CIDER: A Method to Teach Practical Critical Software Design Skills

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

ABSTRACT

Computing students should learn to think critically about the software they design, but prior work in computing and human-computer interaction (HCI) education suggests that developing these skills can be a difficult task. To address some of these challenges, we developed the CIDER (*Critique*, *Imagine*, *Design*, *Expand*, *Repeat*) assumption elicitation technique. CIDER is an educational analytical design evaluation method to help computing students recognize and respond to exclusionary design biases in software user interfaces (UIs), helping them critically consider how their design decisions could differentially impact various user groups and how they might design more inclusively. The CIDER technique is grounded in prior HCI and computing education work and was specifically designed to target learning challenges computing students face in HCI design courses. We hope to evaluate the efficacy of CIDER in a variety of learning contexts, contributing to HCI and computing education foundations for teaching and learning critical software design skills.

CCS CONCEPTS

• **Social and professional topics** → **Computing education**; • **Human-centered computing** → **HCI design and evaluation methods**; *Interaction design process and methods*; • **Software and its engineering** → *Designing software.*

KEYWORDS

HCI education, inclusive design methods, assumption elicitation

ACM Reference Format:
Alannah Oleson. 2022. CIDER: A Method to Teach Practical Critical Software Design Skills. In *Proceedings of the 2022 ACM Conference on International Computing Education Research V.2 (ICER 2022), August 7–11, 2022, Lugano and Virtual Event, Switzerland.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3501709.3544295

1 MOTIVATION AND KEY IDEAS

Software can help us achieve great things, but it can also perpetuate biases, reinforce harmful sociocultural norms, and exclude people from using it due to access barriers. User interfaces (UIs) are a prominent place for design tensions to manifest, since they both support and constrain the interactions people can have with technology. If left unaddressed, biases present during interface design processes can result in software that only works well for some kinds of users, often disproportionately excluding users from minoritized groups. Recent calls for more critical computing education [8, 13] highlight the need for pedagogical methods and strategies that help computing students learn to navigate these tensions. My research works toward a future where computing students can critically reflect on their design designs, understand the real-world implications of their choices, and create more inclusive, accessible, and ethical software interfaces. Specific questions guiding my work include:

(1) What **learning difficulties** do computing students face when trying to learn UI design principles?
(2) What might **a pedagogical technique** to help students think critically about software UI design decisions look like?
(3) How might this technique be **used and adapted** within different HCI and UI design learning contexts?

2 PRIOR WORK

Though HCI education research is a relatively young discipline [12], there is a quickly growing focus on teaching computing students to critically consider the implications of the technologies they design. Prior work has explored how we might best teach and learn topics like tech ethics [4] and software accessibility principles [6] within computing spaces. Even so, several challenges arise when teaching and learning critical UI design skills in computing-centric contexts. Educators may not feel they have the expertise, time, or organizational supports they need to properly teach accessible or inclusive UI design. Computing students can also struggle to learn basic design principles, much less critical approaches to UI design [7]. My research targets five student learning challenges in particular: motivating critical UI design [2]; connecting UI design features to assumptions [5]; designing for diversity [3]; moving from abstract understandings to concrete critical UI design actions [1]; and avoiding stereotyping [12].

3 RESEARCH APPROACH & METHODS

3.1 Surfacing Student Learning Difficulties in HCI & UI Design Courses (RQ1)

To develop effective pedagogical techniques for critical UI design education, we must also understand common learning difficulties and how to mitigate them. I led a series of surveys and semi-structured interviews with post-secondary computing students to gather information about the concepts and activities that students struggled with in their introductory HCI and UI design courses. Through inductive thematic analyses, we surfaced an initial set of learning difficulties, which we then presented to and discussed with post-secondary and professional HCI/UI design educators for verification and triangulation. Our final set of learning difficulties

ICER 2022, August 7–11, 2022, Lugano and Virtual Event, Switzerland

ACM ISBN 978-1-4503-9195-5/22/08.
https://doi.org/10.1145/3501709.3544295

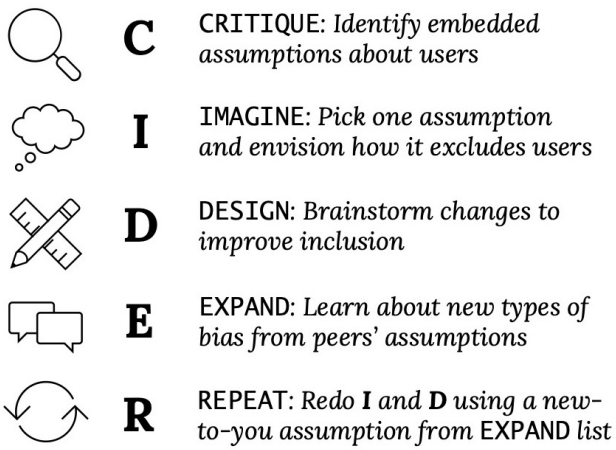

Figure 1: The CIDER assumption elicitation technique. The educator chooses a single artifact for all students to critique for each activity.

included challenges around the mechanics of design work, project management skills, the "wicked" nature of design problems, and distorted perspectives on the design process, many of which might lead to UI biases. More details about this study can be found in our 2020 ACM CHI paper [9]. If these learning challenges are left unaddressed, computing students may not gain the foundational knowledge needed to critically consider their UI design decisions or practice inclusive software design.

3.2 Developing the CIDER Technique (RQ2)

I created the CIDER (*Critique*, *Imagine*, *Design*, *Expand*, *Repeat*) technique to help computing students recognize and respond to exclusionary UI design biases (Figure 1). The novelty of this technique is twofold: CIDER is the one of the first educational UI design techniques to explicitly use *assumptions about users* as its basis, tying together design features, design decisions, and the impacts designers' conceptions of users have on the inclusiveness of a UI. In addition, the stages of the CIDER technique support learners in building concrete critical understandings of design bias and inclusion in actionable ways that they can apply to future design practice. As an analytical design evaluation method, CIDER facilitates critique of already-existing UIs, which can include real-world technology or students' own UI prototypes that they create for a course. A mixed-method case study evaluating CIDER's efficacy in a post-secondary introductory UI design course found that the technique helped significantly increase students' self-efficacy for inclusive UI design, expand students' understandings of design biases, and even had long-lasting impacts on students' UI design approaches. More about this study can be found in our forthcoming 2022 ACM TOCHI paper [10].

3.3 Evaluating CIDER Across Learning Contexts and Creating Educator Resources (RQ3)

To understand how CIDER might work within different educational contexts, my next steps will explore how other educators use and adapt CIDER within their own courses. This work will ideally contribute (1) empirical understandings of how some educators teach CIDER within different learning contexts and (2) a set of educator resources. Through a set of comparative case studies [11], I will detail educators' experiences teaching the CIDER technique in their courses, collaborating with post-secondary HCI and UI design educators whose learning contexts vary across attributes such as:

- Type of program (CS, HCI, UX design, etc.)
- Educator expertise with critical UI design concepts
- Student prior exposure to critical UI design concepts

I plan to conduct semi-structured interviews with these educators both before (to understand their class preparation strategies) and after (to understand students' reactions and any emergent teaching challenges) they use CIDER in their courses, as well as in-class observations or student interviews where feasible. These insights will then inform a set of publicly-available educator resources. We hope that educators who participate and use these resources will be able to successfully help their students develop critical understandings of how UI designs perpetuate inequality and actionable skills to create more inclusive, accessible, and ethical software.

REFERENCES

[1] Shruthi Sai Chivukula, Aiza Hasib, Ziqing Li, Jingle Chen, and Colin M. Gray. 2021. *Identity Claims That Underlie Ethical Awareness and Action.* Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3411764.3445375

[2] 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. https://doi.org/10.1145/2427076.2427086

[3] Sasha Costanza-Chock. 2020. *Design Justice: Community-led practices to build the worlds we need.* MIT Press.

[4] Casey Fiesler, Natalie Garrett, and Nathan Beard. 2020. What Do We Teach When We Teach Tech Ethics? A Syllabi Analysis. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20).* Association for Computing Machinery, New York, NY, USA, 289–295. https://doi.org/10.1145/3328778.3366825

[5] Batya Friedman and Helen Nissenbaum. 1996. Bias in computer systems. *ACM Transactions on Information Systems* 14, 3 (July 1996), 330–347. https://doi.org/10.1145/230538.230561

[6] Lin Jia, Yasmine N. Elglaly, Catherine M. Baker, and Kristen Shinohara. 2021. Infusing Accessibility into Programming Courses. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems.* Number 231. Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/3411763.3451625

[7] Saba Kawas, Laura Vonessen, and Amy J. Ko. 2019. Teaching Accessibility: A Design Exploration of Faculty Professional Development at Scale. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education (SIGCSE '19).* Association for Computing Machinery, New York, NY, USA, 983–989. https://doi.org/10.1145/3287324.3287399

[8] Amy J. Ko, Alannah Oleson, Mara Kirdani-Ryan, Yim Register, Benjamin Xie, Mina Tari, Matthew Davidson, Stefania Druga, and Dastyni Loksa. 2020. It is time for more critical CS education. *Commun. ACM* 63, 11 (Oct. 2020), 31–33. https://doi.org/10.1145/3424000

[9] Alannah Oleson, Meron Solomon, and Amy J. Ko. 2020. Computing Students' Learning Difficulties in HCI Education. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20).* Association for Computing Machinery, Honolulu, HI, USA, 1–14. https://doi.org/10.1145/3313831.3376149

[10] Alannah Oleson, Meron Solomon, Christopher Perdriau, and Amy J. Ko. 2022. Teaching Inclusive Design Skills with the CIDER Assumption Elicitation Technique. *ACM Transactions on Computer-Human Interaction (TOCHI)* (2022).

[11] Michael Quinn Patton. 2014. *Qualitative Research & Evaluation Methods: Integrating Theory and Practice.* SAGE Publications.
[12] 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. https://doi.org/10.1145/3322276.3322381
[13] Aman Yadav, Marie Heath, and Anne Drew Hu. 2022. Toward justice in computer science through community, criticality, and citizenship. *Commun. ACM* 65, 5 (2022), 42–44.