Combining Distinct Graduate and Undergraduate HCI Courses: An Experiential and Interactive Approach

Christopher Brown
Michigan Technological University
Houghton, MI
cdbrown@mtu.edu

Robert Pastel
Michigan Technological University
Houghton, MI
rpastel@mtu.edu

ABSTRACT
We developed combined graduate and undergraduate courses in which undergraduates created a prototype based on user-centered design, and graduate students worked with them to evaluate those prototypes based on common usability principles. It provided undergraduate students experience practicing user centered design, while providing graduate students experience with usability evaluation. It also provided graduate students the opportunity to introduce current HCI research areas to students who may be considering graduate school. The course successfully engaged both graduate and undergraduate students while providing a beneficial experience through their interactions.

Categories and Subject Descriptors
D.2.2 [Software Engineering]: Design Tools and Techniques - evolutionary prototyping;
K.3.2 [Computers and Education]: Computer and Information Science Education - computer science education, curriculum;
H.5.2 [Information Interfaces and Presentation]: User Interfaces - prototyping, user-centered design

General Terms
Experimentation, Measurement, Human Factors

Keywords
Combined Course, Graduate Course, HCI, Interactions

1. INTRODUCTION
A combined course is one in which students enrolled in distinct courses gather in the same room, making insights on the same topics. When combining two courses, it is important to consider the reason they are being combined, and the purpose for each course. For these combined courses to be truly distinct and not cross-referenced courses, such as the multi-faceted course in [7], the students enrolled in the two courses should benefit in different ways through the development of distinct ideas and skills.

Frequently, graduate (grad) and undergraduate (undergrad) courses are combined by requiring the grad students to write an additional term paper and make a presentation to the class. Combining the courses by requiring more work from the grad students does provide higher standards and ensure a richer learning experience for all students. However, after taking one course, neither grads nor undergrads are inclined to enroll in the counterpart in subsequent semesters. Our standard for combining the grad and undergrad Human-Computer Interaction (HCI) courses was to ensure that the learning experience for both the grad and undergrad students was distinct enough that any student could benefit from, and would desire to take, both courses.

HCI is a particularly promising area for combining distinct grad and undergrad courses because it is a broad discipline requiring many skills to implement a successful user interface (UI). HCI is the convergence of at least three disciplines: software engineering, design, and cognitive science[5]. In previous iterations of our undergraduate course, the emphasis was on UI design[5]. However, an interface designer should not only be a skilled programmer, talented in visual and interactive design, but also be able to conduct and evaluate how users will interpret their design. The earlier course did not provide adequate experience evaluating and testing their UIs, and consequently, the HCI students were convinced that their designs were satisfactory despite key usability concerns.

The skills required to implement a successful UI can be divided into two parts:

1. UI design and implementation
2. Evaluation, usability testing, and analysis

However, if a group performs evaluation, testing and analysis on their own project, many of the key usability concerns which were ignored through the design would also be ignored through testing. To have students both develop their own prototype and evaluate others would require a very large time commitment to the course. Although time might permit creating a low-fidelity prototype for this purpose, current research suggests that the benefit of testing high-fidelity prototypes provides more compelling evaluations[2, 8].

The division between developer and evaluator is a natural one to follow when dividing the grad and undergrad courses. The undergrad students at our university are especially proud of their programming skills, and they are eager to design and implement their own unique UI. Furthermore,
grad students are expected to have better analytical skills than undergrad students. Consequently, students enrolled in the undergrad course could design and implement a UI, while students enrolled in the grad course evaluate and test those UIs. This provides an additional level of cooperative learning which is rarely seen in the classroom[1].

Besides ensuring that the combined course provides a distinct and meaningful learning experience for all enrolled students, we had other goals for the combined course that reflected the scheduling and instructor’s workload constraints. One goal was to do external user testing on a high-fidelity prototype. The undergrad groups should have sufficient time to produce high-fidelity UI prototypes suitable for usability testing. Meanwhile, grad students should have sufficient time to conduct the usability test and analyze the test results before the end of the semester. In addition, undergrad groups should receive feedback on their design from several grad students and not feel dominated by any single grad student. Finally, a graduate-level course should provide the grad students the opportunity to study current HCI research topics.

By creating two combined, yet distinct, HCI courses, we can emphasize the importance of design, software engineering, and cognitive evaluation in creative projects. Through the course, we emphasize the benefits of the interaction between grad and undergrad students, as well as the project development by the undergrad students and evaluation process by the grad students.

2. AN EXPERIENCE BASED HCI COURSE

UI design is plagued by programmers who believe that their programs are intuitive and natural, despite problems obvious to HCI experts. To discourage this egocentrism, there is incentive to provide constructive feedback to those programmers. However, without supporting evidence, any constructive feedback given to the programmer, no matter how theoretically sound, is often ignored. One goal, then, is to provide experimental evidence to all students, providing a deeper understanding of the importance of an intuitive user interface.

We introduced a graduate level course as a means of providing real, experimental evidence for students learning interface design in the undergrad course that was already offered. The graduate course was primarily an evaluation course, where the students use projects from the undergrad course to practice analysis of design proposals, heuristic evaluation of low-fidelity prototypes, and experimental study of high-fidelity prototypes. This gave the designers—the undergrad students—meaningful usability feedback on a working project.

2.1 Course Descriptions

The undergrad HCI course was a group project course that had been run in previous semesters[5]. Each group designs and implements a high-fidelity prototype application for a “Tiny Digital Assistant” (TDA), a very small device with only five hardware buttons, one of which is the power button, and limited screen space (320 x 240 pixels). Students worked through an iterative design process, including an initial proposal, a low-fidelity prototype, and a high-fidelity prototype.

The corresponding graduate HCI course was an experimental course. Graduate students took on the role of “expert evaluators”—usability experts who push groups to focus on user-centered design—ultimately testing the final prototype with sample users, and analyzing the results. The graduate students also explored current research areas in HCI, presenting their explorations to all students, including the undergrad students. Their exploration culminated in a short (5-10 page) written assignment on the subject. See [6], the course website, for a complete schedule and detailed description of the assignments.

2.2 Student Interactions

Many of the assignments served two purposes in the course. They were used as graded material and as the method of communication between the grads and the undergrad groups. Both the groups and the grads created websites to publish their documentation for others to review. This allowed everyone immediate and complete access to all of the documentation for each project.

The course was divided into three major sections, which are further divided into phases, as shown in Fig. 1. During each section, grads worked with a different undergrad group, reporting their findings back to that group. The undergrads posted their design documents and the grad student’s feedback publicly, which allowed all groups to learn from each others’ projects. As the class went on, the undergrads could better understand each of the grads’ unique perspectives, and use those to develop their projects. Overall, this provides some consistency while still offering varied viewpoints.

2.2.1 Initial Phases

In the initial phases of the course, the grads and undergrads had separate, but nearly identical functions within the class. The instructor lectured on fundamentals of HCI, while the students prepared for their distinct roles. First,
Each group and grad student was required to create a proposal and a website (Fig. 1, Phase 1). This gave them an idea of what they should be preparing for the course—the undergrad’s group project and the grad’s research presentation and paper.

The undergrad groups were asked to think about the primary users and the tasks they would undertake, to create a User–Task Analysis (Fig. 1, Phase 2). This helped the undergrad groups focus on user-centered design for the remainder of the project. Meanwhile, the graduate students read one group’s proposal, and met with that group to discuss the project and write up a User-Task-Goal Analysis (UTGA). Their function was to decide who the primary, secondary and tertiary users were, what goals they would have, and tasks necessary to achieve their goals. Then, those reports were provided as communication between the grad students and their assigned group.

2.2.2 Intermediate Phases

During the intermediate phases of the course, the grad students worked independently of the undergrad groups. There were only a few lectures by the instructor, since student presentations comprised most of the classroom time. The undergraduate students created a low-fidelity prototype (many of the groups created paper-prototypes), and presented this as a cognitive walkthrough to the class (Fig. 1, Phase 3). Each grad student was assigned to two groups, evaluating each prototype using basic heuristics.[4]

After the grad students provided their evaluations to the undergrad groups, the undergrads worked at developing their prototype outside of class. Meanwhile, the graduate students developed their research topics into presentations for the class (Fig. 1, Phase 4). Though these presentations did not directly relate to the projects, it did provide a means for potential grad students to learn more about the current research state of HCI, and specific areas they might be interested in researching.

2.2.3 Final Phases

As the course was coming to a close, the instructor returned with additional lectures on advanced HCI topics—those that were not presented as part of the graduate’s research topics. Outside of class, the undergrads spent time developing their high-fidelity prototype, while the grads prepared for the usability testing (Fig. 1, Phase 5). Since the grads and undergrads were preparing concurrently, the grads could get updates on the status of the prototype. They could also offer suggestions to the undergrads about providing more depth in certain aspects of the high-fidelity prototype for testing. Undergrad groups would then know which other aspects could be left either incomplete or as a shallow prototype. Many of the grad students also prepared monitoring programs which would run concurrently with the prototype, allowing them the opportunity to analyze their tests more closely.

During the final testing phase (Fig. 1, Phase 6), the interaction between the grads and undergrads culminated. Students from lower level CS courses volunteered to participate for a small grade incentive. Grad students acted as the facilitators for the user testing, while the group members recorded data and monitored their prototypes for errors. This phase fulfilled many of the positive goals for the course to an extent beyond our expectation.

3. Example Projects

This iteration of the course consisted of 30 undergrads broken into 7 small groups, and 7 grads working individually. All of the undergrads were Juniors or Seniors in the CS program. They had also completed a prerequisite of Team Software Project, a group-based software design and quality assurance course. All grad students were in the CS graduate school, with one exception from the Electrical Engineering program. Two of the CS grad students had taken the undergrad course before it was combined with the grad course.

All of the groups successfully created working prototypes, which were then tested for usability with cooperation from students in introductory computer science courses. Two groups included a tactile aspect in their final prototype, allowing the user to test some of the physical constraints of using the device. The wisdom imparted by user testing was not limited to these tactile interfaces, though. In a third project, which was completely simulated on a desktop computer, students studied the difficulties inherent in designing expected functionality in a novel addition to a well-known device.

3.1 Mini-Mote

One project with a physical aspect was Mini-Mote, a universal remote-control application with a dynamic touchscreen interface. The Mini-Mote had built in controls for Televisions, DVD-players, and VCRs. The intended application is to be customizable for any device which accepts remote signals. Since the screen size was very small, and was intended for a human hand, the group developing Mini-Mote limited their menus to four items, one in each quadrant of the device.

In user testing, the group created a prototype with a touchscreen (10in, 800x400 resolution) attached as a second monitor to a laptop. As can be seen in Fig. 2, the user controlled the “television” with a touchscreen remote control. They were asked to perform a series of tasks, such as turning on the TV, changing the channel, even adding a new device. Meanwhile, a background process counts the users clicks, recording the time and correctness of the buttons clicked.
There were some small differences between the prototype and the actual device. Since the resolution was fixed in the device description, and the resolution of the touchscreen was more compact, the device was approximately half the physical size that was intended. The group chose to use a touch-pen to overcome the difference in size and protect the touchscreen from damage.

The grad student for their group collected his experimental data, and created a plot of time against missed clicks (Fig. 3) to determine the probable cause of frustration or misclicks. Each task was identified by number, while letters represent notes from observers about actions the user took, questions the user asked, or visual cues of the user’s emotion. This allowed the grad student to discern the problems with the device that caused the most errors.

3.2 Location Aware Remote Control (LARC)

Another project with a tactile aspect to their prototype was the Location Aware Remote Control (LARC). LARC is a server maintenance application which used Radio Frequency Identification (RFID) cards to decide which server it was controlling. The application used large icons and very few buttons on the touchscreen to avoid errors. The most interesting aspect was the use of real RFID cards which represented the user selecting different servers to make modifications.

There were some small differences between the prototype and the actual device. Since the resolution was fixed in the device description, and the resolution of the touchscreen was more compact, the device was approximately half the physical size that was intended. The group chose to use a touch-pen to overcome the difference in size and protect the touchscreen from damage.

The grad student for their group collected his experimental data, and created a plot of time against missed clicks (Fig. 3) to determine the probable cause of frustration or misclicks. Each task was identified by number, while letters represent notes from observers about actions the user took, questions the user asked, or visual cues of the user’s emotion. This allowed the grad student to discern the problems with the device that caused the most errors.

### Location Aware Remote Control (LARC)

Another project with a tactile aspect to their prototype was the Location Aware Remote Control (LARC). LARC is a server maintenance application which used Radio Frequency Identification (RFID) cards to decide which server it was controlling. The application used large icons and very few buttons on the touchscreen to avoid errors. The most interesting aspect was the use of real RFID cards which represented the user selecting different servers to make modifications.

<table>
<thead>
<tr>
<th>Category</th>
<th>Problem</th>
<th>User 1</th>
<th>User 2</th>
<th>User 3</th>
<th>User 4</th>
<th>User 5</th>
<th>User 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Usage</td>
<td>Scanned using the face of the device</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Struggled finding the RFID tags with reader</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Having to scan/lock each server during search</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>The Interface</td>
<td>Confused about usage of Stop button</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lack of instructional cues</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lack of feedback</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Lack of verification/undo</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Confused about start/stop/restart buttons</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Confused about state of the start/stop/restart buttons</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The Process</td>
<td>Scan-Scan-(Info, Admin, Stop) repetitive fatigue</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Trouble recalling server name after comparison-based searches</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 4: User analysis for LARC.

The LARC developers chose to implement a “Wizard of Oz” prototype. The user used an identification card to log in, then moved the RFID reader across a fake server rack with transponders. Then, each user performed a series of tasks, watching the computer screen for responses, and telling one of the moderators which buttons he pushed. The moderator performed the actions on the computer, which allowed the user to see the response. This method of prototyping was chosen to allow the user to understand how the device feels when operated in its intended environment.

The success of this user testing was primarily due to the graduate student’s preparation. Not only did they create a background program to record automated timings of events (much like Mini-Mote in Fig 3), the graduate student also was careful to examine the device before the experiment, and make predictions about potential user errors. The grad student had prepared observational cues for the group members to look for. This guided their understanding to find the real problems users faced (Fig. 4).

After seeing the results, especially about repeating the “lock” step multiple times, the group members recognized that they could improve their device. One of the comments made at the end of the course was the desire to improve their design and test again. A similar sentiment was shared by other groups as well, including the MP3 Stingray project.

3.3 MP3 Stingray

A third group decided to create an interface for an MP3 player, called the MP3 Stingray. As a common choice for design projects, groups that choose MP3 players are asked to attempt something novel with their device. To meet this requirement, they created a randomized playlist generator which selected from sublists based on genre, artist, album, and existing customized playlists. The generator allows the user to create a mix timeline where the device chooses randomly from the sublists associated for that time.

Although their task seemed to be very straightforward, the small size of the TDA and the inexperience of the group members meant that they were required to get past a number of usability issues. During the Heuristic Evaluations, they were asked to make adjustments based on almost every heuristic, and though they made significant improvements between their low-fidelity prototype and their high-fidelity prototype, the User Testing revealed additional problems with their device.

During their observations, they noticed that each user took significantly longer to complete the tasks than expected. Although users were quicker the second time (Fig. 5), evaluations based on the graduate student and group members’ observations indicated that certain tasks were not intuitive. The users had simply memorized the steps to complete those tasks, instead of basing their actions on the ambiguous visual cues from the device.

Despite all of MP3 Stingray’s challenges, the group members seemed more enlightened at the end of the course; during the exit interview, all of the group members were very intense in their praise of the grads’ feedback and the perspective gained from user testing. We believe that experiences like this will inspire students to practice usability evaluation during the design process in the future.
4. EVALUATION AND CONCLUSIONS

A combined grad and undergrad HCI course benefited both courses and their respective students. Although students did not perform the course work for both courses, they were exposed to the results of both courses and interacted with students in the other course. Each grad student participated in the development of several UIs and had the opportunity to conduct usability testing on a unique UI. Each undergrad group received feedback on their UI design from several grad students, then helped with the usability testing of their UI. Conducting the combined course in such a manner allowed the undergrad and grad students to interact with each other, exposing the undergrad students to graduate school life. We hope that this exposure may encourage more undergrad students to continue their education on to graduate school.

Independent exit interviews were conducted with each grad student and with each undergrad group. Almost every group talked enthusiastically about the usability testing, the graduate students’ evaluations and what they learned from those. When asked about the grads, one group provided a very concise description of the overall feel of all the groups.

“They [the grads] had good ideas and helped make our project a little better. They were sort of like the customer: they had a unique perspective that we maybe wouldn’t have come up with ourselves.”

Many of the grad students said that the course helped them better understand the evaluation process, since they were practicing on developing projects, instead of completed devices. They also experienced many of the pitfalls that a released product could not provide, including: delays, incomplete prototypes, and uncooperative developers. Despite these obstacles, all seven of the grads produced detailed, interesting documentation regarding the analysis of their group projects. Even the two graduate students who had taken the undergraduate course in previous semesters commented that they benefited from taking the graduate course.

The other aspect of the course that the group members discussed was the grads’ research presentations. Many undergrads found the research presentations interesting, and commented on them during the exit interviews. One student said that he “didn’t even know that there was such a thing as Tangible User Interfaces” until he heard the grad’s presentation.

The experimental evidence indicates that this course imparted deeper understanding to the students. The response was well above our expectations. Almost every student, including those grad students who had previously taken the undergrad course, was able to respond with valuable lessons learned during the exit interviews. Therefore, we were successful in meeting our standards: two truly distinct, yet combined courses. We believe that this is largely due to the interactions between the groups and the grad students, and the usability testing on high-fidelity prototypes. These two features, student interaction and high-fidelity user testing, were possible exclusively through the use of a combined course.

5. ACKNOWLEDGMENTS

We would like to thank the Computer Science Department at Michigan Technological University, and especially Chair Linda Ott, for sponsoring this course. We would also like to thank the students of both courses for allowing us to study and publish their work. Finally, we thank the volunteers who took the time to participate in the user testing, permitting the primary focus of the course to be a success.

6. REFERENCES