Teaching HCI Design With the Studio Approach

Yolanda Jacobs Reimer and Sarah A. Douglas*  

1Department of Computer Science, University of Montana, Missoula, MT, USA, and 2Department of Computer and Information Science, University of Oregon, Eugene, OR, USA

ABSTRACT

The studio-based method of teaching has been used for almost 100 years to teach product and architecture design. With ever increasing pressure on HCI to teach competence in designing interactive objects, new ways of teaching need to be explored. This article begins with a review of the studio-based teaching concept and how it has been used in architecture, science/engineering, and computer science education. We then present and discuss the evaluation of an HCI design studio course which we created and taught in spring 2002 in the Computer and Information Science Department of the University of Oregon. This course was based on our observations and study of studio courses in the School of Architecture. Finally, we review general issues about studio teaching including the promises and challenges that it presents to widespread acceptance in the computer science curriculum.

INTRODUCTION

During the past 20 years the goal of a computer science HCI curriculum has changed from primarily teaching the psychological foundations and surveys of human-computer interaction technologies to teaching competence in designing interactive objects. This is no doubt due to the explosion in the 1990s of interactive technologies such as Web user interfaces and mobile communication networks, and the demand to create successful products through improved usability. Although much of the response to this has been
met by newly created departments of information technology or multimedia design, computer science departments have felt the impact.

For HCI educators the difficulty is that we come ill-prepared to teach design itself, much less the real-world replication of it. Traditionally, HCI design has been taught as an abstract process of iterative user-centered design with a recommended set of design aids such as task analysis, GOMS, guidelines, heuristic evaluation and usability testing. During a typical HCI course there might be an occasional practice exercise for the student to evaluate and even improve, for example, an interface with poor usability. Adding a team-based final project to design, implement and evaluate a working interface provides more experience in actual HCI design, but still falls short of teaching the student good design of a real-world artifact while engaging in a real-world design process. In these courses we leave it to faith that students will be able to make the transition from theory to practice. The problem is that we have been trying to teach design without getting our hands dirty.

What can we do to improve the teaching of HCI design? As early as 1990, Professor Terry Winograd of Stanford University in a CHI conference plenary entitled “What can we teach about Human-Computer Interaction” urged the audience to embrace HCI design education and explore the studio as a teaching concept (Winograd, 1990).

Frustrated by the inability to teach HCI design well over the past 20 years in a standard HCI course, the second author (Douglas), who is a professor in the Computer and Information Science Department of the University of Oregon, was inspired by Winograd's ideas. She obtained funding from the Williams Fund for Teaching Innovation to apply studio-based concepts from the University’s School of Architecture to a new HCI design course. This new course is a junior-level HCI design studio for computer science majors, and computer information technology or multimedia minors. It complements an existing senior and graduate-level standard HCI course taught to senior and graduate computer science majors.

This article begins with a review of the studio-based teaching concept and how it has been used in architecture, science/engineering, and computer science education. We then present and discuss an HCI design studio course which we created and taught in spring 2002. Finally, we review general issues about studio teaching including the promises and challenges that it presents to widespread acceptance in the computer science curriculum.
THE STUDIO CONCEPT

The naturalness and success of the studio teaching method recommends it to teaching HCI design. It is commonly found today in an evolved form as the primary pedagogy in both product design and architecture schools (Schon, 1987). To more fully understand this model, we spent approximately a full term observing its use in the School of Architecture at the University of Oregon. Our intention was to understand how these classes are organized, how students learn design, and how students benefit from the intense interaction between themselves and their instructors.

Studio Classes in the School of Architecture

Scheduling and Format
Each term, in addition to regular lecture courses, architecture students are required to enroll in a studio class tailored to their skill level in the program. Studio classes are held 3 days a week, 4 hr per day. Each studio class takes a real-world architectural problem and requires students to produce a final building design through an iterative design process. Problems vary by complexity of function, complexity of environmental conditions, or other aspects depending on the skill level of the students. Each week the instructor will emphasize the design of a particular architectural aspect, such as form, site location, function, and so forth, of the overall project. Depending on the studio class, students often work in collaborative teams to produce a joint design.

Studio classes are held in specially designed rooms and the size of the class is limited to 10–12 students. Students are assigned large drafting-style desks where they work on their design projects both during class and outside classroom time. This is the student’s private workspace for the duration of the term. In addition to the desks, an open area with table, chairs and pin-up wall space is maintained for group conferences.

All of the studio classes are highly interactive (faculty to student; student to student) and involve regular design critique – or crit – sessions. The format of a typical 3-hr studio session is as follows: students work at their desks informally (i.e., they are free to talk with one another, play music quietly, etc.) on their current design project until it is time for them to meet with their instructor for a crit session.
Design Crits

The design crit is the central means of conveying design knowledge. Instructors usually gather from 2 to 4 students together at one time. Each student either brings his or her drawings over to the common meeting area or pins them up on the wall for review. Design representations are often low-fidelity sketches to promote the general communication of ideas and to enable students to throw away bad designs. While the instructor focuses on the work of one individual at a time – taking between 20 and 30 min – the remaining students benefit from the comments made by the faculty member and student.

Design crits start with the student explaining how he or she is meeting the particular design emphasis for the week. To keep the critiques positive, reviewers generally begin their comments with statements like “I like what you’ve done with...”. Many reviewers then use the Socratic method to ask the student a number of strategic questions which serve to highlight perceived weaknesses with the design. Reviewers often end their critique by suggesting similar problems/solutions done by well-known architects, and by asking the student if he or she has any specific problems and/or questions they wish to ask. Finally, faculty reviewers will also make helpful suggestions on the student’s presentation itself (e.g., urging the student to frame the problem and to discuss his or her goals overall before getting into details). This provides the student with direction for future success, both in the current class and elsewhere.

In addition to the format of the design crit session described above, there are midterm and final “pin-up” sessions. Midterm pin-up sessions are slightly more formal than the day-to-day crit sessions in that they typically signify important milestones in the project design process, and they are analogous to midterm exams. During a pin-up session, each student’s work is reviewed separately by two Architecture school faculty members and by a peer. Each review session lasts approximately one-half hour. While a student is having his or her work reviewed, other students in the class are free to sit in and listen and learn. The final session of the term is a formal design crit by invited professional architects. Studio classes are graded on a Pass/Fail scale, so as to encourage collaboration among students.

To summarize, the studio model we observed has the following characteristics:

- It involves experiential, immersive pedagogy: learning by doing.
- Prior knowledge acquired from standard lecture courses is integrated.
- Students produce realistic artifacts using realistic design processes taken from professional practice.
- Students are active learners with teachers as resources or coaches.
- Collaboration is a key process between teacher and student, and student peers.
- Student assessment is based on presentations of design artifacts.
- Learning environment between students and teachers, students and students, and even students and professionals is intense and highly interactive.
- Primary teaching is through the design crit.
- Communication of the design and reflection upon both the product and process is critical to learning.
- It involves specialized studio rooms and scheduling.

As the reader can see, studio teaching is radically different from the usual computer science instruction of lecture/lab/discussion. Studios can be characterized as a form of social constructivism applied to education (Lebow, 1993). The philosophy of social constructivism emphasizes context and the integration of thought and action. It turns the usual instructional model around claiming that “what the student does is actually more important in determining what is learned than what the teacher does” (Shuell, 1986). Rather than students being passive receptacles of transmitted knowledge, students must do the most work! This approach to education is particularly effective when learning is open-ended, focusing on complex contextual problem solving that demands a high degree of analysis, idea generation, reflection and communication. This is exactly the situation that we encounter when teaching design (Petroski, 1996).

**Studios in HCI Education**

Turning now to the use of studios in HCI education, Winograd has developed a course at Stanford, *HCI Design Studio* CS 247A, and a text of readings (Winograd, 2002; Winograd, Bennett, De Young, Gordon, & Hartfield, 1996). Winograd recommends that CS 247A, which is a stand-alone class, be preceded by CS 147, *Introduction to HCI*. Although details are not available, his curriculum follows the spirit of the studio approach that we observed in architecture with one exception: 40% of the final grade is based on a journal that students keep of their design process.

We found three degree programs in information technology based on the studio approach. Two began in the late 1990s and are in Australia at the CSEE
School, University of Queensland (Docherty, Sutton, Brereton, & Kaplan, 2001) and at the School of Information Management and Systems, Monash University (Carbone & Sheard, 2002; Gonsalvez & Atchison, 2000). Both the University of Queensland and the Monash University programs require students to enroll every semester in a project-based studio course in which open-ended design problems are tackled. This is exactly the same format as we observed in architecture. In both universities the student evaluations are quite positive toward the studio approach. A third degree program in information technology was developed at Rensselaer Polytechnic Institute (RPI) in 1998 (Spooner, 2000).

This program was based on the description of the RPI Studio Model for science and engineering. Originally developed at RPI in the early 1990s under the leadership of Professor Jack Wilson for teaching physics, it has since become widespread within RPI in other science and engineering curricula (Wilson, 2002). In many ways, the RPI use of the word studio designates the integration of lecture, lab and discussion into one scheduled time period of the class held in a single room. It also de-emphasizes lectures and emphasizes hands-on team activities. Although similar in pedagogical intent to studio teaching, in our opinion the RPI Studio Model does not parallel closely the use of the studio for design courses. The main difference is that the studio approach supplements regular lecture and discussion courses; it does not replace or condense them. Secondly, the studio course is totally oriented toward practice and usually requires an increase in scheduled time.

**A STUDIO COURSE FOR HCI DESIGN**

Applying our research with the School of Architecture, we implemented a new 4-credit undergraduate course in HCI design based around the studio concept and the integration of weekly design crits. Offered through the CIS Department, this course was intended for computer science majors who had not taken the more advanced HCI course, and for advanced students in the Computer Information Technology minor and the Multimedia minor. This course, called *Designing Software for People*, was offered during a 10-week trimester in the Spring of 2002. The remainder of this paper describes our experiences with this class, the effectiveness of the studio format and integration of weekly design crits, and, ultimately, whether we were able to satisfy our initial goal to improve the teaching of HCI design.
Course Content
Given the time limitations of a 10-week trimester, we opted to concentrate the course content primarily on principles of good design and the key components of the user-centered design (UCD) process (including requirements specification, preliminary system design, heuristic evaluation, prototype implementation, and usability testing). Each design activity—which was evaluated during weekly crit sessions—complemented the lecture material and focused on one or two critical areas of design. After reviewing some of the literature on group work recommended to us by faculty in the Architecture School (e.g., Michaelsen, Fink, & Knight, 1997), we decided to make all design work group-based. Each week students collaborated in groups of 3–4 on design activities.

For final grading purposes, students were assessed based on their weekly group design activities, their individual pop quiz performance, their class participation (particularly during the design crits), and their performance evaluations from fellow group members. (Twice throughout the term, students were given the opportunity to complete a group evaluation form. These forms asked the student to assess his or her own contribution to the group, as well as the contribution offered by the other group members.)

Format
Our class met twice weekly, once for 50 min on Monday and then again on Friday for 110 min. The scheduled class meeting times were used for a lecture component and for weekly design crits, so students were expected to complete all of their group design work out-of-class throughout the week. While most studio classes rely on previous course content and thus do not have a corresponding lecture component, we used Monday meetings to introduce new design concepts and to discuss the weekly design activity (see Table 1). During Friday’s meeting time, the entire class participated in critiquing each group’s solution to the weekly design problem. For these crits, each group prepared a 15-min presentation followed by 10 min of questions and comments from the class and from the instructors. Groups were asked to illustrate their designs by creating low-fidelity sketches of their system, to include physical props or models if appropriate, to walk through common tasks that users would perform using their interface, and to include sample data in the design display to facilitate understanding. Students were encouraged—and in some cases required—to use the classroom facilities (computer hooked up to an overhead projector, Internet connection, VCR) for
<table>
<thead>
<tr>
<th>Week of</th>
<th>Design topic</th>
<th>Activity</th>
<th>Activity focus</th>
<th>Text reading</th>
</tr>
</thead>
<tbody>
<tr>
<td>April 1</td>
<td>Intro; usability</td>
<td>Rapid transit system</td>
<td>Learning; physical design</td>
<td>Ch. 1</td>
</tr>
<tr>
<td>April 8</td>
<td>Design process</td>
<td>Automated sprinkler system</td>
<td>Time critical system; spatial mapping</td>
<td>Ch. 6, 9</td>
</tr>
<tr>
<td>April 15</td>
<td>Functional requirements</td>
<td>Supercard</td>
<td>Numerous functions; privacy, security</td>
<td>Ch. 7</td>
</tr>
<tr>
<td>definition</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>April 22</td>
<td>User requirements</td>
<td>Election information</td>
<td>Design of large information space</td>
<td>Ch. 7, 9</td>
</tr>
<tr>
<td>definition</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>April 29</td>
<td>Task analysis</td>
<td>Election website – system</td>
<td>Requirements specification</td>
<td>Ch. 7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>requirements</td>
<td></td>
<td></td>
</tr>
<tr>
<td>May 6</td>
<td>Prototyping methods</td>
<td>Election website – initial</td>
<td>Storyboards; scenarios of use</td>
<td>Ch. 8</td>
</tr>
<tr>
<td></td>
<td></td>
<td>system design; scenarios of</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>use</td>
<td></td>
<td></td>
</tr>
<tr>
<td>May 13</td>
<td>Heuristic evaluation/</td>
<td>Election website – user</td>
<td>Eliciting user feedback</td>
<td>Ch. 10, 13</td>
</tr>
<tr>
<td></td>
<td>cognitive walkthrough</td>
<td>feedback</td>
<td></td>
<td></td>
</tr>
<tr>
<td>May 20</td>
<td>Usability testing</td>
<td>Election website – system</td>
<td>Implementing system design</td>
<td>Ch. 14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>implementation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>May 27</td>
<td>(Monday holiday)</td>
<td>Election website – system</td>
<td>Implementing system design; preparing for</td>
<td>Ch. 10, 14</td>
</tr>
<tr>
<td></td>
<td></td>
<td>implementation</td>
<td>usability testing</td>
<td></td>
</tr>
<tr>
<td>June 3</td>
<td>Review</td>
<td>Election website – usability</td>
<td>Conducting usability tests</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>testing</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Note. a Required text used was Preece, Rogers, and Sharp (2002).*
their presentations. Additionally, all groups were asked to create a smaller, more manageable version of their design to hand in to the instructors at the end of class.

**Project Descriptions**

There were a total of four projects for this course with 10 weekly presentations required. The first three projects were 1-week "warm-up" problems, each focusing on a different UCD issue. The fourth project was 7-week long.

- **Project 1:** Design of a rapid transit system kiosk to dispense subway tokens. Design issues: (a) how learning affects design and (b) human physical constraints;
- **Project 2:** Design of an automated sprinkler system that operated based on a number of environmental and user-defined parameters. Design issues: (a) handling time critical systems and (b) spatial layout in design;
- **Project 3:** Design of a digital Supercard to be used in place of standard credit cards, driver's license, ATM cards, wallet photographs, and so forth. Design issues: (a) multiple functions and (b) privacy and security issues;
- **Project 4:** Design, prototype, and test a Web-based voting system. Incorporated large-scale information display, requirements specification, preliminary system design, scenarios of use, user feedback, prototype implementation, and usability testing.

Although most students lacked prior design experience, the work submitted for the first three projects was impressive. Students were able to apply design heuristics discussed in lecture, and they learned quickly from the feedback they received during crit sessions. These early successes reinforce our belief that students can be productive and efficient even when immediately immersed in design work; in fact, this approach gives students a good baseline for their own work, and they can often see clear improvements in their designs as the term progresses.

Project 4 focused on the creation of an election website for the state of Oregon for the year 2000. Each week a specialized project activity was assigned, as seen in Table 1. For example, the activity for week number 4 (which was the first in the series of seven activities) involved the design of the overall information space for the website. Students were provided with a hardcopy of the voters' pamphlet which contained all of the requisite data in a flat, static information space. The following week's activity involved the identification of the system requirements for the website. All project activities
culminated in the implementation of a prototype website, which students then used to conduct usability testing with potential users. These prototype implementations can be viewed along with other detailed materials from the course at http://www.cs.uoregon.edu/classes/cis399dsp/.

Results and Observations
We believe that our students benefited enormously from being exposed to a real studio-based experience. In particular, students were immediately immersed in the process of design, they learned by creating realistic artifacts as opposed to listening passively to lectures, they engaged in intense collaboration amongst themselves and their instructors, and they integrated weekly lecture topics into their designs. For these reasons, we think that our primary goal of developing a more effective way of teaching design using the studio concept was ultimately met. We believe our experience using the studio approach with less-knowledgeable students would be equally beneficial to a more skilled student population. More skilled students could be expected to take on more complicated design problems, and they would be held to a higher standard for their finished work. The remainder of this section details some of the benefits of the studio approach based on student evaluations and our observations.

Student evaluations are required for all University of Oregon courses. They consist of a set of 23 standard questions evaluating both the course and the instructor. Students respond to each question as if they are grading the class by choosing one of six possible ranked responses corresponding to a scale from 0 to 10 points in two point increments. For example, response "A" is "exceptionally good" and corresponds to 10 points. The responses for each course are compared through a z statistic with other courses taught in the computer science department during the same term.

Eighty-six percent of the class participated in the evaluation. We focus our analysis of the evaluation primarily on the course-related questions, which are summarized in Table 2. For each question, the overall rating of the class is shown (mean score out of a range from 10.0 to 0.0), along with the computer science departmental mean for the same question and the subsequent z statistic. For example, the class mean for the first question, "In comparison with other UO courses this size and level, how do you evaluate this course?" was 8.0 and the computer science departmental mean for this question was 7.6; this indicates that the students in our course rated it better than students in other courses answering the same question, and is reflected by a z statistic of
Table 2. Student Evaluation Results.

<table>
<thead>
<tr>
<th>Question</th>
<th>Studio class mean</th>
<th>CS dept. mean</th>
<th>z statistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. In comparison with other UO courses this size and level, how do you evaluate this course?</td>
<td>8.0</td>
<td>7.6</td>
<td>0.2</td>
</tr>
<tr>
<td>2. Did you learn something from the class that you consider valuable?</td>
<td>9.1</td>
<td>8.5</td>
<td>0.2</td>
</tr>
<tr>
<td>3. Has your interest in the subject increased as a result of taking this course?</td>
<td>8.5</td>
<td>7.7</td>
<td>0.3</td>
</tr>
<tr>
<td>4. How difficult was the course?</td>
<td>7.5</td>
<td>6.1</td>
<td>0.7</td>
</tr>
<tr>
<td>5. How was the workload of the course?</td>
<td>9.4</td>
<td>6.2</td>
<td>1.6</td>
</tr>
<tr>
<td>6. Did you have a clear sense of the goals and direction of the course?</td>
<td>5.6</td>
<td>7.6</td>
<td>-0.8</td>
</tr>
<tr>
<td>7. Were the course requirements and grading procedures made clear?</td>
<td>6.4</td>
<td>7.8</td>
<td>-0.6</td>
</tr>
</tbody>
</table>

0.2. As seen in Table 2, questions 2 and 3 also reveal important student satisfaction; these are very positive outcomes and ones that we sought in creating this studio course.

Regarding the aspects of the course that the students did not like, it is clear from questions 4 to 7 that students found this course difficult and with a heavy workload, and that they were somewhat confused by the unfamiliar studio approach to teaching and grading. It is important to note that despite these negative aspects, students still rated this course as a very positive overall learning experience. Although it would be easy to dismiss their criticisms as just start-up problems, it does highlight a major difficulty in introducing an entirely new pedagogy into the curriculum. In the future, we will address both the workload and grading clarification issues.

In an informal discussion regarding the format of the class, students said that they particularly appreciated working with a variety of different group members, each of whom contributed unique ideas to design problems; they valued witnessing how other groups approached the same design problems; and they felt that the class helped them develop new perspectives when evaluating everyday objects in the world. As instructors, we agree with the students about the pedagogical value and effectiveness of the design crit sessions. Allowing students direct involvement in critiquing their peers’ designs fostered a highly interactive and constructive learning atmosphere; students willingly helped and offered opinions to their classmates, and they benefited indirectly from the instructors’ design expertise as well.
Furthermore, since each weekly design crit session involved a group presentation, students naturally took ownership and pride in their solutions and late work was never an issue. Finally, in a time when communication skills are becoming increasingly important throughout the sciences, students gained considerable experience with public speaking and developing their writing skills (by submitting a report of their design solutions each week).

Our selection of weekly design activities also turned out very well. The first 3 projects were enjoyable, set the tone for the class (i.e., students were immediately immersed in the design process, the out-of-class time commitments were directly evident, etc.), and were straightforward for all students regardless of their design experience. The larger project, which spanned 7 weeks, exposed students to key aspects of user-centered design – requirements specification, preliminary system design, early user feedback, prototype implementation, and usability testing – and gave students the opportunity to continually improve their designs rather than simply creating “throw-away” low-fidelity prototypes. The larger project chosen also enabled all students to implement a functional prototype despite their technical background. Since the development of a working prototype was critical for usability testing, one of our main goals when planning this course was to devise a project that non-CS students could implement. We discovered that most students, regardless of their technical experience, are proficient in either HTML and Javascript, or in commercially available software used for creating web sites (e.g., Macromedia Dreamweaver). By accepting prototypes that only had to appear fully functional, students could easily create non-functional web pages, and little programming expertise beyond basic HTML links and the inclusion of images was necessary.

Continuing Challenges
Compared to implementing a true studio class, we were not able to meet as often or as long as a full studio schedule demands due to traditional lecture/lab scheduling. Overcoming this challenge will be difficult since it affects all scheduled CS courses. Embracing the schedule required by studio courses means more time spent for students and faculty in the classroom. Because we were unable to provide a true studio schedule, the most pressing problem raised by our students was the large amount of out-of-class group work that was required on a weekly basis and the difficulty for students to find times during which they could meet as a group each week.
Traditional studio classes do not have a lecture component; they are a complement to lecture classes. We incorporated a small lecture component in the class because students had not had the content in other courses. True studio classes should be fully integrated into the overall computer science curriculum. Mixing the lecture with the studio was a mistake because students did not appear to be keeping up with the assigned text readings. They were actually overwhelmed with the design projects. In the future, we recommend trying to offer the class as a true studio course without a lecture component. With this format, instructors might also consider offering the class as Pass/Fail, as is usually done in traditional studio courses, in order to encourage collaboration.

We were unable to secure a real studio for a classroom. This meant that students had no individual desks where they could work and leave their projects, and no place to meet outside of class. During the actual meeting of the studio, the room assigned to us could not be arranged for small group design crits since there was no conference table. We believe that successful implementation of the studio concept requires special studio rooms.

A final concern voiced by the students proves more challenging to handle: How far should one go in implementation? It is clear in architecture that students will not build their designs. However, in computer science it is very difficult to even evaluate the design without some implementation or working prototype. The decision of how far to go in implementation will depend on each teaching situation, but the focus should always be on design, not implementation.

CONCLUSION

Upon reflection, we are quite pleased with our first time offering of Desining Software for People and of the studio approach to teaching design. We were able to incorporate many of the critical aspects of a traditional studio class such as weekly design problems, collaboration between students and faculty, production of realistic artifacts in a realistic process, and weekly design crit sessions. We were not able to provide specialized studio rooms and real studio scheduling; doing so would be a major improvement. Both students and faculty responded very positively to the learning-by-doing approach.

HCI may be the first part of the CS curriculum to be affected by demands to teach competence in design, but it may not be the last. For example, the
latest version of the recommended ACM/IEEE Computing Curriculum for Computer Science puts a heavy emphasis on software engineering and design for current technology (ACM/IEEE Computing Curriculum, 2001). For example, the required undergraduate core includes 31 hr on software engineering and 15 hr on net-centric computing. Traditional operating systems and programming languages are only 18 and 21 hr, respectively. As such, our experiences may be the bellwether of change. An experimental studio course in HCI offers needed insights into conversion of other computer science courses to a more design-centered approach.

ACKNOWLEDGMENTS

We would like to acknowledge the generosity of the Tom and Carol Williams Fund for Undergraduate Education at the University of Oregon. Without their financial and motivational support, course development would not have been possible. Secondly, we could not have been able to grasp many aspects of studio courses without the help of the following members of our School of Architecture faculty: G.Z. Brown, Nancy Cheng, Donald Corner, Ronald Kellet, and Christine Theodoropoulos. They embraced our ideas with enthusiasm, answered our questions with patience, and allowed us unlimited observations of their classes.

REFERENCES


