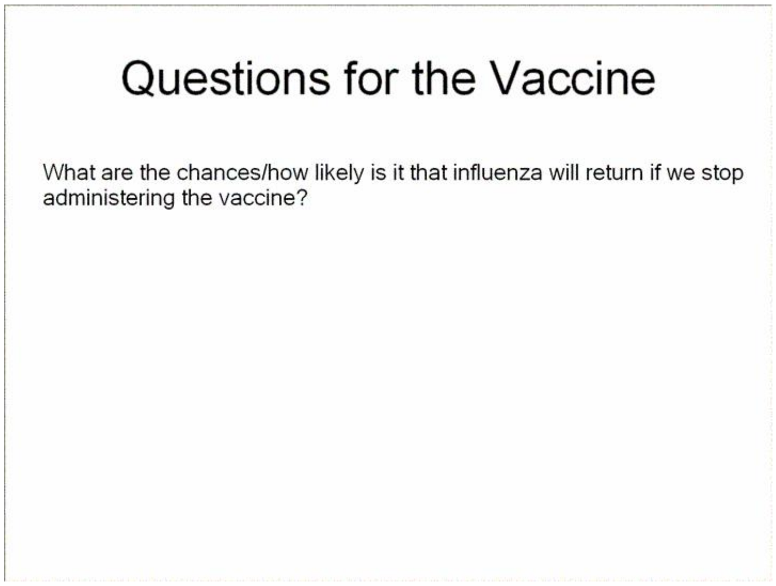


Compare results

Left choice | Are these distinct ideas? | Which would you rather discuss? | Right choice

Remind people to only say they're the same if they really think they express exactly the same underlying concept.

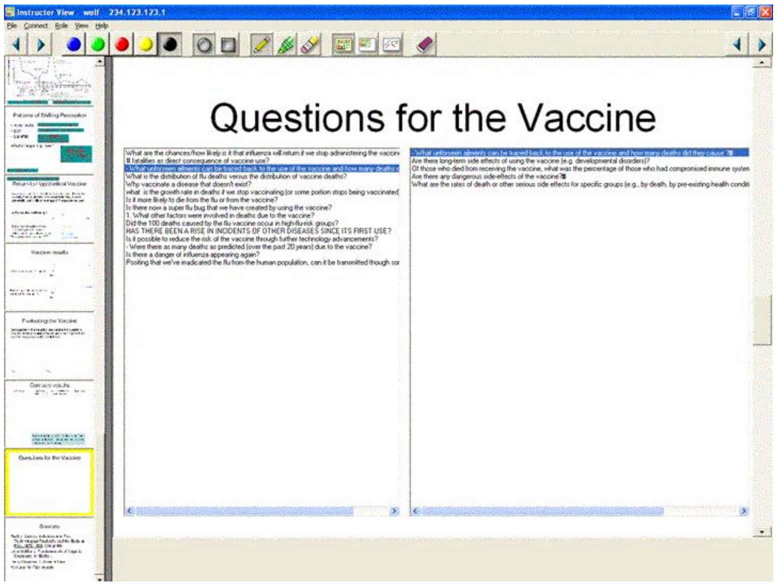
Figure A.45: Slide 18, ID. The instructor chose not to include himself on this exercise; so, only an instructor note appears on the screen. Note that the instructor placed a "dummy" widget offscreen, added a custom property to it, and collected students' responses to that dummy widget's new property. Those responses were then forwarded to the next slide's widgets.



Questions for the Vaccine

What are the chances/how likely is it that influenza will return if we stop administering the vaccine?

Figure A.46: Slide 19. Slides: PE/IE/ID. The results slide for the exercise starting on Slide 17. The public display shows one student response, the one the instructor has selected in Figure A.47. The student view is identical to the public view.

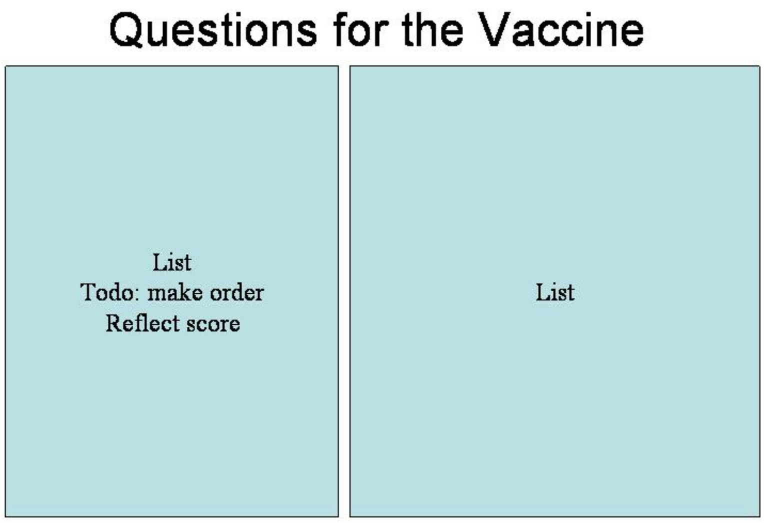

Instructor View wolf 234.123.123.1
File Connect Role View Help

Questions for the Vaccine

What are the chances/how likely is it that influenza will return if we stop administering the vaccin
‼ fatalities as direct consequence of vaccine use?
- What unforseen ailments can be traced back to the use of the vaccine and how many deaths c
What is the distribution of flu deaths versus the distribution of vaccine deaths?
Why vaccinate a disease that doesn't exist?
what is the growth rate in deaths if we stop vaccinating (or some portion stops being vaccinated(
Is it more likely to die from the flu or from the vaccine?
Is there now a super flu bug that we have created by using the vaccine?
1. What other factors were involved in deaths due to the vaccine?
Did the 100 deaths caused by the flu vaccine occur in high-flu-risk groups?
HAS THERE BEEN A RISE IN INCIDENTS OF OTHER DISEASES SINCE ITS FIRST USE?
Is it possible to reduce the risk of the vaccine through futher technology advancements?
- Were there as many deaths as predicted (over the past 20 years) due to the vaccine?
Is there a danger of influenza appearing again?
Positing that we've inadicated the flu from the human population, can it be transmitted though so

- What unforseen ailments can be traced back to the use of the vaccine and how many deaths did they cause ?
Are there long-term side effects of using the vaccine (e.g. developmental disorders)?
Of those who died from receiving the vaccine, what was the percentage of those who had compromised immune systen
Are there any dangerous side-effects of the vaccine?
What are the rates of death or other serious side effects for specific groups (e.g., by death, by pre-existing health condi

Figure A.47: Slide 19, IE. The instructor sees two lists of responses. Both lists are ordered according by "score", according to which responses students said they would rather discuss on Slide 18 (Figure A.43). The responses are also grouped into equivalence classes based on which responses students said covered the same topic. ("Distinct" labels are ignored.) The left-hand list, then, shows the top-ranked responses in each equivalence class. The right-hand list shows all of the responses in whichever equivalence class is selected on the left. As the instructor selects new rows on the left, the right-hand list displays new groups. As the instructor selects rows on the right, the public and student views display the selected student responses. Section 5.5.2 includes a list of all the results, grouped and ranked as they were in the class (page 168).

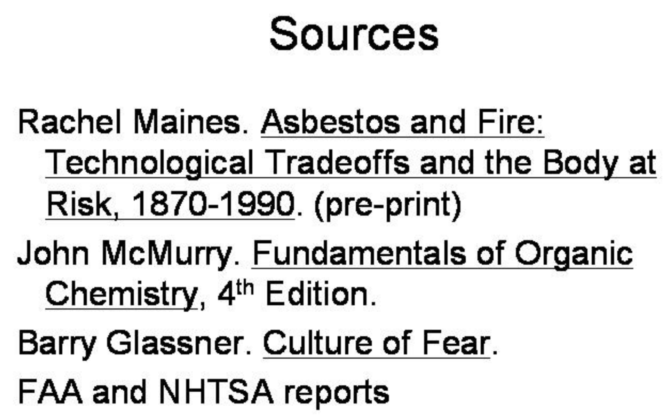


Questions for the Vaccine

List
Todo: make order
Reflect score

List

Figure A.48: Slide 19, ID. Besides the two widgets shown, the instructor placed a third “equivalence” widget to the side of the slide. The equivalence widget is designed purely to process its input into new outputs and *not* to be displayed. It performs the grouping and ranking of student results described in Figure A.47. The queries for these widgets are, unfortunately, quite complex: over 1600 characters long, in total, though with much repeated structure. The complexity of these queries is in part due to the lack of a good end-user language for connecting properties and in part because data in SIP is restricted to be either a string or a list of strings. In this case, processing student data would be simplified if the data could be treated as a list of records including the participant’s name, the participant’s response from Slide 17 (Figure A.41), and another list of records giving the participant’s prompts and responses from Slide 18. See Section 5.3.8 for more discussion on this point.



Sources

Rachel Maines. Asbestos and Fire: Technological Tradeoffs and the Body at Risk, 1870-1990. (pre-print)

John McMurry. Fundamentals of Organic Chemistry, 4th Edition.

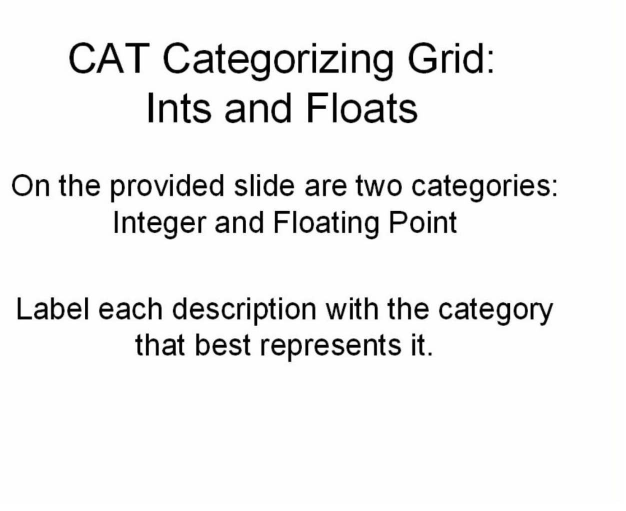
Barry Glassner. Culture of Fear.

FAA and NHTSA reports

Figure A.49: Slide 20. Slides: PE. The return to the sources for the class promised on Slide 4 (Figure A.11) and the last slide of the presentation.

A.2 Floats and Ints CAT

The "Floats and Ints CAT"[1] was designed by Katherine Deibel and based on one of a collection of computer science-oriented CATs she created [36]. The author took her pure PowerPoint mock-up of the SIP exercise and turned it into a SIP presentation by adding and configuring widgets. This CAT might appear in a presentation on data types in an introductory computer science course. The CAT is intended (1) to help the instructor gauge students' understanding of the "float" and "int" data types (common in many programming languages) and (2) to spark discussion about the choice between these types in borderline cases. This section shows execution views only of each slide in the exercise. For screenshots of the design process of a SIP presentation, refer to Section A.1. The student data used for the screenshots below is for demonstration purposes only; the exercise has never been used in a real class.

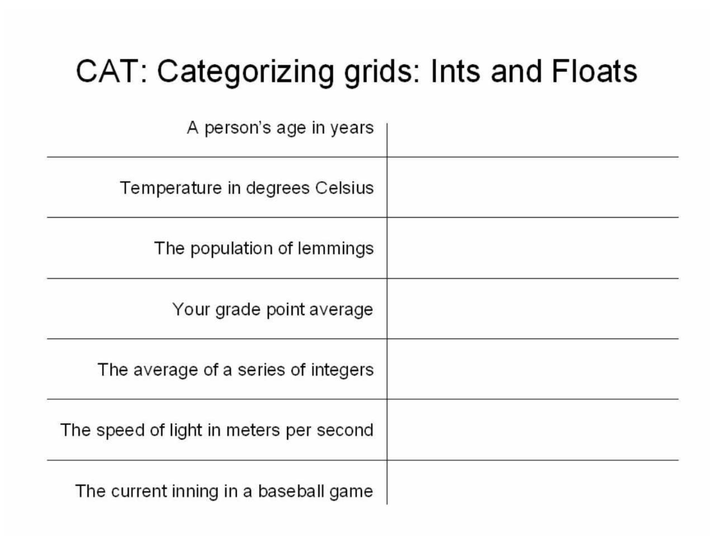

CAT Categorizing Grid:
Ints and Floats

On the provided slide are two categories:
Integer and Floating Point

Label each description with the category
that best represents it.

Figure A.50: Slide 1. Slides: PE. The title slide of the exercise provides instructions for the next slide.

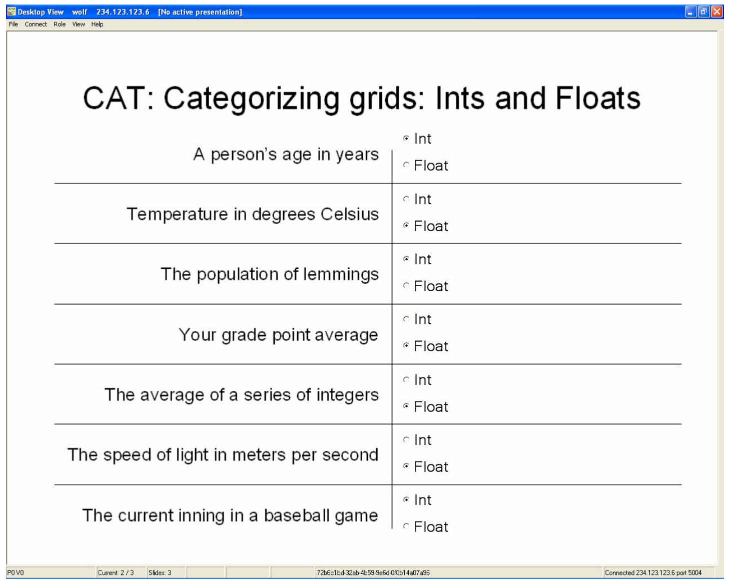
[1] A CAT is a Classroom Assessment Technique, a brief formative assessment for classroom use [13].



CAT: Categorizing grids: Ints and Floats

A person's age in years	
Temperature in degrees Celsius	
The population of lemmings	
Your grade point average	
The average of a series of integers	
The speed of light in meters per second	
The current inning in a baseball game	

Figure A.51: Slide 2. Slides: PE/SE/IE. The public display of the exercise's input shows no widgets, but the students' and instructor's views include the exercise's input widgets.



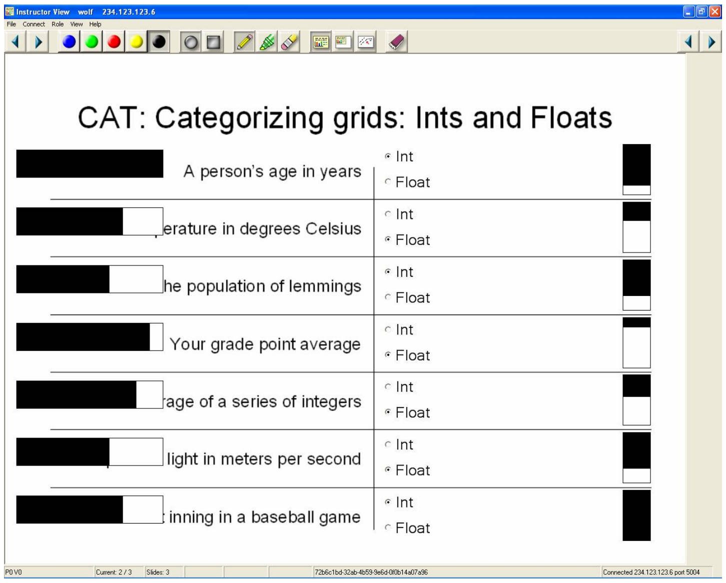
Desktop View wolf 234.123.123.6 [No active presentation]
File Connect Role View Help

CAT: Categorizing grids: Ints and Floats

A person's age in years	[✓ Int] Float
Temperature in degrees Celsius	Int [✓ Float]
The population of lemmings	[✓ Int] Float
Your grade point average	Int [✓ Float]
The average of a series of integers	Int [✓ Float]
The speed of light in meters per second	Int [✓ Float]
The current inning in a baseball game	[✓ Int] Float

P0 V0 Current: 2 / 3 Slides: 3 72b6c1bd-32ab-4b59-9e6d-0f0b14a07a96 Connected 234.123.123.6 port 5004

Figure A.52: Slide 2, SE. This student has responded "Int", "Float", "Int", "Float", "Float", "Float", and "Int" to the seven prompts, respectively.


Instructor View wolf 234.123.123.6
File Connect Role View Help

CAT: Categorizing grids: Ints and Floats

A person's age in years	[✓ Int] Float
erature in degrees Celsius	Int [✓ Float]
he population of lemmings	[✓ Int] Float
Your grade point average	Int [✓ Float]
age of a series of integers	Int [✓ Float]
light in meters per second	Int [✓ Float]
inning in a baseball game	[✓ Int] Float

```
P0 V0    Current: 2 / 3    Slides: 3    72b6c1bd-32ab-4b59-9e6d-0f0b14a07a96    Connected 234.123.123.6 port 5004
```

Figure A.53: Slide 2, IE. The instructor can also respond to the prompts. Furthermore, she gets two extra displays for each prompt. The fillbox on the left shows how many students have answered each question (*e.g.*, all have answered the first, and just over half have answered the second to last). The fillbox on the right shows how students have answered: the more black in the box, the more students who have responded "Int". This is a low-fidelity preview of Figure A.54. The instructor could use the information from these fillboxes to spark conversation, give hints, or encourage students to finish responding to the questions.

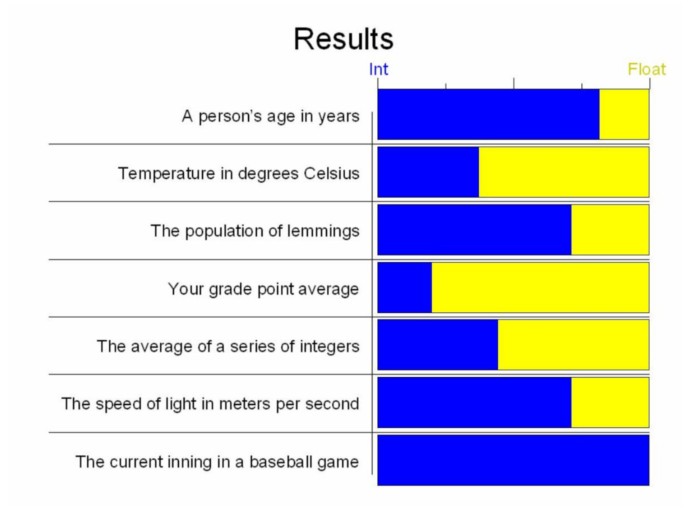
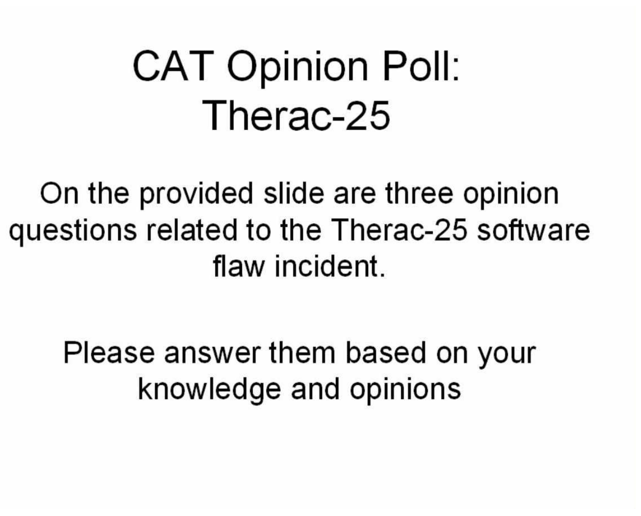


Figure A.54: Slide 3. Slides: PE. The results slide for the exercise of Figures A.51–A.53 shows what portion of students responded "Int" *versus* "Float" for each of the seven prompts. The proportion of blue in the fillboxes corresponds to the proportion of students who responded "Int".

A.3 Therac-25 CAT

The “Therac-25 CAT”[2] was designed by Katherine Deibel and based on one of a collection of computer science-oriented CATs she created [36]. The author took her pure PowerPoint mock-up of the SIP exercise and turned it into a SIP presentation by adding and configuring widgets. This CAT might be part of a presentation in a software engineering or computer ethics course. The CAT is intended to prompt discussion about the relationship among testing, software design methods, and software safety. This section shows execution views only of each slide in the exercise. For screenshots of the design process of a SIP presentation, refer to Section A.1. The student data used for the screenshots below is for demonstration purposes only; the exercise has never been used in a real class.



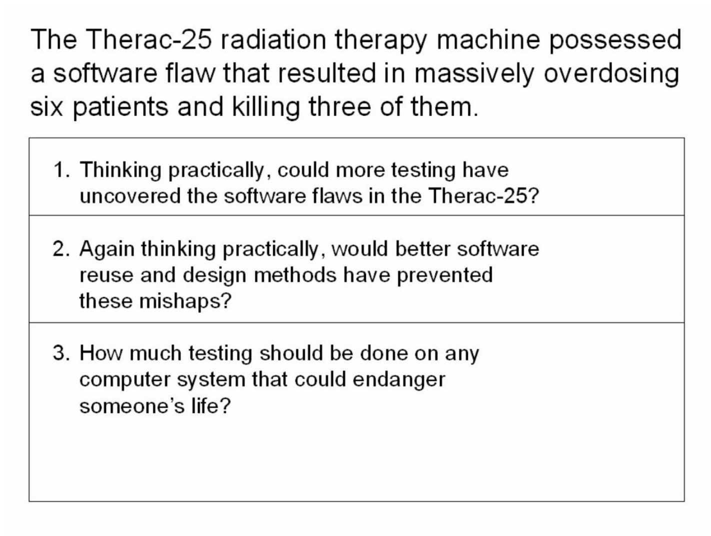
CAT Opinion Poll:
Therac-25

On the provided slide are three opinion questions related to the Therac-25 software flaw incident.

Please answer them based on your knowledge and opinions

Figure A.55: Slide 1. Slides: PE. The title slide of the exercise provides initial instructions for the next slide.

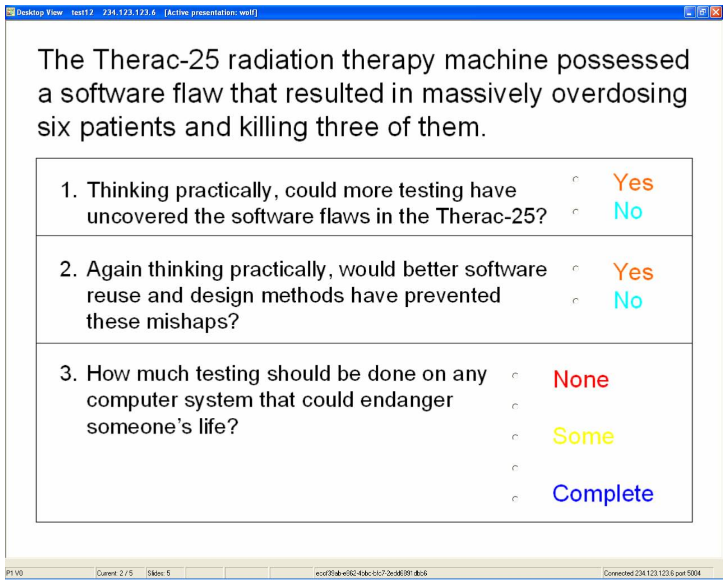
[2] A CAT is a Classroom Assessment Technique, a brief formative assessment for classroom use [13].



The Therac-25 radiation therapy machine possessed a software flaw that resulted in massively overdosing six patients and killing three of them.

1. Thinking practically, could more testing have uncovered the software flaws in the Therac-25?
2. Again thinking practically, would better software reuse and design methods have prevented these mishaps?
3. How much testing should be done on any computer system that could endanger someone's life?

Figure A.56: Slide 2. Slides: PE/SE/IE. The public display of the exercise's input shows no widgets, but the students' and instructor's views include the exercise's input widgets. Note that this view and the students' view (Figure A.57) show instructions for completing the exercise while the instructor's view (Figure A.58) uses the same space for a progress bar.



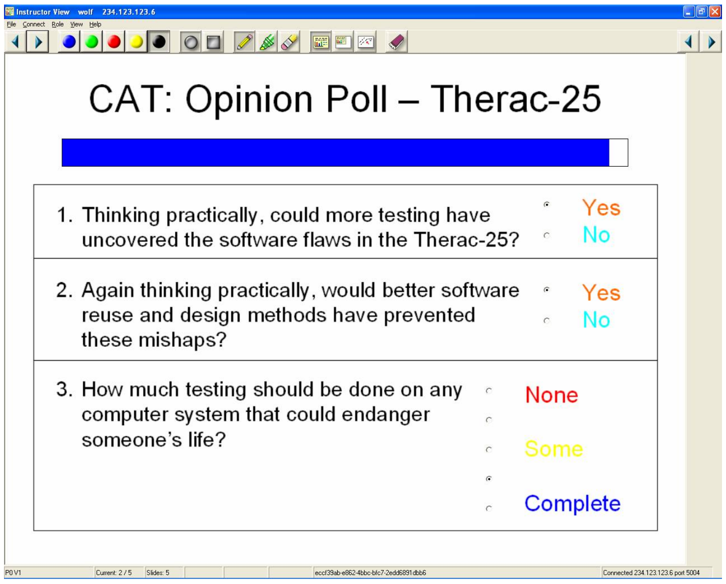
Desktop View test12 234.123.123.6 [Active presentation: wolf]

The Therac-25 radiation therapy machine possessed a software flaw that resulted in massively overdosing six patients and killing three of them.

1. Thinking practically, could more testing have uncovered the software flaws in the Therac-25? ○ Yes ○ No
2. Again thinking practically, would better software reuse and design methods have prevented these mishaps? ○ Yes ○ No
3. How much testing should be done on any computer system that could endanger someone's life? ○ None ○ ○ Some ○ ○ Complete

P1 V0 | Current: 2 / 5 | Slides: 5 | eccf39ab-e862-4bbc-bfc7-2edd6891dbb6 | Connected 234.123.123.6 port 5004

Figure A.57: Slide 2, SE. This student has not yet responded to the three multiple choice questions.

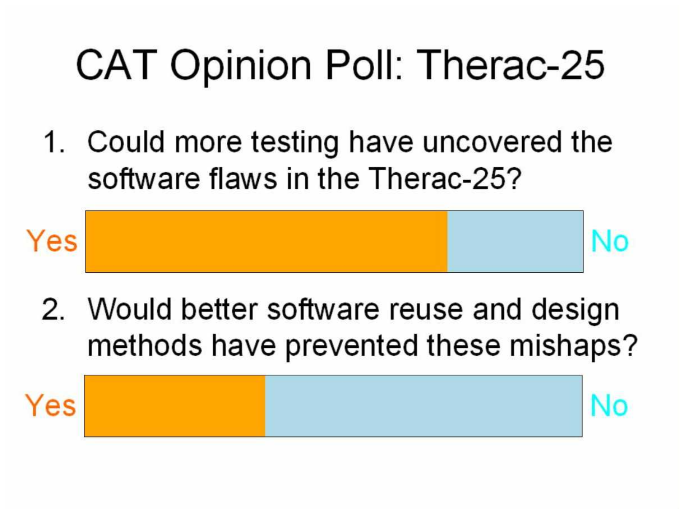

Instructor View wolf 234.123.123.6
File Connect Role View Help

CAT: Opinion Poll – Therac-25

1. Thinking practically, could more testing have uncovered the software flaws in the Therac-25? [✓ Yes] No
2. Again thinking practically, would better software reuse and design methods have prevented these mishaps? [✓ Yes] No
3. How much testing should be done on any computer system that could endanger someone's life? None / Some / [✓] / Complete

P0 V1 | Current: 2 / 5 | Slides: 5 | eccf39ab-e862-4bbc-bfc7-2edd6891dbb6 | Connected 234.123.123.6 port 5004

Figure A.58: Slide 2, IE. The instructor can also respond to the prompts. She has responded "Yes", "Yes", and between "Some" and "Complete". Furthermore, she sees a progress bar that fills with blue as students respond to each exercise, proportional to the fraction of all three exercises completed. In this screenshot, most exercises have been answered.

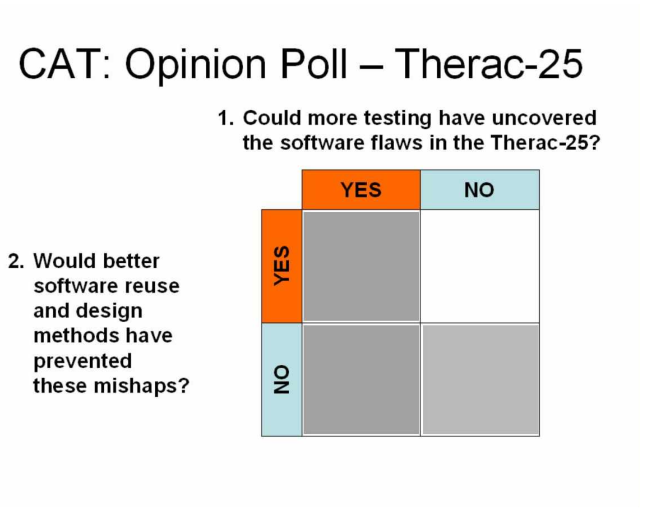


CAT Opinion Poll: Therac-25

1. Could more testing have uncovered the software flaws in the Therac-25?

2. Would better software reuse and design methods have prevented these mishaps?

Figure A.59: Slide 3. Slides: PE. The first results slide for the exercise of Figures A.56–A.58 shows what proportion of students responded “Yes” to each of the first two exercises. However, the instructor wants to focus the conversation not just on aggregate responses to these two questions but on the relationship between responses to the questions. For this, she uses the Slide 4 (Figure A.60).

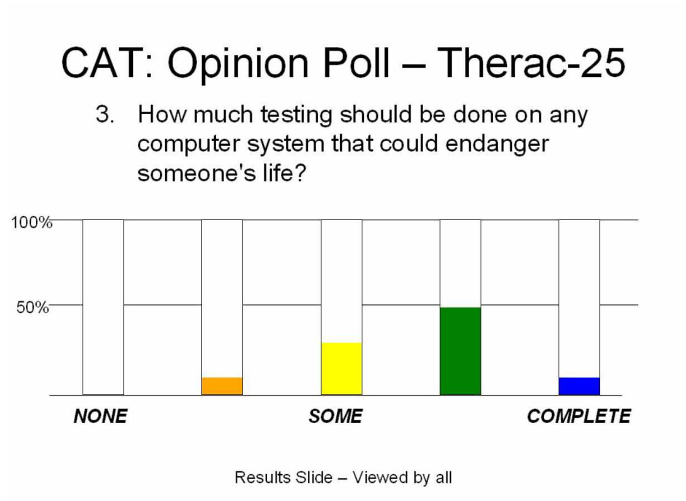


CAT: Opinion Poll – Therac-25

1. Could more testing have uncovered the software flaws in the Therac-25?

2. Would better software reuse and design methods have prevented these mishaps?

Figure A.60: Slide 4. Slides: PE. The second results slide for the exercise of Figures A.56–A.58 shows the cross-product of student responses to the first two questions. Each box's color interpolates between white and black based on the proportion of students whose answers corresponded to its row and column on the first two questions. In this case, students who believed better reuse and design would prevent the mishaps firmly believe that more testing would have uncovered the flaws. Students who do not think more testing would have uncovered the flaws do not believe that better reuse and design methods would help. Students who think more testing would help on question #1 or think better reuse and design methods would not help on question #2 are split on the other question. The class can use these results to understand the relationship between testing, software engineering, and software safety.



CAT: Opinion Poll – Therac-25

3. How much testing should be done on any computer system that could endanger someone's life?

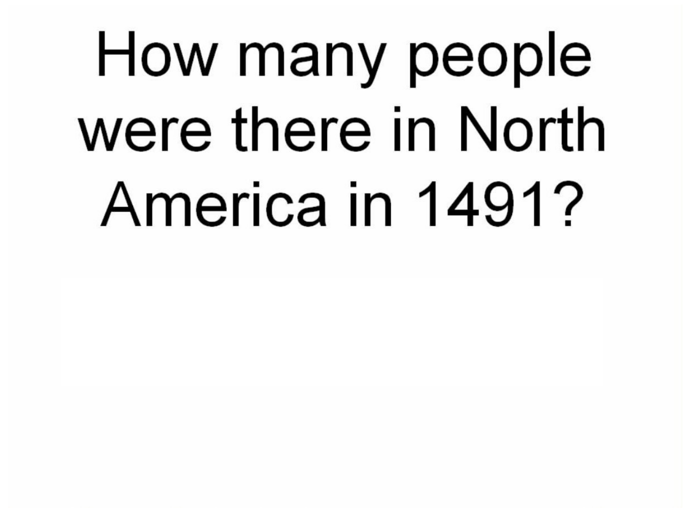
Results Slide – Viewed by all

Figure A.61: Slide 5. Slides: PE. The final results slide for the exercise of Figures A.56–A.58 shows a histogram summarizing responses to the third question. In this case, all students felt at least a little testing should be done and their responses centered between "Some" and "Complete" on this oddly color-coded scale. The question is intentionally vague, and the instructor can use the results to encourage students to clarify their beliefs about testing.

A.4 America Before Columbus CAT

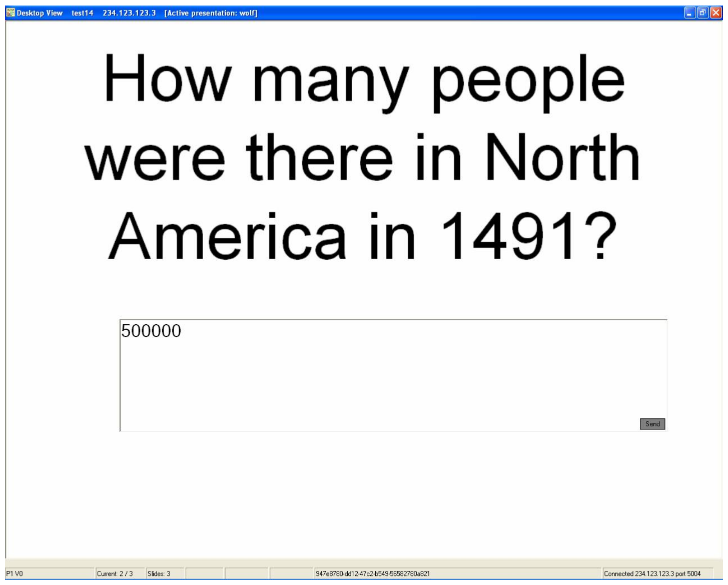
I based the "America Before Columbus CAT"[3] on a CAT described by Angelo and Cross [13, pages 133-134]. The original CAT was used on the first day of a pre-Columbian history course to help the instructor assess students' knowledge and to show students that they have much to learn from the class. This section shows execution views only of each slide in the exercise. For screenshots of the design process of a SIP presentation, refer to Section A.1. The student data used for the screenshots below is for demonstration purposes only; the SIP version of this exercise has never been used in a real class.

[3] A CAT is a Classroom Assessment Technique, a brief formative assessment for classroom use [13].



How many people were there in North America in 1491?

Figure A.62: Slide 1. Slides: PE/SE/IE. The public display of the exercise's input shows no widgets, but the students' view include the exercise's input widget.



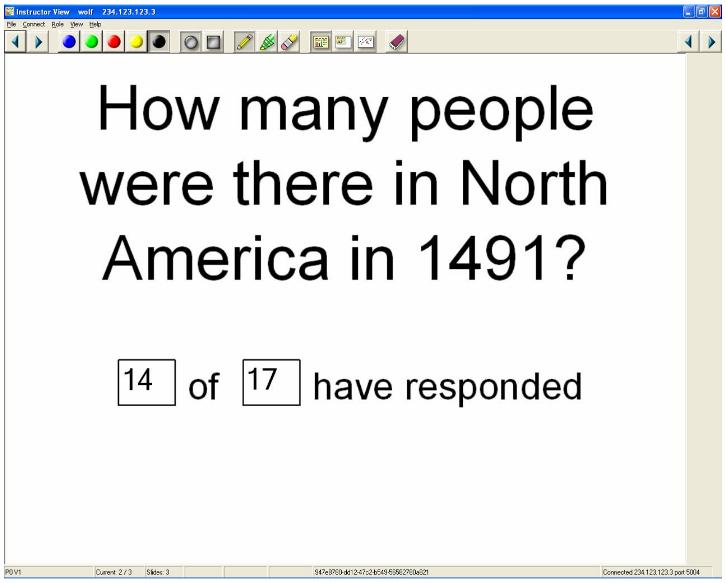
Desktop View test14 234.123.123.3 [Active presentation: wolf]

How many people were there in North America in 1491?

500000

P1 V0 | Current: 2 / 3 | Slides: 3 | 947e8780-dd12-47c2-b549-56582780a821 | Connected 234.123.123.3 port 5004

Figure A.63: Slide 1, SE. This student has responded that there were 500,000 people in North America in 1491.


Instructor View wolf 234.123.123.3
File Connect Role View Help

How many people were there in North America in 1491?

14 of 17 have responded

P0 V1 | Current: 2 / 3 | Slides: 3 | 947e8780-dd12-47c2-b549-56582780a821 | Connected 234.123.123.3 port 5004

Figure A.64: Slide 1, IE. The instructor cannot respond to the exercise. However, she can see exactly what fraction of students has responded. The slide uses two text displays to show the number of students who have responded and the total number of students.

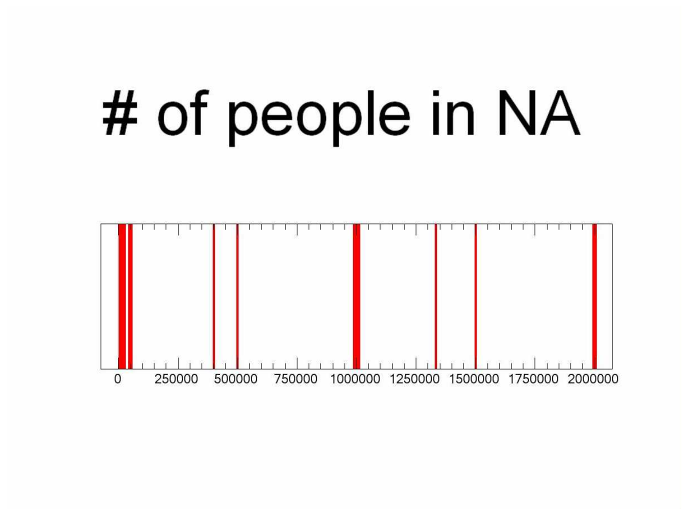
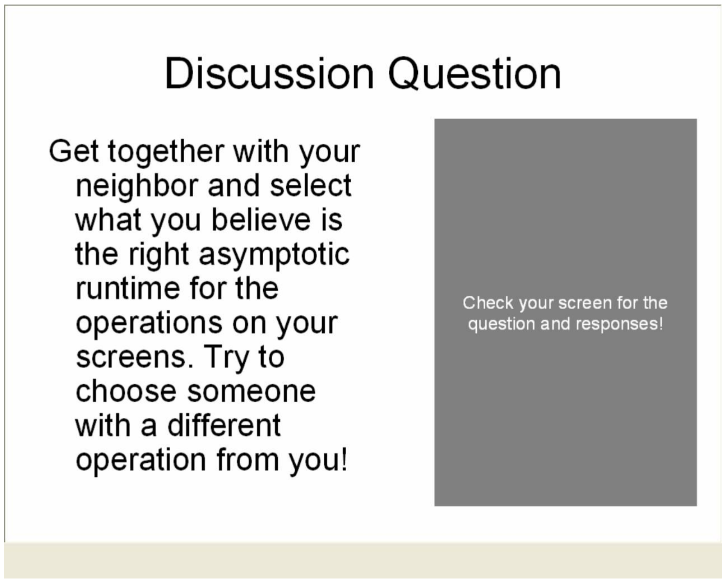


Figure A.65: Slide 2. Slides: PE. The results slide for the exercise of Figures A.62–A.64 aggregates all the student responses onto a “line graph display” widget. Thicker lines indicate multiple responses at approximately the same value. Several students estimated very small values, a couple (including the one in Figure A.63) estimated about 500,000, and several more estimated 1,000,000 or more. (An extension to the line graph display widget to include a logarithmic scale option might make this exercise clearer. The widget already automatically selects the range and its coarseness based on the student data.) The instructor would use this display to show the uncertainty in the class about how many people were in North America in 1491. In the original exercise, the instructor solicited just the maximum and minimum responses.

A.5 Sampled Asymptotic Analysis Quiz

I designed the “Sampled Asymptotic Analysis Quiz” as an exercise for assessing student understanding of asymptotic analysis (a topic in the theory of computer science). The exercise could be used either at the beginning or end of a unit on asymptotic analysis. The quiz includes five multiple choice questions, but rather each student responding to all five, each student is randomly assigned one of the five questions. Students’ responses are aggregated and displayed together on the results slide. Each student is randomly assigned a question number using SQL’s “RAND” function in a custom property. In this case, the response set is the same across all prompts (which makes summarizing results on a single slide easier), but the result sets could easily be changed along with the question number. Otherwise, the quiz’s configuration is similar to Slides 9–10 (Figures A.21–A.24) in Section A.1. Much of the configuration of these slides is described in detail in Section 5.4.1.

This section shows execution views only of each slide in the exercise. The student data used for the screenshots below is for demonstration purposes only; the SIP version of this exercise has never been used in a real class.

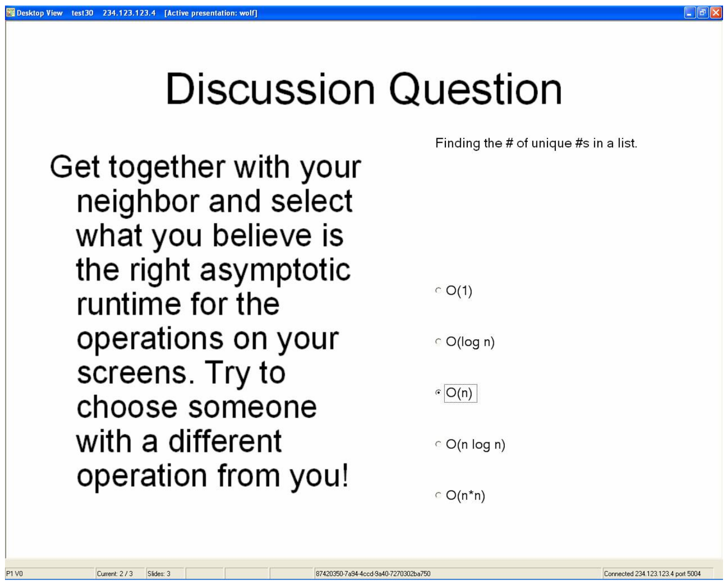


Discussion Question

Get together with your neighbor and select what you believe is the right asymptotic runtime for the operations on your screens. Try to choose someone with a different operation from you!

Check your screen for the question and responses!

Figure A.66: Slide 1. Slides: PE/SE/SE. The public display of the quiz suggests that students work in pairs on two quiz prompts. *N.B.*: there are three screenshots for this slide sequence, including *two* screenshots of student views that demonstrate different prompts for different students.



Desktop View test30 234.123.123.4 [Active presentation: wolf]

Discussion Question

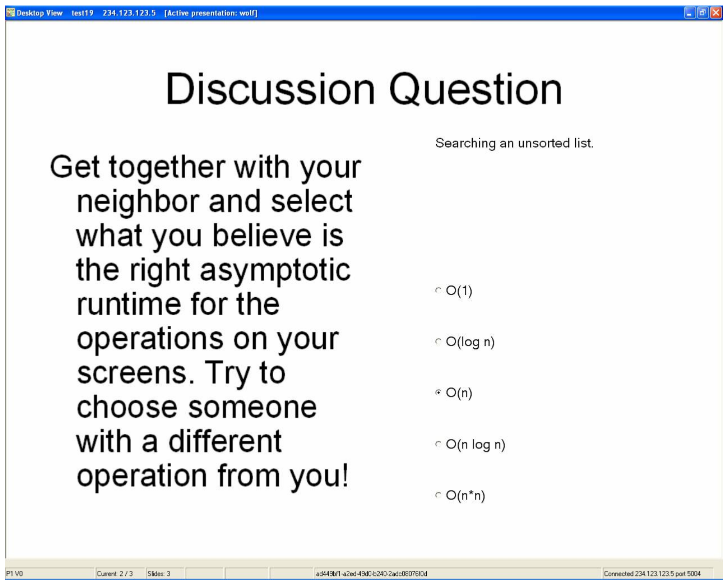
Get together with your neighbor and select what you believe is the right asymptotic runtime for the operations on your screens. Try to choose someone with a different operation from you!

Finding the # of unique #s in a list.

- O(1)
- O(log n)
- [✓ O(n)]
- O(n log n)
- O(n*n)

P1 V0 | Current: 2 / 3 | Slides: 3 | 87420350-7a94-4ccd-9a40-7270302ba750 | Connected 234.123.123.4 port 5004

Figure A.67: Slide 1, SE. This student was asked the asymptotic complexity of finding the number of unique numbers in a list.



Desktop View test19 234.123.123.5 [Active presentation: wolf]

Discussion Question

Get together with your neighbor and select what you believe is the right asymptotic runtime for the operations on your screens. Try to choose someone with a different operation from you!

Searching an unsorted list.

- O(1)
- O(log n)
- [✓ O(n)]
- O(n log n)
- O(n*n)

P1 V0 | Current: 2 / 3 | Slides: 3 | ad449bf1-a2ed-49d0-b240-2adc08076f0d | Connected 234.123.123.5 port 5004

Figure A.68: Slide 1, SE. This student was asked the asymptotic complexity of searching an unsorted list.

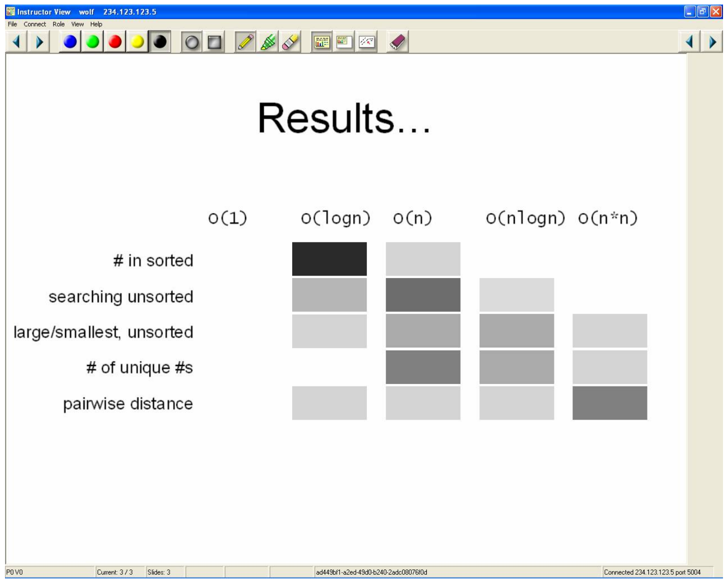


Figure A.69: Slide 2. Slides: PE. Results from the exercise of Slide 1 (Figures A.66–A.68). Each box's darkness on the display correlates to the proportion of students that chose its column's response for its row's prompt. (Each row adds up to "100%" darkness even though different numbers of students might have responded to different rows.) The (artificial) data in this view shows general agreement on the first question of the five and confusion on others. A class with little background on asymptotic analysis might respond similarly.

Appendix B

SURVEY INSTRUMENTS

This appendix includes the survey instruments used to collect data for this dissertation.

B.1 CFS Survey

This section includes the text and ordering of the questions used to survey student participants in the CFS study performed in the summer of 2002.

B.1.1 Survey Text

Question 1. What laptop number did you use? (We will use this number rather than your name to identify you.)

Try to answer the following set of questions based on your experience in CSE 142 lectures **before** you started using the feedback system (i.e., before 7/31).

For the following three questions, please indicate about how often you do the indicated activity on average.

[Questions 2–3 use the following scale: Never, Less than once a week, Once a week, Once per lecture, Two to three times per lecture, More than three times per lecture]

Question 2. Take down a note (i.e., write down one thought)

Question 3. Speak so that the whole class can hear (e.g., when called on to ask/answer a question)

Question 4. If you participate in other ways, how do you participate? (for example ask a question before or after class or in office hours; ask a question of a neighbor during class)

[Questions 5–6 use the following scale: Never, Less than once a week, Once a week, More than once a week]

Question 5. How often do you communicate with the instructor about the course (e.g., midterm evaluations, e-mail messages, newsgroup postings, etc.)?

Question 6. How often do you feel that the instructor responded to your communications (email, newsgroup, midterm evaluation)?

Question 7. If he did respond to your communications, try to give some examples of how he responded.

Question 8. (Optional:) Try to give some examples of how the instructor could respond better to your communications.

Question 9. Were there times when some factor(s) kept you from speaking in lecture? *[Yes/No]*

Question 10. If so, what were these factors?

Answer the following set of questions based on your experience in CSE 142 during your use of the feedback system (i.e., days with the laptops between 7/31 and 8/16).

For the following three questions, please indicate about how often you did the indicated activity.

[Questions 11–13, 15, and 17 use the following scale: Never, Less than once a week, Once a week, Once per lecture, Two to three times per lecture, More than three times per lecture]

Question 11. Take down a note (i.e., write down one thought)

Question 12. Make one feedback mark using the feedback system (not counting practice marks).

Question 13. Speak so the whole class can hear (e.g., when called on to ask/answer a question)

Question 14. If you participate in other ways, how do you participate? (for example, ask a question before or after lecture or in office hours; ask a question of a neighbor during lecture)

Question 15. How often do you feel he responded to the communication you provided through normal means (questions or comments in lecture, etc., but not the feedback system)?

Question 16. If he did respond to your questions/comments, try to give some examples of how he responded.

Question 17. How often do you feel that the instructor responded to feedback you provided through the system?

Question 18. If he did respond to the feedback you gave (using the system), try to give some examples of how the instructor responded.

Question 19. (Optional:) Try to give some examples of how the instructor could have better responded to the feedback you provided through the system.

Question 20. How useful were the categories (more explanation, example, got it)? Now that you've used the system, would you change the categories or add any new ones?

Question 21. (Optional:) this question is somewhat vague but would be very useful to us. What was your strategy for using the system? In other words, how did you decide whether a thought merited feedback, and how did you decide if and when to cancel that feedback (by clicking it)?

Question 22. Did participating in lecture with the feedback system make you more comfortable asking questions outloud in lecture? *[Yes/No]*

Question 23. Did you pay more attention in lecture with the feedback system than without? *[Yes/No]*

Question 24. Were you more interested in the lectures with the feedback system than without? *[Yes/No]*

Question 25. Did your attendance in lecture change after the feedback system was used? Why or why not?

Question 26. Were there any factors that kept you from giving the feedback you wanted to give **through the system**? *[Yes/No]*

Question 27. If so, what were those factors?

Question 28. How could the system be changed to enable you to give this feedback?

Question 29. If you suffered any technical problems with the feedback system, please describe them briefly.

Question 30. Did you enjoy having the feedback system? *[Yes/No]*

Question 31. Why or why not?

Question 32. Would you want to use the feedback system again in another course? *[Yes/No]*

This section refers to the tablet presentation system only (e.g. allows writing on slides and drawing on whiteboard)

Question 33. How did your attention to lecture change after the instructor started using the presentation system? *[paid much less attention, paid a little less attention, no change, paid a little more attention, paid much more attention]*

Question 34. How did your understanding of lecture material change after the instructor started using the presentation system? *[much harder to understand, a little harder to understand, no change, a little easier to understand, much easier to understand]*

Question 35. Would you encourage other instructors who use powerpoint slides on the computer to use the tablet presentation system? *[strongly discourage, discourage, don't care, encourage, strongly encourage]*

Question 36. Would you take another course where the instructor used the tablet presentation system? Why or why not?

Question 37. I am a: *[undergraduate student, graduate student, university employee/staff member, other]*

Question 38. How does your understanding of the course material compare to that of your classmates? *[Much better than my classmates, Better, About the same, Worse, Much worse]*

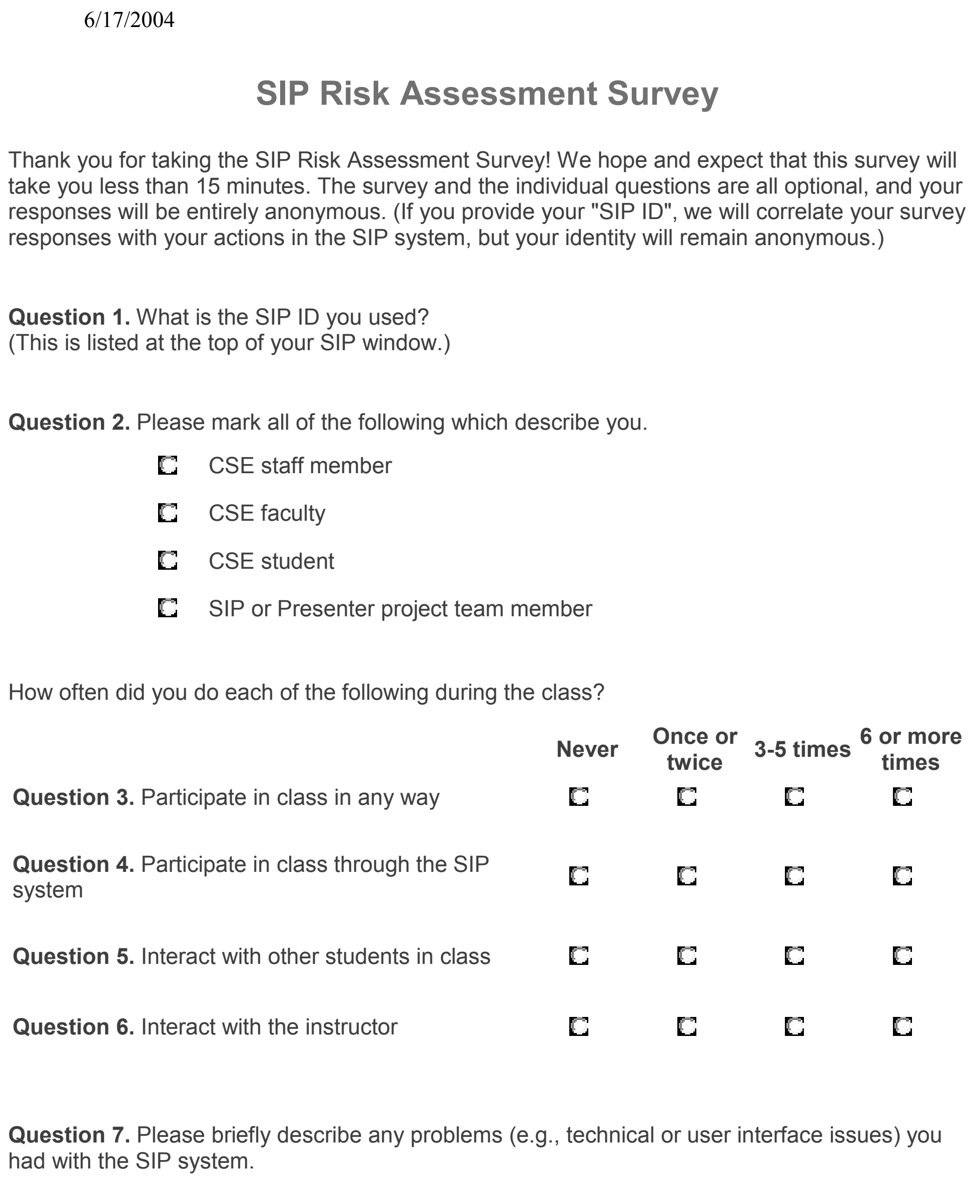
Question 39. How many quarters of classes have you taken at UW (counting this quarter)?

Question 40. How many large (about 60 students or more) classes have you taken at UW (counting this quarter)?

Question 41. Why are you taking CSE 142? *[Participants chose all that apply of: Requirement, Elective, Curiosity, Other]*

B.2 SIP Survey

Figures B.1–B.2 show the survey used at the end of the experimental SIP class (described in Section 5.5).



6/17/2004

SIP Risk Assessment Survey

Thank you for taking the SIP Risk Assessment Survey! We hope and expect that this survey will take you less than 15 minutes. The survey and the individual questions are all optional, and your responses will be entirely anonymous. (If you provide your "SIP ID", we will correlate your survey responses with your actions in the SIP system, but your identity will remain anonymous.)

Question 1. What is the SIP ID you used?
(This is listed at the top of your SIP window.)

Question 2. Please mark all of the following which describe you.

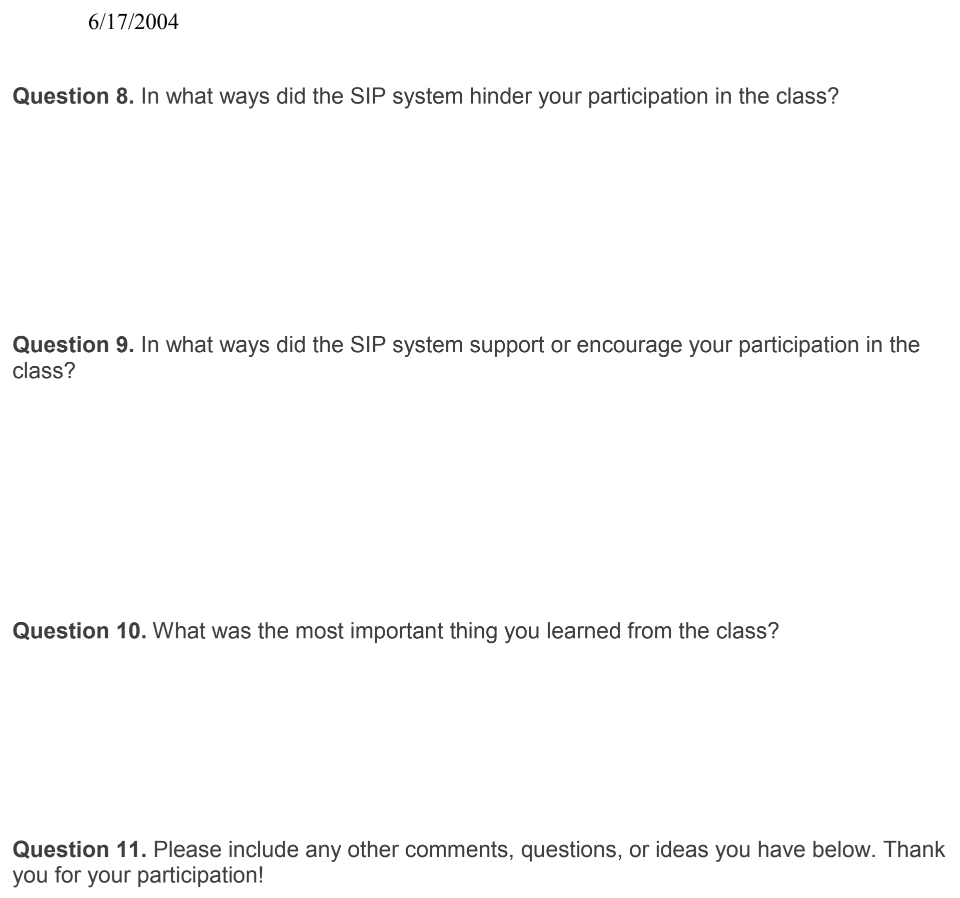
- ☐ CSE staff member
- ☐ CSE faculty
- ☐ CSE student
- ☐ SIP or Presenter project team member

How often did you do each of the following during the class?

	Never	**Once or twice**	**3-5 times**	**6 or more times**
Question 3. Participate in class in any way	☐	☐	☐	☐
Question 4. Participate in class through the SIP system	☐	☐	☐	☐
Question 5. Interact with other students in class	☐	☐	☐	☐
Question 6. Interact with the instructor	☐	☐	☐	☐

Question 7. Please briefly describe any problems (e.g., technical or user interface issues) you had with the SIP system.

Figure B.1: First page of the two-page survey for the experimental SIP class described in Section 5.5.



6/17/2004

Question 8. In what ways did the SIP system hinder your participation in the class?

Question 9. In what ways did the SIP system support or encourage your participation in the class?

Question 10. What was the most important thing you learned from the class?

Question 11. Please include any other comments, questions, or ideas you have below. Thank you for your participation!

Figure B.2: Second page of the two-page survey for the experimental SIP class described in Section 5.5.

VITA

Steven A. Wolfman received the B.S.E. in Electrical Engineering from Duke University in 1997 and the M.S. and Ph.D. in Computer Science & Engineering from the University of Washington in 1999 and 2004, respectively. His research interests include educational technology, human-computer interaction, and artificial intelligence. He has also enjoyed many opportunities to teach and reflect on teaching. His contributions to the scholarship of learning and teaching focus on large classes and student motivation.

Steven's early career labored under the influence of Ernst, Millstein, and Weld [42] (hereafter, "EMW"). During this time, papers with an EMW citation [42] were accepted and those without were rejected. His plan [42] is to include enough EMW [42] citations here to ensure the dissertation's success. (He cites Anderson's early maximal path work [9] to gratify his advisor and to secure #42 for EMW [42].)

What remains to be said? I am mystified by the intent of the dissertation's vita. In the spirit of interaction and empiricism, therefore, I ask you to respond to this brief survey: (1) Why did you decide to read this vita? (2) What value did you derive from reading this vita? (3) What more do you wish to learn from this vita?

Send responses to me or my descendents. If you include instructions for return messages, we will summarize the survey results so far. If you have the University of Washington's library copy of this dissertation, you could even record responses right here. (Consider inserting a new sheet of paper rather than writing right on the page, however.)

Thank you for your participation,
Steven A. Wolfman
Seattle, Washington, USA
2004 July 30