Lessons from Teaching HCI for a Diverse Student Population

Bowen Hui
Department of Computer Science, University of British Columbia, Okanagan, Canada

ABSTRACT

HCI education has been an active area in the past two decades. Many researchers have reported on challenges unique to teaching HCI and investigated different pedagogical approaches to improving the students' learning experience. We add to this body of research by describing our experience in teaching HCI as a mandatory undergraduate course, but with the added complexity of having non-majors and graduate students in it. We describe three pedagogical approaches to teaching this course and summarize the lessons learned. We outline our next steps and hope our discussion can help shape future HCI course developments.

CCS CONCEPTS

• **Social and professional topics** → Professional topics; Computing education; Computing education programs; Computer science education; • **Human-centered computing** → Human computer interaction (HCI).

KEYWORDS

HCI education, community service learning, student choice, studio approach, large classes

ACM Reference Format:
Bowen Hui. 2020. Lessons from Teaching HCI for a Diverse Student Population. In *Koli Calling '20: Proceedings of the 20th Koli Calling International Conference on Computing Education Research (Koli Calling '20), November 19–22, 2020, Koli, Finland, Finland*. ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3428029.3428054

1 INTRODUCTION

Due to the important role that human-computer interaction (HCI) plays in computer science careers as well as technological advances in our society, our department made the HCI course mandatory in the undergraduate computer science program. Similar to other experiences encountered by other HCI educators, we found that students came to the course with negative preconceptions about HCI and thought that the content was too easy, the grading was too subjective, and the difficulty level to be not challenging enough (e.g., [1, 8, 18]).

Different opinions and pedagogical strategies have been reported to counteract these problems. Suggestions and case studies include strategies such as involving real user needs [16], changing the culture of individualized summative assessments common in computer science [2], focusing more on the design process rather than design outcomes [8], creating a platform to enable students to experience and explore a design space [17], allowing students to work on projects based on students' own interests [27], and rebranding the HCI discipline globally [15].

The challenges we face in our HCI course is exacerbated by the diversity of academic backgrounds in our student population. Historically, this course was created as an elective course for our computer science majors. As the focus of it weighted more heavily on the design side, the course did not require any programming prerequisites so that non-majors could successfully complete the course also. This curriculum decision fostered interdisciplinary enrollment and team work in the course. Once the course became a core requirement in the computer science program, the obstacles in keeping students engaged became harder to overcome. In a given year, about 80-90% of the students are computer science majors while the rest consist of students majoring in other disciplines, such as mathematics, management, and psychology. Since this is the single HCI course offered by our department, it is also cross-listed as a graduate-level course (although graduate students are evaluated somewhat differently). Thus, balancing theoretical and practical content and teaching at a pace that is suitable for each student subgroup are key challenges.

To better understand the active research in the recent decades, we first provide an overview of the relevant literature in Section 2. We describe our experience in teaching the undergraduate HCI course in Section 3. Here, we describe past approaches used to improve student's learning experiences: a community service learning approach when class sizes were small, a dual-track approach that gave students choice in the type of assessment they worked on, and a studio-like approach to foster more design thinking activities. Based on these experiences, Section 4 reflects on the positive outcomes and identifies the challenges that remain, while Section 5 outlines future improvements needed for HCI course development.

2 RELATED WORK

Many efforts in HCI education has highlighted challenges unique to teaching HCI in comparison to traditional areas of computer science. A major challenge is the fact that technology changes so rapidly, which inevitably leads to an increased growth in user populations and diversified user needs [4]. Consequently, many scholars have identified what we need to teach in HCI as a moving target. As such, new design methodologies needed to account for the changes in new technologies are needed but are often not yet readily available for teaching purposes. As a result, researchers proposed that the pedagogical focus should be placed on helping students develop skills and competencies rather than content knowledge [8, 20, 26].

In 1992, significant efforts were made by Hewitt et al. in creating a curriculum blueprint to guide the development of HCI courses [10]. Part of this report provides an inventory of the state-of-the-art content areas in HCI at that time. The authors further propose four types of HCI courses with different content emphasis that could be offered as courses in computer science, psychology, and management information systems. In particular, this proposal includes two computer science courses – the first would be a practical course that focuses on user interface design and development while a follow-up course would focus on the theoretical aspects of HCI. Although this report provided a set of recommendations for teaching HCI, content and method renewal are needed in order to keep up with the changing technologies and evolving student populations.

Despite the fact that design thinking plays a key role in building usable systems, educators have found this subject matter to be generally poorly received by HCI students housed in technical disciplines. A major contributing factor is that design thinking has traditionally followed an experiential paradigm that enables a student to become an expert by repeatedly doing design thinking projects, receiving formative feedback, and conducting design critiques [16, 23, 25]. In contrast, the early years in a computer science degree typically teach formal and technical content that have a right answer and adopt summative assessments. In order to teach design thinking effectively, researchers have proposed incorporating more experiential learning opportunities [16, 17], using different teaching methods such as studio pedagogy [23], changing the physical space in a classroom [2], and moving away from the individual summative assessment culture [2].

For these reasons, many classroom cases have reported that students perceive HCI content to be "too easy", that it is "all common sense", or "too fuzzy" [1, 8]. The overall negative perception of HCI has led to the emergence of various studies to improve students' learning experience. Suggestions include focusing on the more technical aspects of HCI (such as tools and techniques) and making the content more relevant by applying design concepts to real applications by involving real users or an external client [1, 17].

There has been much interest in shaping HCI education in the recent years. Churchill et al. [4] reported on the global perspective of HCI education by surveying HCI professionals from over 30 countries. The paper reported on common topics and methods used in various geographic regions, and showed that there is a lack of consensus on what and how to teach HCI globally. Inspired by the ongoing changes in the field, the authors proposed that a group of "progressionals" (researchers, educators, and practitioners) be formed to maintain the core values, tenants, and perspectives unique to HCI. Subsequently, Churchill et al. [5] held a workshop to discuss the development of a "living HCI curriculum". The idea of a living curriculum is for HCI educators to embrace change so that the curriculum would evolve alongside the changes observed in the field. The vision of this proposal would offer a flexible, global, and frequently refreshed curriculum. Major challenges in this initiative include defining the co-design process in developing such a curriculum and the ongoing maintenance of the material. Nonetheless, vibrant discussions began to pursue this goal.

Efforts in understanding pedagogical trends began to appear, most noticably in North America [14], Brazil [3] and Asia-Pacific [7, 19, 24]. Forums, special interest groups, and workshops to better understand teaching methods, university teaching programs, and HCI pedagogy internationally were held as well [9, 11, 12]. Churchill et al. [6] reported on an extensive project that interviewed 52 SIGCHI community members and surveyed 872 educators worldwide. Along with the many resources compiled, one outcome from this study includes a list of topics that survey respondents considered to be "very important" or "important" (on a 5-point Likert scale). This list of topics could serve as an initial attempt to develop a core set of topics for a global curriculum. In the last two years, several workshops and symposiums focused on different aspects of realizing the living curriculum, such as prototyping it [13], incorporating curricular needs at an international level [18, 21], and developing its information architecture needs [22].

3 OUR EXPERIENCE IN TEACHING HCI

Here, we describe our experience teaching an undergraduate HCI course and the unique constraints we face. Due to limited resources, our department offers a single third-year HCI course. This course is taught by a computer science faculty member and does not require any programming prerequisites. The course was designed to provide students with an overview of the area as well as to probe students into pursuing HCI in subsequent project courses. In contrast to the recommendation of having two undergraduate HCI courses focusing on practical and theoretical content separately [10], our HCI course combined both aspects together.

While most of the students are computer science majors, this course has traditionally attracted students from a variety of disciplines, including management, psychology, and mathematics. In a given year, approximately 10-20% of the students do not major in computer science and have no programming experience. Due to the important role that HCI plays in career development as well as technological advances in our society, this course has recently become a requirement for all majors in the undergraduate computer science degree. At the same time, it became a prerequisite for a fourth-year capstone project course which students typically take in their final year. Depending on graduate enrolment, this course may also be cross-listed as a graduate-level HCI course. There are typically two major changes made to the graduate version of the course. First, workload would include additional readings and more difficult assessments. Second, graduate students would be required to complete an extra research project that would be worth a significant portion of the course.

This diversity in the student population makes it challenging to teach design and evaluation techniques using examples and language suitable to the students' background. Additionally, graduate students are arguably forced to learn at a slower pace because the curriculum is designed for third-year undergraduate students. The challenges we face in designing a suitable HCI course for our student population is comparable to the experience of offering HCI as a mandatory course reported by others [15]. Furthermore, our situation has added complexities because the course must additionally accommodate students with no programming background as well as graduate-level students.

3.1 A Community Service Learning Approach

In the earlier years when class size was small (less than 40 students), the HCI course had a term project involving the re-design of a website for a non-profit organization. Working collaboratively with community members to improve their website engaged students in a model of experiential pedagogy.

Students were divided into teams that maximized diversity in academic backgrounds and performance. Each team was assigned to a different organization and worked with a representative client throughout the semester. The lecture material was structured around the various milestones leading up to the final production of the website. These milestones include understanding the organizational context, gathering target user needs, developing alternative design mockups, implementing a functional website in WordPress, and conducting a usability evaluation of that site.

A major benefit is that students worked in an interdisciplinary team (where possible) to tackle a real world project. This structure gave students hands-on experience working with an external, non-technical client. Students were exposed to real user needs and learned to interact with end-users in a non-technical way. This allowed students to gain appreciation of user needs, which often resulted in a deeper exploration of the design space.

To run this course successfully, the instructional team had to overcome a series of administrative challenges. First, due to the HCI course being a core requirement in the program, the majority of students were computer science majors while only a minority were non-majors. Second, most students did not have any design experience prior to this course. Therefore, in practice, teams were formed by diversifying the following factors: gender, programming expertise, and overall academic performance. As a result, some teams were composed of computer science male students only.

In order for them to learn more about the organizational culture of the client company and target user needs, students were tasked with visiting the organizations to gather user information on-site. This created a logistics obstacle for students who had to commute each time they had to meet the client or one of the users to gather data.

A third challenge was managing and giving individualized feedback to each team. Since every team worked with a different client and, therefore, worked on a different project, the instructor and teaching assistants had to ensure that feedback was tailored accordingly. This made grading extremely time consuming.

Although course evaluation data is unavailable, students reported their feedback to the program advisor as having a generally negative experience. They reported that workload division was not fair among the team members because the non-majors often did not participate in building the website. They also complained about the time spent commuting for data collection. (Although we could have required the clients to meet students on campus instead, other problems with this pedagogical approach still remained.) The overall perception is that the students did not appreciate design and wanted to learn more programming skills. For these reasons, the community service learning model was abandoned in subsequent years where class sizes increased. Among the challenges associated with this pedagogical approach, the one issue that remains is the students' desire to learn more technical skills.

3.2 A Dual-Track Approach

In 2015, the HCI class had 76 students with approximately 15% of them being non-majors with no programming experience. The goal of this pedagogical approach is to meet the programming needs of the students in our majors as well as to accommodate for the students with no programming experience. As such, for each of the four individual assignments and final team project, we created two options – one option focused on solving design issues by developing programming solutions, while the other focused on solving design issues by creating paper or digital prototypes. These assessments addressed different topics of the course and gave students a hands-on perspective of the abstract concepts covered in the lectures. Lectures were taught following a traditional format. Although everyone had the same lectures and exams, students had a choice in the type of assessments they wanted to do. In fact, there was a small number of computer science majors who chose the non-programming options because they wanted to develop better design skills or because they felt the design option was easier.

A shortcoming of this dual-track approach is that students who chose to complete only design assignments failed to see how different programming environments impose added design restrictions. Likewise, students who only chose the programming assignments continued to neglect various design issues. Overall, the iterative software development process was never practiced. Another obstacle we encountered was creating two sets of assignments and projects that met the same learning outcomes and required a similar amount of work from students.

The main advantage of this course structure is that students had choice. Many students commented that they enjoyed having the choice of different assignments to work on while focusing on the same underlying concepts. A total of 26 students responded to the course evaluations. Among 52 comments made towards the strengths and the most enjoyable part of the course, 18 comments mentioned the applicability of the project, while 16 commented positively on the variety of the content and how it was interesting or relevant to real world applications.

This experience demonstrated that giving students choice helped meet the needs of students with diverse backgrounds. However, the administrative overhead in designing and grading the assessments indicates that a better process is needed to sustain this pedagogical approach. Moreover, the assessments need to be aligned in a way that enable students to practice HCI principles by doing both hands-on design and programming work.

3.3 A Mini-Studio Approach

Subsequently, the HCI course was modified to alleviate the administrative overhead in the dual-track approach so that all the students completed the same assessments. This time, we wanted to develop assessments that would allow students to practice the same HCI principles by completing both design and programming exercises. We will explain below how this is accomplished in our discussion of the alternate interfaces assignments. This HCI class was taught in 2017 with 77 students (about 10% of non-majors).

Separately, we created a new type of assessments called design challenges because students had commented positively on lectures having examples from real world applications. At the beginning of the semester, we elicited day-to-day challenges from students in the course. Examples include driving into a parking lot only to find that it was full or forgetting about the food going bad in the fridge. Next, we selected the most common problems that students identified and used them in these design challenges. Modeled after a studio learning environment, students worked on these design challenges in teams over the course of three to four classes. Although each team tackled the same topic in a design challenge, students often came up with very different solutions. A total of four design challenges were completed in the 13-week semester.

In addition, students were required to complete four programming assignments that applied certain lecture topics. We provided supporting tutorials and starter code to help students complete these assignments. Two programming assignments involved gesture recognition which were intended to be a continuation of the third design challenge that asked students to design alternative interfaces for regular household objects. Lastly, we had one evaluation activity where students conducted heuristic evaluation using one of the prototypes from a design challenge.

In order to provide more support for the design challenges, lecture content was condensed into mini-lectures with accompanying resources. This strategy was used to counteract the perception that lectures dwelled on content that was too easy. Lecture topics were organized so that students were asked to apply the concepts taught in class to the design challenges.

From an administrative perspective, this course flowed more smoothly and had positive outcomes for engaging students. However, the intended connection between the design challenge and the programming assignments on alternative interfaces was not made clear enough to the students. From the students' perspective, the additional design challenges, programming assignments, and evaluation activity seemed mostly disconnected. Recall that the application context used in design challenges were chosen based on student input. To foster creativity, design solutions were not restricted to any particular technology. On the other hand, we controlled for the level of difficulty of the programming assignments by standardizing the interaction techniques that needed to be implemented. As a result, the context used in the design challenges did not match what students had to program.

In total, 60 students responded to the course evaluations. Among 120 comments made towards the strengths and the most enjoyable part of the course, 29 commented that the course was interesting or relevant to real world applications, 39 commented on the design challenges or the design content, and 19 commented on the programming assignments. Many students indicated what they felt was obvious from the lectures often was not easy to solve in practice and were surprised to find there was so much to learn in the design process. This provides positive feedback for the mini-lectures and design challenges. At the same time, among the 60 comments on weaknesses of the course, 10 of them did not find value in the design challenges and 28 of them commented on the difficulty of the programming required or how the programming assignments were disconnected from the rest of the course.

The use of design challenges promoted an increase in the perceived value of design content in HCI. Despite making the programming assignments apply specific lecture concepts, a significant number of students still felt they were too abstract and disconnected from the rest of the course. Lastly, the connection made in the alternate interfaces exercises need to be made more explicit so that students see those design and programming activities as a coherent process.

4 LESSONS LEARNED

Each of our approaches had positive and negative outcomes. In this section, we highlight the lessons learned from our experience.

Lesson 1: Community service learning provided a wholesome perspective of an HCI process in a real world application context. Unfortunately, this approach is not scalable for large classes.

Lesson 2: Computer science majors expect to gain programming skills from an HCI course. When programming assignments were introduced, students found them to be too difficult or disconnected from the rest of the course. Thus, programming assignments need to be well integrated with other design and evaluation activities.

Lesson 3: Students were happy to be given a choice between doing a design assignment versus a programming assignment. However, providing two options for every assessment created too much overhead. As HCI material changes rapidly, care must be taken to develop a sustainable model as HCI content evolves and new material is added.

Lesson 4: Condensing lectures into a mini-lecture format and allocating class time to support design challenges enabled students to apply what they felt was obvious concepts in real world applications. Many students changed their opinions on the value of design and appreciated having a chance to apply those concepts.

Lesson 5: A separate project is needed to illustrate a complete software development lifecycle so that students can design, program, and evaluation their solution to a single problem. Unlike the community service learning approach, this project needs to be scalable to large classes.

5 SUMMARY AND FUTURE WORK

It has been a challenge designing an HCI course that serves the technical needs of computer science undergraduate students, while making the course accessible to non-majors and still engaging for graduate students. Overall, choice has garnered success in making the course more flexible to accommodate diverse student backgrounds. However, care must be taken to structure the course so to not create an unmanageable amount of overhead. We also saw that studio-like activities, such as design challenges, were well received by many students and effective in helping students meet the learning outcomes. Ideally, these activities can be carried through the implementation and evaluation phases.

Our next step is to develop a project that enables students to conduct design, programming, and evaluation activities for a single application context. Unlike the community service learning approach, this project needs to be scaled down so that it can be implemented in large classes and in conjunction with other successful ideas (such as design challenges) without overworking our students. We will also explore ways to incorporate choice into this project so to foster a flexible learning approach.

REFERENCES

[1] J., Aberg. 2010. Challenges with Teaching HCI Early to Computer Science. In Proceedings of the ACM 15th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE'10). ACM Inc., New York, NY, 3–7.

[2] E. Blevis, Y. Rogers, M. Siegel, W. Hazlewood, and A. Stephano. 2004. Integrating HCI and design: HCI/d at IUB, a design education case story. In Proceedings of the ACM CHI 2004 Workshop on the relationship between design and HCI. ACM Inc., New York, NY.

[3] C. Boscarioli, M.S. Silveira, R.O. Prates, S.A. Bim, and S.D. Barbosa. 2014. Charting the Landscape of HCI Education in Brazil. In M. Kurosu (ed.), Human-Computer Interaction. Theories, Methods, and Tools, Lecture Notes in Computer Science. Springer International Publishing, Cham, 177–186.

[4] E. Churchill, A. Bowser, and J. Preece. 2013. Teaching and Learning Human-computer Interaction: Past, Present, and Future. Interactions 20, 44–53.

[5] E. Churchill, A. Bowser, and J. Preece. 2014. Developing a Living HCI Curriculum to Support a Global Community. In Proceedings of the ACM CHI'14 Conference on Human Factors in Computing System. ACM Inc., New York, NY, 135-138.

[6] E. Churchill, A. Bowser, and J. Preece. 2016. The Future of HCI Education: A Flexible, Global, Living Curriculum. Interactions 23, 70–73.

[7] A. Dey, Y. Shi, F. Tian, and S. Cheng. 2015. Developing HCI Education Crossing Asia. In Proceedings of the ACM CHI'15 Conference on Human Factors in Computing Systems. ACM Inc., New York, NY, 2361–2364.

[8] A.D.N. Edwards, P. Wright, and H. Petrie. 2006. HCI education: We are failing – why. In Proceedings of HCI Educators Workshop'2006, 23–24.

[9] S. Grandhi. 2015. Educating Ourselves on HCI Education. Interactions, 22, 69–71.

[10] T.T. Hewett, R. Baecker, S. Card, T. Carey, J. Gasen, M. Mantei, G. Perlman, G. Strong, and W. Verplank. 1992. ACM SIGCHI Curricula for Human-Computer Interaction. ACM, New York, NY.

[11] K. Hornbæk, A. Oulasvirta, S. Reeves, and S. Bødker. 2015. What to Study in HCI? In Proceedings of the ACM CHI'15 Conference on Human Factors in Computing Systems. ACM Inc., New York, NY, 2385–2388.

[12] Z. Jordan, J.A. Nocera, A. Peters, S. Dray, and S. Kimani. 2016. A Living HCI Curriculum. In Proceedings of the 1st African Conference on Human Computer Interaction (AfriCHI'16). ACM Inc., New York, NY, 229–232.

[13] A. Jovanovic, M. Chignell, and O. St-Cyr. 2018. Prototyping the HCI Living Curriculum. Workshop at the ACM CHI'18 Conference on Human Factors in Computing Systems.

[14] J.A. Koepfler, L. Stark, P. Dourish, P. Sengers, and K. Shilton. 2014. Values & design in HCI education. In Proceedings of the ACM CHI'14 Conference on Human Factors in Computing System. ACM Inc., New York, NY, 127–130.

[15] I. Larsen-Ledet, N. Bressa, and J. Vermeulen. 2019. Reflections on Teaching a Mandatory HCI Course to Computer Science Undergraduates. In Proceedings of the 2019 EduCHI Symposium on HCI Teaching and Learning.

[16] J. Lazar, J. Preece, J. Gasen, and T. Winograd. 2002. New issues in teaching HCI: pinning a tail on a moving donkey. In Proceedings of the ACM CHI'02 Conference on Human Factors in Computing Systems. ACM Inc., New York, NY, 696–697.

[17] Z. Obrenovic. 2012. Rethinking HCI Education: Teaching Interactive Computing Concepts Based on the Experiential Learning Paradigm. Interactions 19, 66–70.

[18] A. Roudaut, O. Shaer, A. Girouard, and A.L. Kun. 2018. Identifying Challenges within HCI Education. In Proceedings of the ACM CHI 2018 Workshop on Developing a Community of Practice to Support Global HCI Education. ACM Inc., New York, NY.

[19] E. Sari and B. Wadhwa. 2015. Understanding HCI Education Across Asia-Pacific. In Proceedings of the ASEAN CHI Symposium'15, ASEAN CHI Symposium'15. ACM, New York, NY, 36–41.

[20] A. Sears, M.G. Williams, J.B. Gasen, T.T. Hewett, J. Karat, and G. McLaughlin. 1997. None of the above: What's really essential in HCI education? In Proceedings of the ACM CHI'97 Conference on Human Factors in Computing System. ACM Inc., New York, NY, 109.

[21] O. St-Cyr, C.M. MacDonald, E. Churchill. 2019. Global Perspectives on HCI Education. Workshop at the 2019 EduCHI Symposium on HCI Teaching and Learning.

[22] O. St-Cyr, C.M. MacDonald, E. Churchill, J. Preece, and A. Bowser. 2018. Developing a Community of Practice to Support Global HCI Education. Workshop at the ACM CHI'18 Conference on Human Factors in Computing Systems.

[23] M. Vorvoreanu, C.M. Gray, P. Parsons, and N. Rasche. 2017. Advancing UX Education: A Model for Integrated Studio Pedagogy. In Proceedings of the ACM CHI'17 Conference on Human Factors in Computing Systems. ACM Inc., New York, NY, 1441–1446.

[24] B. Wadhwa and E. Sari. 2014. HCI Education in Asia-Pacific. Workshop at the 26th Australian Computer-Human Interaction Conference on Designing Futures: the Future of Design (OzCHI'14).

[25] L. Wilcox, B. DiSalvo, D. Henneman, and Q. Wang. 2019. Design in the HCI Classroom: Setting a Research Agenda. In Proceedings of the 2019 Designing Interactive Systems Conference (DIS '19). ACM Inc., New York, NY, 871–883.

[26] T. Winograd. 1990. What Can We Teach About Human-computer Interaction? In Proceedings of the ACM CHI'90 Conference on Human Factors in Computing Systems. ACM Inc., New York, NY, 443–448.

[27] Y. Yi, W. Shengjin, S. Jiasong, and Z. Xian. 2017. Interest-Based Learning for Teaching a Human-Computer Interaction Course: Media and Cognition Course. In Proceedings of the IEEE International Conference on Frontiers in Education (FIE'17), 62–67.