# Computing Students' Learning Difficulties in HCI Education

**Alannah Oleson**
The Information School
University of Washington
Seattle, WA, USA
olesona@uw.edu

**Meron Solomon**
Art + Art History + Design
University of Washington
Seattle, WA, USA
meron@uw.edu

**Amy J. Ko**
The Information School
University of Washington
Seattle, WA, USA
ajko@uw.edu

## ABSTRACT
Software developers often make interface design decisions and work with designers. Therefore, computing students who seek to become developers need some education about interface design. While prior work has studied difficulties that *educators* face when teaching design to computing students, there is comparatively little work on the difficulties computing *students* face when learning HCI design skills. To uncover these difficulties, we conducted two qualitative studies consisting of surveys and interviews with (1) computing students and (2) educators who teach interface design to computing students. Qualitative analysis of their responses revealed 18 types of learning difficulties students might experience in HCI design education, including difficulties around the mechanics of design work, project management skills, the wicked nature of design problems, and distorted perspectives on design.

## Author Keywords
HCI education, interface design education, learning difficulties, pedagogical content knowledge

## CCS Concepts
•**Social and professional topics** → **Computing education;** •**Human-centered computing** → *Human computer interaction (HCI);*

## INTRODUCTION
In higher education computing programs, students are often taught to *engineer* software. However, developers often make *design* decisions that impact the usability, accessibility, and inclusiveness of their software, including at companies that lack design cultures [41], startups that lack designers [41], in open source projects without design workflows [42], and even at large companies where they manage or collaborate with design teams [46]. In each of these settings, understanding user experience (UX) and interaction design concepts is key to creating and collaborating on high-quality software.

Unfortunately, partly because many developers lack design literacy, software still routinely fails to be usable for diverse populations (e.g. [5, 13, 28, 43, 53]). Professional software engineers still struggle with design-related tasks like requirements elicitation [2] and interface creation [56]. Since design choices are not value-neutral, poorly designed software can unintentionally perpetuate harmful stereotypes [10, 11, 62] or disadvantage already-marginalized populations [13].

At the heart of this problem is the fact that many developers receive little to no design training before entering the workforce: Students in traditional computer science (CS) degree programs may take at most one human-computer interaction (HCI) or interface design class prior to graduation. Even when computing students take an HCI class, teaching them design skills is hard [68]. Educators often struggle to engage students [33, 48, 60], to override persistent perceptions that designerly aspects of HCI are "inessential" [15], "easy," or "commonsense" [17], and to accurately assess students' design work [9, 65, 70]. Additionally, much of this research is limited to educators' reflections on their own particular courses [45], so students may face difficulties that educators do not perceive.

The goal of this paper is to understand what computing students struggle with when learning to design software interfaces in order to inform HCI pedagogy. Our working definition of HCI design in this paper draws on the model described in Park and McKilligan's review of HCI design and design thinking [54], which focuses heavily on software interface design as a basic HCI design proficiency. For this exploratory study, we scope our investigation to software interface design learning, and we use the broad term *learning difficulty* to represent anything that prevents a student from effectively learning or applying software interface design concepts in school or as a practicing software developer. Our intended audience consists of higher education instructors who teach software interface design principles within computing-focused (rather than design-focused) departments, as well as researchers who study formal and informal HCI design learning contexts.

In this paper, we ask the research question *What difficulties do computing students face when learning and applying software interface design skills?* To answer this, we qualitatively analyzed survey responses (n=117) and interview transcripts (n=15) from computing students learning design skills in both formal (post-secondary classes) and informal (on-the-job) settings. From this, we identified 15 types of learning difficulties. We then validated the set through surveys (n=35) and follow-up interviews (n=8) with HCI educators who teach software interface design concepts to computing students. From educators' responses, we identified a further 3 types of difficulty, resulting in a final set of 18 learning difficulties. This set of student difficulties provides a potential basis for improving HCI design education. Finally, we discuss implications of this study for future research on HCI design pedagogy and educational practice. Our contributions are: (1) A set of difficulties computing students may face when learning and applying software interface design concepts, with examples; and (2) Validation of these difficulties from educators who teach interface design to computing students, with examples.

## RELATED WORK: HCI & UX PEDAGOGY

HCI has many of its roots in traditional CS programs, though it encompasses many disciplines. Faiola's Design Enterprise Model (DEM) [18] situates design, especially interface and interaction design, as a core competency for students learning HCI principles. Design skills are known to be difficult to teach and learn within HCI education contexts [18, 64, 68]. As discussed in the introduction, computing students often fail to engage meaningfully with the designerly aspects of HCI [33,48,60] or erroneously view design as easy, inessential work [15, 17]. HCI educators, who (especially in computing-focused departments) may not have design backgrounds, often struggle to assess design-related work adequately [9, 65]. HCI educators often face a time crunch in already overcrowded computing curricula as well, [15, 30], forcing educators to prioritize some topics and exclude others, which leads to wide variation in HCI coursework and course content.

To address these challenges, prior work has explored the efficacy of the studio approach for HCI design topics. Studios are intended to be "bridges" [7] between education and professional settings, providing a semi-authentic environment for students to develop skills they will use in future workplaces. While this body of work has provided valuable insights into how an HCI classroom can be structured, computing students may experience difficulty learning in studio environments due to their unstructured nature [36, 60]. Educators have also reported significant challenges successfully teaching studio-based classes [48], at least some of which were centered around motivating students to engage with concepts.

The related field of user experience (UX) pedagogy has explored the space of teaching and learning software interface design more extensively, often with potential implications for HCI education. Getto and Beecher drew on their combined expertise from teaching and industry to propose a model for incorporating UX education into existing higher education curricula [20]. Gray and colleagues' work on UX competency also provides recommendations for UX pedagogy which may transfer to HCI contexts. For instance, Gray's analysis of how student designers transition into professional communities indicates that perceptions of competence are often situated in a particular context and individual experience, which implies a need to focus on designerly identity formation and working within organizational constraints in UX higher education [26]. Gray et al. subsequently proposed a model of the interactions that occur between individual and group competence in professional UX contexts: Individual knowledge (often gained through formal education) forms a basis for individual competence, but it also interacts with informally gained knowledge which comes from the group or organizational context [27]. Both formal and informal learning play a role in UX competence under this model, which suggests a need to study learning difficulties in both contexts to gain a complete picture of students' educational experiences.

In both HCI and UX pedagogy, prior work has leveraged the frame of learning difficulties to gain insights into how to improve education. This kind of framing is beneficial to both students and educators [64]. Getto and Beecher's investigation identified the presence of significant institutional barriers to implementation of UX education and provided strategies for surmounting these barriers by calling for increased partnerships between academia and industry, though it did not directly address *student* learning difficulties [20]. Siegel and Stolterman framed their study of non-designers transitioning into design roles around a set of observed student learning barriers, which they drew upon to propose a framework for designing instructional activities for HCI design classrooms [64]. However, this particular study drew on the experiences of students of many different educational backgrounds taking part in a design-focused program. In contrast, our investigation seeks to understand the experiences of students with a specific (computing-based) educational background enrolled in a program where design is not the main focus. We seek to extend this body of work which uses student learning difficulties as a focus to inform HCI education around software interface design principles.

## STUDY 1: STUDENT PERSPECTIVES

### Method: Surveys and Interviews

To begin identifying the kinds of difficulties computing students face when learning software interface design skills, we collected and analyzed data from both surveys and interviews.

#### *Surveys of Students Formally Learning Design*

We surveyed undergraduate computing students enrolled in introductory software interface design classes at two large, public, U.S. universities:

- Computer science students enrolled in a course at University 1 (U1A), which focused on usability principles. This course was only mandatory for students enrolled in a non-default option within their major, and was generally taken in students' 3rd or 4th year.

- Information science students enrolled in a course at University 2 (two sections – U2B and U2C) focused on design methods [12] with a human-centered design model. This course was mandatory for all students in the major and typically taken in students' 2nd year or later. We considered the information science students at University 2 to have computing backgrounds due to the highly technical nature of their program and its strong emphasis on software development.

Due to the sequencing of the universities' programs, this was likely the first formal exposure to software interface design concepts for many students.

The surveys we used at each university contained similar items, adjusted only to fit the particular class's context. We prompted students to "*Write down any questions you still have about prototyping [or other relevant design topic] after today's lesson.*" We elicited learning difficulties by asking for students' questions about design-related topics (as opposed to asking them to directly report the nature of their struggle) because students may not have had the language or awareness to accurately identify the causes of their confusion, especially if they were new to design work and unfamiliar with terminology.

We also collected demographic data to verify students' fit with the population of study, including students' self-reported fields of study (to verify they were computing-related) and their self-reported experience with designing interfaces (to verify their novice status). We did not collect further demographic information (e.g. gender, race/ethnicity, age) both because we wished to keep the survey lightweight and because we did not believe it to be relevant to the topic of study during this initial investigation. At University 1, we administered the surveys electronically via a link sent to a learning management system; At University 2, a researcher attended the classes and administered surveys to students on paper.

From these three courses at two universities, we collected 117 responses (U1A: 13, U2B: 33, U2C: 71) representing perspectives from 88 students (U1A: 9, U2B: 41, U2C: 38) since some students wrote multiple questions.

### *Interviews with Students Informally Learning Design*

The perspectives collected through the above surveys represent learning difficulties experienced in formal educational contexts. However, learning can also occur in informal contexts, such as on the job or while working on software design projects for external clients. We therefore conducted interviews with developers who had practiced software interface design in diverse contexts, allowing us additional insight into informal learning processes. To qualify for an interview, participants needed to (a) self-identify as having a computing background, (b) have designed an interface for at least one piece of software, and (c) self-identify as a novice-to-intermediate software interface designer. We recruited participants through mailing lists and forum announcements and provided incentives of $20 to those who completed the interview. Though we did not limit our recruitment to students only, all interview participants were current or recently graduated undergraduate and graduate computing students. Therefore, we refer to interviewees as students when reporting results.

Interview participants met with a researcher on University 2's campus for a semi-structured interview consisting of two main sections. First, participants relayed their background and education, including details about how they first learned to design software interfaces. Second, participants described the most recent piece of software for which they had designed or created a software interface, then walked the interviewer through their design process in as much detail as they could recall. After that, they discussed their "typical" design process (whether or not it aligned with their most recent project) and any particular challenges they recalled during design work.

We conducted 15 interviews in total. Participants described interface design work in a wide array of contexts, ranging from intensive year-long projects with external clients such as graduate capstones, to entirely on-the-job self-teaching of design principles and concepts, providing rich insights into varied learning difficulties. We audio-recorded and transcribed the interviews, and took handwritten notes during the interviews to provide context. In total, we collected and analyzed about 245 minutes of audio.

### *Qualitative Analysis*

Two researchers performed the qualitative analyses:

- The 1st author, a computing education researcher with five years of research experience in HCI and design methods, including two years researching the overlap of software interface design and computing education.

- The 2nd author, a research assistant with one year of research experience in computing education and design methods as well as one year of UX design experience.

First, the two researchers collaboratively affinity diagrammed the 117 survey responses to generate initial themes for our coding effort with a sensitizing concept of *types of learning difficulty* [55]. Through iterative refinement and discussion, the researchers identified 13 types of difficulties, which formed the basis of the code set used in subsequent analysis.

Next, the researchers performed two rounds of deductive qualitative coding on the transcribed interview data, segmented by sentence for analysis. In the first round of coding, to scope the amount of data to analyze, the researchers marked each sentence in the interview transcript as containing (1–N) or not containing (0) evidence to suggest the presence of a learning difficulty. Sentences could contain multiple types of learning difficulties, in which case researchers marked the number of distinct types present. The researchers conducted the first round of coding collaboratively over three 1-hour meetings, discussing interpretations and eventually achieving agreement.

The second round of deductive coding focused on sentences that contained at least one type of difficulty. The researchers divided the data set and each qualitatively coded half the data using the code set of learning difficulties identified in the surveys as a basis. We allowed for multiple codes per sentence (since one long sentence might contain evidence of many types of difficulties) and no codes per sentence (since an interview participant might talk about a new type of difficulty not observed in the survey data). Once the researchers finished their respective deductive analyses, they met to discuss interpretations and address any discrepancies in the application of the code set, then adjusted their coded data as needed.

Finally, for sentences in the interview data that appeared to contain difficulties but did not fit into our existing code set, the researchers collaboratively affinity diagrammed and inductively coded the sentences to identify themes. This inductive analysis identified two additional types of learning difficulties that were not present in the survey data. Adding these two new difficulties produced an updated set of 15 total student-reported learning difficulties, encompassing data from both the surveys and the interviews.

| Tag | Student Learning Difficulty |
|---|---|
| WHAT | What is design? |
| WHY | Why do we do this design activity in this way? |
| HOW | How do I perform this design method? |
| INFO | How/where do I find a design resource? |
| ADAPT | How do I adapt parts of this design into my design? |
| SYNTH | How do I interpret this feedback? |
| TEAM | How do I work with my teammates effectively? |
| STAKE | How do I work with clients and stakeholders effectively? |
| LIMIT | How do I design with limited resources? |
| SCOPE | How do I scope this design problem? |
| STAGE | When should I move to the next design stage? |
| EVAL | How can I choose between options? |
| BIAS | How can I avoid biasing my design? |
| DIVRS | How do I design for diversity? |
| ID | Am I the kind of person that can or should do design? |

**Table 1. Descriptions of student-reported learning difficulties identified in Study 1 surveys and interviews.**

## Study 1 Results: Student-Reported Difficulties

Table 1 describes the 15 student-reported difficulties we found during our analysis. We adhere to the perspective on qualitative coding presented by Hammer and Berland [31], treating the results of our coding effort as organizations of claims about data rather than quantitative data (i.e., measuring inter-rater reliability) in and of themselves. As a result, we do not report code frequencies, preferring instead to focus on descriptions of code instances observed in our data. For ease of reference, we assign each learning difficulty observed in Study 1 a tag by which we refer to it throughout the paper (see Table 1). Due to length constraints, we illustrate each difficulty by providing two representative quotes: One from the survey data and one from the interview transcripts (or two from the interviews, if the difficulty was not observed in surveys). IDs preceding survey quotes (e.g. U2C) represent the university and class of the quoted student. IDs preceding interview quotes represent the speaker (e.g. P8). We then provide a short description of each difficulty by characterizing common themes that represented it in surveys and interviews.

`WHAT`*: What is design?*

- U2C: *"I'm confused about what exactly counts as a prototype? Does it have to be a physical object?"*
- P8: *"[Researcher asks how they first learned design.] I did not. [laughs] I just did it. ... I don't think I've actually ever been told how things should be or how things should look... I have no knowledge of how I *should* design things."*

`WHAT` difficulties occurred when a student lacked declarative knowledge, such as facts about design. Students reporting these kinds of learning difficulties were confused about the nature of various design objects (e.g. prototypes, wireframes) and activities (e.g. stakeholder analysis, sketching). In the surveys, though some `WHAT` difficulties should be expected with novice designers, students still reported uncertainty about the nature of design concepts *directly after* lessons on that topic, as the U2C student did above after a lesson on prototyping. In interviews, students reported `WHAT` difficulties when they discussed having to do design work without much (if any) formal training. Recall that an inclusion criteria for the interviews was that the participant had designed at least one software interface; despite this, some of the interviewees reported that they did not even know what UX design was when they began their projects, or that they never had a concrete design process.

`WHY`*: Why do we do this design activity in this way?*

- U2C: *"The thing I made on prototype won't show on the actual app. What's [the] point for us to prototype unuseful interface?"*
- P5: *"So unfortunately user experience was the last part of it. ... We started off by making it a very useful tool. And usable, but then usable came second."*

`WHY` difficulties arose when students did not understand the reasoning behind performing a design activity in a particular way. In the surveys, these difficulties manifested as questions about why students were spending time working on interfaces (prototypes) that were not the actual end products, as well as questions about the reasoning behind particular design methods' utilities (such as brainstorming in isolation before bringing your ideas to a large group). `WHY` difficulties did not necessarily prevent students from practicing design: Students reporting them in interviews still completed design work, though they simultaneously reported confusion about the rationale behind design tasks. This led some students to question the importance of design work overall (such as P5 described above) and put off interface design work until the final stages of implementation.

`HOW`*: How do I perform this design method?*

- U2B: *"How do we deal with having more than one design in the beginning? ... How do you compare ideas?"*
- P7: *" [We used] trial and error... My mentor and I would have an idea, we'd implement it, and we'd test it with people. And then it would crash and burn and work terribly."*

Students who reported `HOW` difficulties in the surveys asked questions about how to perform the steps of different design methods. In the interviews, students like P7 often knew that their current practices were not necessarily optimal, but they did not know what steps they could take toward formal design methods. Interview participants that indicated these types of difficulties sometimes went on to describe how confusing their (lack of a) design process was, or how the resulting design was poor quality and took longer to finalize than expected.

`INFO`*: How/where do I find a design resource?*

- P14: *"When you learn new software, you don't know what the software is capable of doing. [P14 describes how they would follow YouTube tutorials when available.] But there's sometimes things that I cannot look for [with] tutorials. There's no tutorial online about the topic that I want to go in. ... all little skills that I needed to pick up because someone didn't teach me that."*
- P12: *"It was just us developers trying to do as well as we can. I hadn't studied the UX process before. ... While I was working, I just would consume as many articles as I could on the web. But there was no process I could follow."*

We only observed `INFO` difficulties in the interview data. These kinds of difficulties most often manifested as students sought information to help them design – whether in the form of a tutorial, a description of a design method, an article about interface design, or an example to inform their design. Interview participants who indicated they struggled with INFO difficulties sometimes recalled that issues finding resources slowed their progress or prevented them from doing design work altogether. INFO difficulties often came up when students had to learn design concepts independently (i.e. that their coursework had not taught them).

ADAPT*: How do I adapt parts of this design into my design?*

- P11:*"I found some websites for [design] references in different systems. But, it's like, this looks pretty, but how do I apply it to my prototype? I don't know how to do that."*
- P14:*" It's better for me to look for how people do it because when they're doing it, I can learn about other stuff as well as like how the system works and how do they come together to achieve the thing that I want to do... Later when I start to get used to the software, I would change a little bit. It's like, oh, okay, here they do [a transition] like 0.5 second, I can do like 0.3. ... Start doing those little changes that is fitting my expectation more."*

ADAPT difficulties also only appeared in the interview data. These difficulties revolved around students' struggles to adapt elements of an example to their own designs. "Stealing" successful solutions to analogous design problems [23, 32] and composing features of those ideas to create novel solutions [51,66] are both known design proficiencies, but computing students reporting ADAPT difficulties struggled to perform these tasks. Some students who reported this kind of difficulty were on teams consisting only of software developers (no designers) and felt they did not have anyone to turn to for assistance. In any case, students experiencing these difficulties often took longer than they expected to finish design tasks due to the extra time needed to adapt designs.

SYNTH*: How do I interpret this feedback?*

- U1A:*"How do I prioritize the information/research I've completed so that I can properly inform my design?"*
- P9:*"Our findings from research were in a different format than the design itself. So it's going from that stuff [results], like, we were doing an affinity analysis of findings from the research, from that to the visual layout and stuff like that. ... [You] kind of have to double check to make sure that you're actually doing the research justice and you're actually serving their needs in this new interface [version]."*

Students who reported SYNTH difficulties often struggled to synthesize feedback they received from critique sessions or user evaluations in ways that could inform subsequent design choices. In the surveys, students who experienced this difficulty often asked about how to derive requirements from broad initial research efforts, as well as how to determine the severity of usability issues discovered through testing. In interviews, students who struggled with SYNTH difficulties spoke about uncertainty over whether their interpretations of feedback were "correct," especially when the feedback received did not correspond to an obvious design decision. Students also reported interpreting feedback incorrectly (i.e. in ways that did not make the interface more usable or useful), which required extra time and resources to remedy the error.

TEAM*: How do I work with my teammates effectively?*

- U2C:*"How to efficiently communicate with team members?"*
- P15:*"The initial developer I worked with, from the [country name] team, he had come with this fixed mindset about how much effort he needed to put in, not how much effort was actually required for the project. It was more like, 'Oh, okay, maybe we do need to do all this [work], but I'm only going to put in this much amount of time."'*

Collaboration with teammates on design work was often difficult for students. Students reported TEAM difficulties effectively communicating ideas to others, resolving conflicts (such as over differing interpretations of results or ownership of ideas), or working alongside teammates who did not want to put in sufficient time to do design work well. Students reported that TEAM difficulties slowed their progress, and a few students said they wished they had received training on how to effectively design in teams.

STAKE*: How do I work with clients/stakeholders effectively?*

- U2C:*"Is it true that clients don't know what they want until they have seen a lot of things they don't want?"*
- P9:*"The biggest issue that jumps to mind is the jargon that we [designers] use. ... Eventually, if you're in these environments, you just use it to describe normal situations. And then once you're with users... You have to kind of catch yourself, when you're describing how something is going to be used or how you're going to collect information ... explaining these concepts to non-technical users."*

Students also reported difficulties collaborating with clients and stakeholders on design projects. STAKE difficulties manifested when students struggled to elicit requirements, communicate domain-specific information, or present results of design work to clients who lacked design domain expertise. In some cases, such as P9's above, students had to adapt their communication styles around clients and stakeholders, which many found difficult. In other cases, STAKE difficulties arose when clients collaborated poorly with students, which prevented the students from gaining access to needed resources.

LIMIT*: How do I design with limited resources?*

- U2C:*"How to balance creativity and do-ability?"*
- P5:*"We started working on it [the design] and realized that maybe we should have asked them [users] that question, and then that becomes the second iteration, which is costly in terms of time and money. ... You don't want to go in iterations. You want to actually spend time to get lesser iterations of things. Ask the right questions the first time."*

Students often reported difficulties managing limited resources—whether the resource in question was time, money, access to users, or other constraints imposed by the environment they were working in. In surveys, students questioned how to prioritize design tasks based on cost-effectiveness, time-efficiency, and feasibility of implementation. In the interviews, students reported struggling with tight deadlines (especially those that forced them to change their planned designs), accessing representative users, and balancing practical concerns with the desire to meet all design goals. Students who experienced LIMIT difficulties sometimes had to skip parts of the design process or otherwise leave out features.

SCOPE: *How do I scope this design problem?*

- U1A: *"How do you define a design problem?"*
- P9: *"Sometimes there is changing expectations ... There's certain things that we just couldn't do anymore ... scope, basically. Challenging to figure out, if we have one intention and that ends up not being feasible, how do we still honor the users, the expectations and requirements, while having to compromise on other parts of the application."*

Students reported SCOPE difficulties when they tried to define the boundaries of design problems. These attempts at scoping sometimes occured at the beginning of the design process, when students first began to decide on what they wanted to create. SCOPE difficulties also occurred in the middle of students' design processes as students realized that their initial conception of the problem was not adequate—a known byproduct of the design process in literature (c.f. "productive failure" as described in [57]). Students who struggled with SCOPE difficulties reported delays in getting started or having to do "extra" unwanted iteration.

STAGE: *When should I move to the next design stage?*

- U2B: *"How many ideas is too many [during ideation]? When do you know you have enough?"*
- P5: *"It [the project]'s still not done yet, and I don't think there will ever be a point. I think we're going to keep taking suggestions. For me personally, I don't think we ever reach a point where everybody becomes happy."*

Students often reported STAGE difficulties in the middle of their design processes, especially when trying to determine criteria which signified the need to move on with design work. Students wanted to know what was "good enough" to move on from brainstorming, how many user studies or usability tests needed to be run, or what constituted enough prototype iterations created. Those who experienced STAGE difficulties reported frustration with design work that never seemed to be "done" (similar to P5 above) and confusion over when design requirements were adequately met.

EVAL: *How can I choose between options?*

- U2B: *"How do you choose the right method for the job?"*
- P10: *"We actually conduct different interviews with different user groups. Then we understand each groups' pain points, then kinda have to try to solve all the pain points at the same time. But of course, it's really hard. This is why sometimes you have to do some tradeoffs. ... For this one [project] we only focus on one of the user groups."*

EVAL difficulties arose when students struggled to evaluate which of several options was most fit for their project. In the surveys, students asked questions about deciding on the "best" or "ideal" design method or activity, as well as how to decide between multiple applicable approaches. In interviews, students who reported EVAL difficulties spoke about the difficulty inherent in evaluating tradeoffs when trying to satisfy the requirements of competing design goals and deciding which user needs to prioritize over others. Students who did not know how to progress past EVAL difficulties sometimes reported spending more time than they would have liked in planning stages, or choosing arbitrarily between options without grounding the decision in design rationale.

BIAS: *How can I avoid biasing my design?*

- U1A: *"How do we know if a project is truely [sic] a usability issue or if we are displaying confirmation bias?"*
- P8: *"I don't think I've actually ever been told how things should be or how things should look ... I just judge based on things I've seen that I like. I kind of evaluate everything as *I* look at it. But I have no idea how other people will interact with it."*

Students who reported BIAS difficulties struggled to prevent their own inclinations and biases from impacting their design decisions. In the surveys, students often reported being aware that their biases could influence their designs (as in the above survey response), but did not know what to do about it or how to recognize when it had happened. In the interviews, students experienced BIAS difficulties when they assumed that if they personally found the design aesthetically pleasing and usable, others surely would as well. Sometimes, like P8 described above, students did not know any better ways to design interfaces than simply relying on their own evaluations. Students who reported these difficulties often later (e.g. during user testing) found that their designs were not as high quality as they had believed, leading some to start over completely.

DIVRS: *How do I design for diversity?*

- U2C: *"How do you design an application that works for everyone?"*
- P1: *"[When designing] I'm thinking about how do I build something for the people that I don't understand, necessarily, their cultural experiences or how they view, or their perception of something. ... It [the application] was supposed to be specific to [region other than designer's place of residence], but you know, when it comes to development stuff, there's just so, so many differences between culture and that kind of thing that it still makes it difficult."*

DIVRS difficulties imply that students struggled to perspective-take or empathize with their interface's target users. Students reported difficulties designing for diverse abilities and usage styles, especially when users' experiences were very dissimilar to their own. Difficulties around DIVRS often came up alongside designing for accessibility or inclusion, but were part of broader discussions around usability as well, especially when students began to realize that people could interpret designs quite differently. Students who reported DIVRS difficulties in the interviews sometimes went on to describe how their design solutions failed to represent users' true needs, and were therefore less useful than intended.

`ID`*: Am I the kind of person that can/should do design?*

- U2C: *"Is design an easy job that every one can do?"*
- P7: *"I'm a person who does design to fill the need of there being design done. ... If I compare myself to people who identify as UX designers, I think they spend a lot more time with wireframes and paper prototypes and thinking about the theory behind their designs. I don't really identify as a UX designer, I'm just a person who designs things. I build stuff and test it, and if it doesn't work I change it."*

`ID` difficulties were one of the least commonly observed in our data, though potentially some of the most problematic. These kinds of learning difficulties manifested in the surveys when students asked about intrinsic qualities designers should possess to be successful. In the interviews, students' `ID` difficulties sometimes manifested when students were reluctant to claim the title of designer, even when they clearly performed design tasks, like P7 above. Students reporting this reluctance spoke about designing out of necessity rather than choice (e.g., they were working on a developer team with no designers) and not feeling like they actually "did" design work, even if they had clearly made design decisions and performed design activities. `ID` difficulties may be tied to a lack of design self-efficacy [3]: Students may not have been confident in their design abilities, and thus chose not to identify as designers.

## STUDY 2: EDUCATOR PERSPECTIVES

The student perspectives represented in the previous section are important to understand HCI education learning difficulties, especially as the 15 difficulties we found were observed across multiple learning contexts. However, educators can also provide perspectives on learning difficulties in the HCI classroom, which can consist of multiple years of experience watching their students struggle to learn software interface design concepts. To learn from these experiences, and to further validate the existence of the student difficulties presented above, we designed and deployed an online survey.

### Method

#### *Survey Structure*

We created and deployed the English-only educator survey using an online survey platform. The survey began by verifying that the educator met inclusion criteria: (a) 18 years of age or older, (b) taught computing students (here, presented as "students who may create software interfaces in their future careers" to signal inclusion of non-CS departments), and (c) taught software interface design concepts to these students. The survey took educators who affirmed all three to the next set of questions, which we designed to validate the existence of the 15 student-reported difficulties uncovered by Study 1.

For each of the 15 difficulties, we presented educators with a description of the difficulty type (similar to the text in Table 1), then asked them to report if they had observed this type of difficulty in their classes. Educators could respond in three ways: "Yes", "No, but I believe students might experience this difficulty", or "No, and I don't believe this difficulty exists." The two "No" variants added a small amount of descriptive data to an otherwise closed-ended survey response and allow us to better understand educators' perceptions of these difficulties. We held the order of items corresponding to `WHAT` and `WHY` difficulties constant across surveys to allow educators to acclimate to the question format, since we believed these two kinds of difficulties to be easily identifiable to educators. The subsequent order of items corresponding to the other difficulties was randomized to limit fatigue effects.

To supplement the above closed-ended survey responses with qualitative data, we included one open-ended item asking educators to describe an interesting instance of student learning difficulty they had observed. We hoped for this item to surface learning difficulties we had not observed in Study 1. Finally, we ended the survey with demographic questions about educators' backgrounds and a field for educators to leave their email if they were open to a follow-up interview.

#### *Recruiting & Respondents*

We recruited through four channels, offering a high-level summary of the survey's responses and implications for teaching as an incentive:

- Twitter. Our tweet received 7,204 impressions, 18 retweets, and 21 likes, and 33 clicks on the link by survey close.
- A closed Facebook HCI educator group with 217 members. The post was seen by 80 members and received 3 likes.
- Two Slack groups (40 members and 54 members) targeted at HCI educators, whose membership likely overlapped significantly with the Facebook group due to shared leadership.
- Targeted emails (77 total) to HCI and interface design educators who provided their contact information to the research team during previous studies relating to this topic.

The survey remained open for 26 days, with the majority of responses received in the 1st week. We received 52 responses to the survey. Of those 52, 36 finished the entire survey (a drop out rate of 30.8%), and of those 36, 35 (97.2%) met our inclusion criteria. We discarded the 17 responses that were unfinished or did not meet inclusion criteria. The results below therefore represent perspectives from 35 HCI educators who teach software interface design concepts to computing students. To put the 35 responses in perspective, in recent proceedings, a few hundred institutions publish HCI-related work at the ACM CHI conference each year. Assuming one educator who fits our target population at each institution, our 35 responses might represent 5-10% of the current population of HCI educators. While this number is still relatively small, we feel that it is representative enough to provide initial insights, especially since many educators reported similar themes in their open-ended responses (suggesting saturation).

Table 2 shows an overview of educators' demographics. Most taught in the United States at large, public universities, and most self-reported their main field of study or practice as HCI or CS. The educators reported a wide range of years of teaching experience, though most reported 1-5 years of overall experience and 1-5 years of experience specifically teaching interface design skills to computing students.

| Country | | Main Field | | Years Teaching (Total) | | Years Teaching HCI | | Main Teaching Institution | |
|---|---|---|---|---|---|---|---|---|---|
| US: | 26 | HCI: | 16 | <1 year: | 0 | <1 year: | 2 | Large, public university: | 21 |
| Canada: | 2 | CS: | 9 | 1-5 years: | 15 | 1-5 years: | 15 | Small, public university: | 5 |
| Germany: | 2 | Soft. Eng: | 3 | 5-10 years: | 5 | 5-10 years: | 5 | Large, private university: | 3 |
| Austria: | 1 | Design: | 3 | 10-20 years: | 6 | 10-20 years: | 8 | Small, private university: | 2 |
| Denmark: | 1 | CSCW: | 1 | 20+ years: | 9 | 20+ years: | 4 | Community/Junior College: | 1 |
| Morocco: | 1 | Web Design: | 1 | | | | | Professional training program: | 1 |
| Philippines: | 1 | User Research: | 1 | | | | | Other (did not report): | 1 |
| UK: | 1 | UX: | 1 | | | | | | |

**Table 2. Demographics of the 35 educators who responded to the Study 2 survey.**

*Follow-up Interviews*
To further explore educators' perspectives, we followed up with a subset of the educators who both left their email and answered the open-ended question about an interesting instance of student learning difficulty. The goals of these follow-up emails included clarifying details of educators' responses and gaining insight into educators' perceptions of the student learning difficulties. Of the 35 educators, 17 provided both contact information and an description of a time they noticed students struggle. The 1st author reached out to 13 of these educators by email with targeted questions about their open ended responses. Of these, 8 responded, providing additional qualitative data.

*Qualitative Analysis*
We combined the data received from educators' follow-up interviews with the data from the open-ended responses on the survey. The resulting data consisted of a set of qualitative descriptions of student learning difficulties, encompassing perspectives from 27 of our 35 educators. To analyze this data, the same pair of researchers from Study 1 performed a thematic analysis [55]. The primary analyst (1st author) examined the text of the qualitative data and annotated it with memos indicating each time an educator wrote about a type of student learning difficulty. The secondary analyst (2nd author) did the same, verifying the primary analyst's notations and adding their own. The two analysts then collaboratively affinity diagrammed the memoized data with a sensitizing concept of *types of learning difficulty* to align with the analysis perspective used in Study 1. Loosely, this resulted in two categories of data: Student learning difficulties that we identified in Study 1 (which served to verify the existence of student-reported difficulties), and new difficulties that we had not observed in Study 1. For the data that indicated new difficulties, the analysts performed a subsequent round of collaborative inductive coding to surface 3 new types of student learning difficulties, which we present below. As before, we do not treat the results of this analysis as quantitative data, but rather as an organization of claims about data [31].

## Study 2 Results: Educator-Reported Difficulties

Table 3 shows the results for the closed-ended survey questions about the student-reported difficulties from Study 1. For each of the 15 student-reported learning difficulties, at least some educators reported they had observed it in their classes. Educators' open-ended responses also described instances of nearly all of the struggles that students had self-reported in Study 1 (see Table 4 in the Discussion for an overview).

Our qualitative analysis of educators' open-ended responses discovered three additional learning difficulties beyond those

| | Educator Responses (out of 35) | | |
|---|---|---|---|
| Difficulty | Yes, seen it | No, but might exist | No, does not exist |
| `WHAT` | 19 (54.3%) | 14 (40.0%) | 2 (5.7%) |
| `WHY` | 22 (62.9%) | 13 (37.1%) | 0 (0.0%) |
| `HOW` | 21 (60.0%) | 13 (37.1%) | 1 (2.9%) |
| `INFO` | 15 (42.9%) | 14 (40.0%) | 6 (17.1%) |
| `ADAPT` | 13 (37.1%) | 17 (48.6%) | 5 (14.3%) |
| `SYNTH` | 23 (65.7%) | 11 (31.4%) | 1 (2.9%) |
| `TEAM` | 31 (88.6%) | 4 (11.4%) | 0 (0.0%) |
| `STAKE` | 18 (51.4%) | 15 (42.9%) | 2 (5.7%) |
| `LIMIT` | 24 (68.6%) | 10 (28.6%) | 1 (2.9%) |
| `SCOPE` | 30 (85.7%) | 5 (14.3%) | 0 (0.0%) |
| `STAGE` | 25 (71.4%) | 9 (25.7%) | 1 (2.9%) |
| `EVAL` | 21 (60.0%) | 13 (37.1%) | 1 (2.9%) |
| `BIAS` | 17 (48.6%) | 17 (48.6%) | 1 (2.9%) |
| `DIVRS` | 19 (54.3%) | 13 (37.1%) | 3 (8.6%) |
| `ID` | 14 (40.0%) | 18 (51.4%) | 3 (8.6%) |

**Table 3. Frequency of educator responses on the Study 2 survey for items corresponding to student-reported learning difficulties from Study 1. Percentages indicate proportions out of 35.**

we discovered in Study 1. The overarching theme tying these three difficulties together was that *students did not necessarily perceive difficulties they experience as struggles*, even though, from an educator's perspective, it was clear that the student was not successfully learning or applying design knowledge. One educator characterized these difficulties as follows:

- *"Often the problems I see are best categorized as "unknown unknowns"—where the student confidently conclude[s] they know what to do next, how to ask a question, or how to apply a design principle (or decide they don't need to apply it), but are actually wrong."*

`WARP`*: Students hold inaccurate perceptions of design*
Some educators reported that their students held inaccurate perceptions of what design entailed or how it related to technical (programming) work. For instance, one educator described how making the interface design class mandatory for software engineering majors revealed resistance to learning:

- *"[M]any of the students actually had little to no interest in engaging with the material and often had condescending comments such as "I don't get the point of all this requirements gathering"... definitely was a challenge to explain to a lot of these students why design thinking mattered."*

Other educators reported similar resistance in their classes, relaying that students who thought interface design was only about making the software "look pretty" sometimes failed to engage with class material enough to learn anything. One educator tied `WARP` difficulties to design self-efficacy:

- *"A lot of students have been conditioned to think that they "can't" do certain things (e.g. drawing), and it's really hard* *to get them out of the mindset. It sometimes turned into stubbornness, where a small number of students have tried to "prove" they don't need interaction design to do things and they know better."*

These difficulties are consistent with prior work in HCI education reporting inaccurate perceptions of interface design from computing students [15, 25].

`STUCK`*: Students fixate on conventional design patterns*

Educators also reported that students often adopted elements of conventional designs without considering if these elements fit their specific design goals, assuming that there were certain aspects of interfaces that were "not allowed" to be changed:

- *"[S]tudents struggle most with thinking deeply about the root cause of usability issues and rethinking bigger decisions... Students are often most comfortable adopting what they see as a standard design or approach and have a harder time rethinking fundamental assumptions [that] they never considered to be explicit choices at all."*

One educator elaborated that `STUCK` difficulties prevented students from designing software that fit their users' needs:

- *"In many cases it seems they had a solution in mind and focussed [sic] on this solution rather than finding out more about the participants."*

Design fixation is a known problem for novice designers, who may not even be aware that they are fixating [38].

`RUSH`*: Students rush to implement and discount design work*

Finally, educators reported that students often rushed through the early stages of design work and focused entirely too much on implementation details. One educator reported their students rushed through prototyping:

- *"Many students like to jump into creating a higher-fidelity prototype from the beginning. They struggle to justify why it is important to start implementing their design ideas through low-fidelity prototyping."*

Another reported that `RUSH` difficulties might lead students to focus on low-level details before solidifying the high-level structure of their designs:

- *"Confusion between wireframing and high fidelity mockups. Students might spend time on visual design while still in the ideation/architecture stage."*

One educator related `RUSH` difficulties to the way prior classes conditioned students to approach programming problems:

- *"They tend to approach interface design like programming in that they assume that if they do the steps and get some results, then they are successful. It's something of a "as long as it compiles and runs on the test data, my job is done" mentality. I find the most success when I (or my TAs) push them to consider many of the issues you brought up [in the survey]; otherwise they will just get things done as quickly as possible, a bad recipe for interface design."*

Educators reported that students who struggled with `RUSH` difficulties produced designs that provided little value to their users, though students often failed to identify this behavior as the cause of their poor results.

## DISCUSSION AND CONCLUDING REMARKS

The goal of our study was to identify different kinds of difficulties computing students face when learning about software interface design in order to support the development of HCI pedagogy. Table 4 lists each type of difficulty we observed and the data sources supporting it, including relevant ties to prior work. We found at least four overarching categories of difficulty reported by students and educators:

- *Difficulties around how to do design work (`WHAT, WHY, HOW, INFO, ADAPT, and SYNTH`).* These arose when students struggled to understand the mechanics of interface design work, and often slowed down or prevented students' progress on design problems.
- *Difficulties around project management skills (`TEAM, STAKE, and LIMIT`).* These arose when students struggled to collaborate with others or manage limited resources, sometimes leading to communication breakdowns or the abandonment of parts of the design process.
- *Difficulties around the wickedness of design problems (`SCOPE, STAGE, and EVAL`).* These arose when students struggled with the "wickedness" [61] of design problems with unclear definitions and no definitively correct answers. Students facing these difficulties reported frustration and confusion over the ambiguity of design work.
- *Difficulties around distorted perspectives (`BIAS, DIVRS, ID, WARP, STUCK, and RUSH`).* These arose when students either had difficulties taking the perspectives of others, or when they did not realize that their own perspectives were at odds with designing high quality interfaces. Students may or may not have realized they faced these difficulties.

The set of 18 student learning difficulties presented in this paper provides one component of the knowledge needed to more effectively teach software interface design concepts to computing students. For some of the difficulties (`WHAT`, `WHY`, `HOW`, `INFO`, `ADAPT`, `TEAM`, `STAKE`, `SCOPE`, `BIAS`, `DIVRS`, `ID`, `WARP`, and `STUCK`), prior work from learning science, HCI, software engineering, or design education indicates that they might be difficult for students who are novice designers (see Table 4). Others (`SYNTH`, `LIMIT`, `STAGE`, `EVAL`, and `RUSH`) appear to be undiscussed in relevant prior literature, which may imply that they are unique to this topic and audience.

Though the data we collected was rich, some aspects of our study design limit the generalizability of these findings. Due to the high variation between HCI courses across institutions, we cannot be sure that these observations generalize across all contexts. For Study 1, our surveys gathered data from students at only one single instant during instruction and were presented slightly differently to fit the context of each class. The surveys also were only deployed at two U.S. based universities. Our Study 1 interviews were conducted in-person on a university campus, which may have limited participation. Further, students in Study 1 likely varied in their ability to reflect on their own learning. The educators' perspectives

| | | Students (Study 1) | | Educators (Study 2) | | |
|---|---|---|---|---|---|---|
| **ID** | **Description** | **Surveys** | **Interviews** | **Surveys** | **Qual. Data** | **Prior Work** |
| `WHAT` | What is design? | ✓ | ✓ | ✓ | ✓ | [8] |
| `WHY` | Why do we do this design activity in this way? | ✓ | ✓ | ✓ | ✓ | [8] |
| `HOW` | How do I perform this design method? | ✓ | ✓ | ✓ | ✓ | [36] |
| `INFO` | How/where do I find a design resource? | | ✓ | ✓ | | [58] |
| `ADAPT` | How do I adapt parts of this design into my design? | | ✓ | ✓ | ✓ | [23, 66] |
| `SYNTH` | How do I interpret this feedback? | ✓ | ✓ | ✓ | ✓ | |
| `TEAM` | How do I work with my teammates effectively? | ✓ | ✓ | ✓ | ✓ | [14] |
| `STAKE` | How do I work with clients and stakeholders effectively? | ✓ | ✓ | ✓ | ✓ | [14] |
| `LIMIT` | How do I design with limited resources? | ✓ | ✓ | ✓ | ✓ | |
| `SCOPE` | How do I scope this design problem? | ✓ | ✓ | ✓ | ✓ | [2, 22] |
| `STAGE` | When should I move to the next design stage? | ✓ | ✓ | ✓ | ✓ | |
| `EVAL` | How can I choose between options? | ✓ | ✓ | ✓ | ✓ | |
| `BIAS` | How can I avoid biasing my design? | ✓ | ✓ | ✓ | ✓ | [53] |
| `DIVRS` | How do I design for diversity? | ✓ | ✓ | ✓ | ✓ | [53] |
| `ID` | Am I the kind of person that can or should do design? | ✓ | ✓ | ✓ | ✓ | [3, 4] |
| `WARP` | Students hold inaccurate perceptions of design. | | | | ✓ | [15, 25] |
| `STUCK` | Students fixate on conventional design patterns. | | | | ✓ | [16, 38] |
| `RUSH` | Students rush to implement and discount design work. | | | | ✓ | |

**Table 4. Triangulation: Each student-reported learning difficulty was supported by at least three data sources, while the three educator-reported learning difficulties indicate struggles students might not have known they faced.**

provided in Study 2 expanded our understanding of student learning difficulties, but they also came from a relatively small number of educators who fit our inclusion criteria. Several factors likely influenced what kind of data we were able to collect, such as the timing of the survey's deployment, the kind of educators who were motivated enough to answer our survey, and educators' own abilities to reflect upon and recall students' experiences in their classes. To safeguard against these limitations, we relied on extensive use of triangulation with multiple data sources and with prior work, as seen in Table 4. However, some of the interpretations we present might have been different if we had studied other students or other teachers. Future work in this area should attempt to discover if these difficulties persist across varied educational contexts and whether other difficulties exist that we did not observe.

Nonetheless, our findings reveal a number of interesting implications for research. For instance, how prevalent are these difficulties in broader contexts? Under what conditions (e.g., studio-based vs. traditional lecture-based classes) might computing students experience these kinds of difficulties more or less often? As HCI expands beyond higher education into primary and secondary curricula (like Exploring Computer Science [24] or Code.org [1]), will these learning difficulties still hold? And what are effective strategies to mitigate students' learning difficulties that fit these categories? The `RUSH` difficulty revealed by educators in Study 2 also suggests an interesting hypothesis: the way we teach computing students to create software and write code may make them less likely to succeed at interface design work. Future work in this area should explore the extent to which prior computing knowledge influences students' experiences with these difficulties.

Our results also contribute to the discourse around *pedagogical content knowledge (PCK)* [63] development for HCI design education. PCK is domain-specific [29, 35, 37] and consists of knowledge of pedagogical strategies to teach a particular topic, in a particular context, to a particular audience. Exact definitions of the components of PCK vary (c.f. [8, 19, 50]), but knowledge of student learning difficulties is generally considered a core aspect. Our field has only begun to investigate the nature of computing PCK within the past decade, from primary and secondary learning environments [6, 21, 49, 59, 67], to both general [34, 35] and specific [39, 40, 44, 47, 52, 53, 69] aspects of post-secondary CS education. A prior study of ours did explore PCK for teaching software interface design skills [53], but it was scoped specifically to teaching a particular gender-inclusive interface design method and focused on educators' pedagogical strategies rather than students' perspectives. Therefore, the set of student learning difficulties described in this paper provides some of the first foundations for future research on PCK for general HCI design education. Further exploring this space might enable more effective use of instruction time in HCI classes (which are known to suffer from time constraints already [15]) through the development of more effective learning materials, or even help shorten the onboarding time for new HCI design educators—an important pursuit to ensure we have enough teachers to keep pace with the rapid growth of computing education.

Equipped with this better understanding of student learning difficulties, we can begin to deepen our understanding of how to provide computing students with effective design educations. Implementing this newly gained knowledge in curricula and pedagogy will lead to better teaching and learning around HCI design concepts. Through this effort, the software industry as a whole will benefit from a pool of design-literate computing graduates who enter the workforce ready to understand and contribute to many aspects of large projects, aware of the impacts of their design choices. Developers will be empowered to design usable, accessible, ethical, and inclusive software interfaces, allowing more diverse populations to engage with various technologies and participate in today's computing-infused world.

## ACKNOWLEDGMENTS

We thank our participants for their role in helping us build a better understanding of HCI education, as well as our reviewers for their advice. This work was supported by NSF grants DGE-1762114, 1735123, 1539179, 1703304, and 1836813.

## REFERENCES

[1] 2018. CS Discoveries Curriculum Guide 2018 - 2019. (2018). https://curriculum.code.org/csd-18/

[2] Lucas F. Abreu, Glivia A.R. Barbosa, Ismael S. Silva, and Natalia S. Santos. 2016. Characterizing Software Requirements Elicitation Processes: A Systematic Literature Review. In *Proceedings of the XII Brazilian Symposium on Information Systems: Information Systems in the Cloud Computing Era - Volume 1 (SBSI 2016)*. Brazilian Computer Society, Porto Alegre, Brazil, Brazil, 26:192–26:199. http://dl.acm.org/citation.cfm?id=3021955.3021988

[3] Albert Bandura. 2010. Self-efficacy. *The Corsini encyclopedia of psychology* (2010), 1–3.

[4] R. J. Barnes, D. C. Gause, and E. C. Way. 2008. Teaching the Unknown and the Unknowable in Requirements Engineering Education. In *2008 Requirements Engineering Education and Training*. 30–37. DOI: http://dx.doi.org/10.1109/REET.2008.6

[5] Engin Bozdag. 2013. Bias in Algorithmic Filtering and Personalization. *Ethics and information technology* 15, 3 (2013), 209–227. DOI: http://dx.doi.org/10.1007/s10676-013-9321-6

[6] Ofra Brandes and Michal Armoni. 2019. Using Action Research to Distill Research-Based Segments of Pedagogical Content Knowledge of K-12 Computer Science Teachers. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '19)*. ACM, New York, NY, USA, 485–491. DOI: http://dx.doi.org/10.1145/3304221.3319773 event-place: Aberdeen, Scotland Uk.

[7] Carol B. Brandt, Katherine Cennamo, Sarah Douglas, Mitzi Vernon, Margarita McGrath, and Yolanda Reimer. 2013. A Theoretical Framework for the Studio as a Learning Environment. *International Journal of Technology and Design Education* 23, 2 (2013), 329–348.

[8] John D. Bransford, Ann L. Brown, Rodney R. Cocking, and others. 2000. *How People Learn*. Vol. 11. Washington, DC: National academy press.

[9] Robin Braun, Wayne Brookes, Roger Hadgraft, and Zenon Chaczko. 2019. Assessment Design for Studio-Based Learning. In *Proceedings of the Twenty-First Australasian Computing Education Conference (ACE '19)*. ACM, New York, NY, USA, 106–111. DOI: http://dx.doi.org/10.1145/3286960.3286973

[10] Samantha Breslin and Bimlesh Wadhwa. 2014. Exploring Nuanced Gender Perspectives Within the HCI Community. In *Proceedings of the India HCI 2014 Conference on Human Computer Interaction (IndiaHCI '14)*. ACM, New York, NY, USA, 45:45–45:54. DOI: http://dx.doi.org/10.1145/2676702.2676709

[11] Samantha Breslin and Bimlesh Wadhwa. 2015. Towards a Gender HCI Curriculum. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. ACM, New York, NY, USA, 1091–1096. DOI: http://dx.doi.org/10.1145/2702613.2732923

[12] Tim Brown and others. 2008. Design thinking. *Harvard Business Review* 86, 6 (2008), 84.

[13] Margaret Burnett, Anicia Peters, Charles Hill, and Noha Elarief. 2016. Finding Gender-Inclusiveness Software Issues with GenderMag: A Field Investigation. In *Proceedings of the 2016 CHI conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 2586–2598. DOI: http://dx.doi.org/10.1145/2858036.2858274

[14] Parmit K. Chilana, Rishabh Singh, and Philip J. Guo. 2016. Understanding Conversational Programmers: A Perspective from the Software Industry. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI'16)*. ACM Press, Santa Clara, California, USA, 1462–1472. DOI: http://dx.doi.org/10.1145/2858036.2858323

[15] Elizabeth F. Churchill, Anne Bowser, and Jennifer Preece. 2013. Teaching and Learning Human-computer Interaction: Past, Present, and Future. *interactions* 20, 2 (March 2013), 44–53. DOI: http://dx.doi.org/10.1145/2427076.2427086

[16] Alma Leora Culén. 2015. HCI Education: Innovation, Creativity and Design Thinking. *International Conferences on Advances in Computer-Human Interactions* (2015), 125–130. https://www.duo.uio.no/handle/10852/46215

[17] Alistair D. N. Edwards, Peter Wright, and Helen Petrie. 2006. HCI Education: We are Failing - Why?. In *In Proceedings of HCI Educators Workshop 2006*. 23–24.

[18] Anthony Faiola. 2007. The Design Enterprise: Rethinking the HCI Education Paradigm. *Design Issues* 23, 3 (2007), 30–45. DOI: http://dx.doi.org/https://doi.org/10.1162/desi.2007.23.3.30

[19] Julie Gess-Newsome. 1999. Pedagogical Content Knowledge: An Introduction and Orientation. In *Examining Pedagogical Content Knowledge: The Construct and its Implications for Science Education*, Julie Gess-Newsome and Norman G. Lederman (Eds.). Springer Netherlands, Dordrecht, 3–17. DOI: http://dx.doi.org/10.1007/0-306-47217-1_1

[20] Guiseppe Getto and Fred Beecher. 2016. Toward a Model of UX Education: Training UX Designers within the Academy. *IEEE Transactions on Professional Communication* 59, 2 (2016), 153–164. DOI: http://dx.doi.org/10.1109/TPC.2016.2561139

[21] Susannah Go and Brian Dorn. 2016. Thanks for Sharing: CS Pedagogical Content Knowledge Sharing in Online Environments. In *Proceedings of the 11th Workshop in Primary and Secondary Computing Education on ZZZ - WiPSCE '16*. ACM Press, Münster, Germany, 27–36. DOI:**http://dx.doi.org/10.1145/2978249.2978253**

[22] J. A. Goguen and C. Linde. 1993. Techniques for Requirements Elicitation. In *[1993] Proceedings of the IEEE International Symposium on Requirements Engineering*. 152–164. DOI: **http://dx.doi.org/10.1109/ISRE.1993.324822**

[23] Gabriela Goldschmidt and Anat Litan Sever. 2011. Inspiring Design Ideas with Texts. *Design Studies* 32, 2 (March 2011), 139–155. DOI: **http://dx.doi.org/10.1016/j.destud.2010.09.006**

[24] Joanna Goode and Gail Chapman. 2011. *Exploring Computer Science*. Technical Report. Computer Science Equity Alliance. 296 pages. **http://www.exploringcs.org/wp-content/uploads/2010/08/ExploringComputerScience-v4.0.pdf**

[25] Sukeshini Grandhi. 2015. Educating Ourselves on HCI Education. *interactions* 22, 6 (Oct. 2015), 69–71. DOI: **http://dx.doi.org/10.1145/2834811**

[26] Colin M. Gray. 2014. Evolution of Design Competence in UX Practice. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 1645–1654. DOI: **http://dx.doi.org/10.1145/2556288.2557264**

[27] Colin M. Gray, Austin L. Toombs, and Shad Gross. 2015. Flow of Competence in UX Design Practice. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 3285–3294. DOI: **http://dx.doi.org/10.1145/2702123.2702579**

[28] Tom Gross. 2014. Human-Computer Interaction Education and Diversity. In *Human-Computer Interaction. Theories, Methods, and Tools (Lecture Notes in Computer Science)*, Masaaki Kurosu (Ed.). Springer International Publishing, 187–198.

[29] Jan H. van Driel, Nico Verloop, and Wobbe de Vos. 1998. Developing Science Teachers' Pedagogical Content Knowledge. *Journal of Research in Science Teaching - J RES SCI TEACH* 35 (1998), 673–695. DOI:**http://dx.doi.org/https://doi.org/10.1002/(SICI)1098-2736(199808)35:6<673::AID-TEA5>3.0.CO;2-J**

[30] Rich Halstead-Nussloch and Han Reichgelt. 2013. Teaching HCI in a "Crowded" Computing Curriculum. *J. Comput. Sci. Coll.* 29, 2 (Dec. 2013), 184–190. **http://dl.acm.org/citation.cfm?id=2535418.2535447**

[31] David Hammer and Leema K. Berland. 2014. Confusing Claims for Data: A Critique of Common Practices for Presenting Qualitative Research on Learning. *Journal of the Learning Sciences* 23, 1 (2014), 37–46.

[32] Steve Harrison and Deborah Tatar. 2011. On Methods. *Interactions* 18, 2 (March 2011), 10–11. DOI: **http://dx.doi.org/10.1145/1925820.1925823**

[33] Chenglie Hu. 2016. Can Students Design Software?: The Answer Is More Complex Than You Think. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education (SIGCSE '16)*. ACM, New York, NY, USA, 199–204. DOI: **http://dx.doi.org/10.1145/2839509.2844563**

[34] Peter Hubwieser, Marc Berges, Johannes Magenheim, Niclas Schaper, Kathrin Bröker, Melanie Margaritis, Sigrid Schubert, and Laura Ohrndorf. 2013a. Pedagogical Content Knowledge for Computer Science in German Teacher Education Curricula. In *Proceedings of the 8th Workshop in Primary and Secondary Computing Education (WiPSE '13)*. ACM, New York, NY, USA, 95–103. DOI: **http://dx.doi.org/10.1145/2532748.2532753** event-place: Aarhus, Denmark.

[35] Peter Hubwieser, Johannes Magenheim, Andreas Mühling, and Alexander Ruf. 2013b. Towards a Conceptualization of Pedagogical Content Knowledge for Computer Science. In *Proceedings of the ninth annual international ACM conference on international computing education research (ICER '13)*. ACM, New York, NY, USA, 1–8. DOI: **http://dx.doi.org/10.1145/2493394.2493395**

[36] C. D. Hundhausen, D. Fairbrother, and M. Petre. 2012. An Empirical Study of the "Prototype Walkthrough": A Studio-Based Activity for HCI Education. *ACM Transactions on Computer-Human Interaction (TOCHI)* 19, 4 (Dec. 2012), 26:1–26:36. DOI: **http://dx.doi.org/10.1145/2395131.2395133**

[37] N. H. Ibrahim, J. Surif, A. H. Abdullah, and N. A. S. Sabtu. 2014. Comparison of Pedagogical Content Knowledge between Expert and Novice Lecturers in Teaching and Learning Process. In *2014 International Conference on Teaching and Learning in Computing and Engineering*. 240–246. DOI: **http://dx.doi.org/10.1109/LaTiCE.2014.53**

[38] David G. Jansson and Steven M. Smith. 1991. Design Fixation. *Design Studies* 12, 1 (1991), 3–11.

[39] Yvonne Kao, Katie D'Silva, Aleata Hubbard, Joseph Green, and Kimkinyona Cully. 2018. Applying the Mathematical Work of Teaching Framework to Develop a Computer Science Pedagogical Content Knowledge Assessment. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education (SIGCSE '18)*. ACM, New York, NY, USA, 888–893. DOI: **http://dx.doi.org/10.1145/3159450.3159521**

[40] Cazembe Kennedy and Eileen T. Kraemer. 2018. What Are They Thinking?: Eliciting Student Reasoning About Troublesome Concepts in Introductory Computer Science. In *Proceedings of the 18th Koli Calling*

*International Conference on Computing Education Research (Koli Calling '18)*. ACM, New York, NY, USA, 7:1–7:10. DOI: **http://dx.doi.org/10.1145/3279720.3279728**

[41] Amy J. Ko. 2017. A Three-Year Participant Observation of Software Startup Software Evolution. In *Proceedings of the 39th International Conference on Software Engineering: Software Engineering in Practice Track*. IEEE Press, 3–12. DOI: **http://dx.doi.org/10.1109/ICSE-SEIP.2017.29**

[42] Amy J Ko and Parmit K Chilana. 2011. Design, Discussion, and Dissent in Open Bug Reports. In *Proceedings of the 2011 iConference*. ACM, 106–113. DOI:**http://dx.doi.org/10.1145/1940761.1940776**

[43] Amy J. Ko and Richard E. Ladner. 2016. AccessComputing Promotes Teaching Accessibility. *ACM Inroads* 7, 4 (2016), 65–68. DOI: **http://dx.doi.org/10.1145/2968453**

[44] Aubrey Lawson, Eileen T. Kraemer, S. Megan Che, and Cazembe Kennedy. 2019. A Multi-Level Study of Undergraduate Computer Science Reasoning About Concurrency. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '19)*. ACM, New York, NY, USA, 210–216. DOI: **http://dx.doi.org/10.1145/3304221.3319763** event-place: Aberdeen, Scotland Uk.

[45] Sarah Lewthwaite and David Sloan. 2016. Exploring Pedagogical Culture for Accessibility Education in Computing Science. In *Proceedings of the 13th Web for All Conference (W4A '16)*. ACM, New York, NY, USA, 3:1–3:4. DOI: **http://dx.doi.org/10.1145/2899475.2899490**

[46] Paul Luo Li, Amy J Ko, and Andrew Begel. 2017. Cross-Disciplinary Perspectives on Collaborations with Software Engineers. In *Cooperative and Human Aspects of Software Engineering (CHASE), 2017 IEEE/ACM 10th International Workshop on*. IEEE, 2–8. DOI: **http://dx.doi.org/10.1109/CHASE.2017.3**

[47] Neomi Liberman, Yifat Ben-David Kolikant, and Catriel Beeri. 2009. In-service Teachers Learning of a New Paradigm: A Case Study. In *Proceedings of the Fifth International Workshop on Computing Education Research Workshop (ICER '09)*. ACM, New York, NY, USA, 43–50. DOI: **http://dx.doi.org/10.1145/1584322.1584329**

[48] D. Scott McCrickard, C. M. Chewar, and Jacob Somervell. 2004. Design, Science, and Engineering Topics?: Teaching HCI with a Unified Method. In *Proceedings of the 35th SIGCSE technical symposium on computer science education (SIGCSE '04)*. ACM, New York, NY, USA, 31–35. DOI: **http://dx.doi.org/10.1145/971300.971314**

[49] Tom McKlin, Taneisha Lee, Dana Wanzer, Brian Magerko, Doug Edwards, Sabrina Grossman, Emily Bryans, and Jason Freeman. 2019. Accounting for Pedagogical Content Knowledge in a Theory of Change Analysis. In *Proceedings of the 2019 ACM Conference on International Computing Education Research (ICER '19)*. ACM, New York, NY, USA, 157–165. DOI: **http://dx.doi.org/10.1145/3291279.3339412**

[50] Punyashloke Mishra and Matthew J. Koehler. 2006. Technological Pedagogical Content Knowledge: A Framework for Teacher Knowledge. DOI:**http://dx.doi.org/10.1111/j.1467-9620.2006.00684.x**

[51] Eduardo Navas, Owen Gallagher, and Borrough, xtine. 2014. *The Routledge companion to Remix studies*. Routledge.

[52] Laura Ohrndorf and Sigrid Schubert. 2013. Measurement of Pedagogical Content Knowledge: Students' Knowledge and Conceptions. In *Proceedings of the 8th Workshop in Primary and Secondary Computing Education (WiPSE '13)*. ACM, New York, NY, USA, 104–107. DOI: **http://dx.doi.org/10.1145/2532748.2532758**

[53] Alannah Oleson, Christopher Mendez, Zoe Steine-Hanson, Claudia Hilderbrand, Christopher Perdriau, Margaret Burnett, and Amy J. Ko. 2018. Pedagogical Content Knowledge for Teaching Inclusive Design. In *Proceedings of the 2018 ACM Conference on International Computing Education Research (ICER '18)*. ACM, New York, NY, USA, 69–77. DOI: **http://dx.doi.org/10.1145/3230977.3230998**

[54] Hye Park and Seda McKilligan. 2018. A Systematic Literature Review for Human-Computer Interaction and Design Thinking Process Integration. In *Design, User Experience, and Usability: Theory and Practice (Lecture Notes in Computer Science)*, Aaron Marcus and Wentao Wang (Eds.). Springer International Publishing, 725–740.

[55] Michael Quinn Patton. 2014. *Qualitative Research & Evaluation Methods: Integrating Theory and Practice*. SAGE Publications.

[56] Marian Petre and Andre Van Der Hoek. 2013. *Software Designers in Action: A Human-Centric Look at Design Work* (1st ed.). Chapman & Hall/CRC.

[57] Henry Petroski. 2006. *Success Through Failure: The Paradox of Design*. Princeton University Press.

[58] Peter Pirolli and Stuart Card. 1995. Information Foraging in Information Access Environments. In *Chi*, Vol. 95. 51–58.

[59] Chris Proctor, Maxwell Bigman, and Paulo Blikstein. 2019. Defining and Designing Computer Science Education in a K12 Public School District. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19)*. ACM, New York, NY, USA, 314–320. DOI: **http://dx.doi.org/10.1145/3287324.3287440**

[60] Yolanda Jacobs Reimer and Sarah A. Douglas. 2003. Teaching HCI Design with the Studio Approach. *Computer Science Education* 13, 3 (2003), 191–205.

[61] Horst W J Rittel. 1987. The Reasoning of Designers. (1987), 12.

[62] Luciana Salgado, Roberto Pereira, and Isabela Gasparini. 2015. Cultural Issues in HCI: Challenges and Opportunities. In *Human-Computer Interaction: Design and Evaluation (Lecture Notes in Computer Science)*, Masaaki Kurosu (Ed.). Springer International Publishing, 60–70.

[63] Lee Shulman. 1987. Knowledge and Teaching: Foundations of the New Reform. *Harvard Educational Review* 57, 1 (1987), 1–23. DOI: **http://dx.doi.org/https://doi.org/10.17763/haer.57.1.j463w79r56455411**

[64] Martin A. Siegel and Erik Stolterman. 2008. Metamorphosis: Transforming Non-Designers into Designers, Vol. 378. Sheffield, UK: Sheffield Hallam University, 1–13.

[65] Charles Thevathayan and Margaret Hamilton. 2017. Imparting Software Engineering Design Skills. In *Proceedings of the Nineteenth Australasian Computing Education Conference (ACE '17)*. ACM, New York, NY, USA, 95–102. DOI: **http://dx.doi.org/10.1145/3013499.3013511**

[66] Anna Vallgarda and Ylva Fernaeus. 2015. Interaction Design As a Bricolage Practice. In *Proceedings of the Ninth International Conference on Tangible, Embedded, and Embodied Interaction (TEI '15)*. ACM, New York, NY, USA, 173–180. DOI: **http://dx.doi.org/10.1145/2677199.2680594**

[67] Rebecca Vivian and Katrina Falkner. 2019. Identifying Teachers' Technological Pedagogical Content Knowledge for Computer Science in the Primary Years. In *Proceedings of the 2019 ACM Conference on International Computing Education Research (ICER '19)*. ACM, New York, NY, USA, 147–155. DOI: **http://dx.doi.org/10.1145/3291279.3339410**

[68] Lauren Wilcox, Betsy DiSalvo, Dick Henneman, and Qiaosi Wang. 2019. Design in the HCI Classroom: Setting a Research Agenda. In *Proceedings of the 2019 on Designing Interactive Systems Conference (DIS '19)*. ACM, New York, NY, USA, 871–883. DOI: **http://dx.doi.org/10.1145/3322276.3322381**

[69] Aman Yadav and Marc Berges. 2019. Computer Science Pedagogical Content Knowledge: Characterizing Teacher Performance. *ACM Trans. Comput. Educ.* 19, 3 (May 2019), 29:1–29:24. DOI: **http://dx.doi.org/10.1145/3303770**

[70] Helen Z. Zhang, Charles Xie, and Saeid Nourian. 2018. Are their Designs Iterative or Fixated? Investigating Design Patterns from Student Digital Footprints in Computer-Aided Design Software. *International Journal of Technology and Design Education* 28, 3 (Sept. 2018), 819–841. DOI: **http://dx.doi.org/10.1007/s10798-017-9408-1**