frontiers | Frontiers in Computer Science

ORIGINAL RESEARCH
published: 26 May 2022
doi: 10.3389/fcomp.2022.813889

An Open-Ended Blended Approach to Teaching Interaction Designers to Code

Kazjon Grace, Brittany Klaassens, Liam Bray and Alex Elton-Pym*

Design Lab, School of Architecture, Design and Planning, The University of Sydney, Sydney, NSW, Australia

This article reports on a three and a half year design-led project investigating the use of open-ended learning to teach programming to students of interaction design. Our hypothesis is that a more open-ended approach to teaching programming, characterized by both creativity and self-reflection, would improve learning outcomes among our cohort of aspiring HCI practitioners. The objective of our design-led action research was to determine how to effectively embed open-endedness, student-led teaching, and self-reflection into an online programming class. Each of these notions has been studied separately before, but there is a dearth of published work into their actual design and implementation in practice. In service of that objective we present our contribution in two parts: a qualitatively-derived understanding of student attitudes toward open-ended blended learning, as well as a matching set of design principles for future open-ended HCI education. The project was motivated by a search for better educational outcomes, both in terms of student coding self-efficacy and quantitative metrics of cohort performance (e.g., failure rates). The first year programming course within our interaction design-focussed Bachelors program has had the highest failure rate of any core unit for over a decade. Unfortunately, the COVID-19 pandemic confounded any year-to-year quantitative comparison of the learning efficacy of our successive prototypes. There is simply no way to fairly compare the experiences of pre-pandemic and pandemic-affected student cohorts. However, the experience of teaching this material in face-to-face, fully online, and hybrid modalities throughout the pandemic has aided our qualitative exploration of why open-ended learning helps some students but seems to harm others. Through three sets of student interviews, platform data, and insights gained from both the instructional and platform design process, we show that open-ended learning can empower students, but can also exacerbate fears and anxieties around inadequacy and failure. Through seven semesters of iterating on our designs, interviewing students and reflecting on our interventions, we've developed a set of classroom-validated design principles for teaching programming to HCI students without strong computational backgrounds.

Keywords: open-ended learning, student-led teaching, blended learning, interaction design, programming education, creative coding

OPEN ACCESS

Edited by:
Karin Slegers,
Zuyd University of Applied Sciences, Netherlands

Reviewed by:
Jan de Wit,
Tilburg University, Netherlands
Christos Troussas,
University of West Attica, Greece

****Correspondence:***
Kazjon Grace
kazjon.grace@sydney.edu.au

Specialty section:
This article was submitted to Digital Education, a section of the journal Frontiers in Computer Science

Received: *12 November 2021*
Accepted: *12 April 2022*
Published: *26 May 2022*

Citation:
Grace K, Klaassens B, Bray L and Elton-Pym A (2022) An Open-Ended Blended Approach to Teaching Interaction Designers to Code. Front. Comput. Sci. 4:813889. doi: 10.3389/fcomp.2022.813889

1. INTRODUCTION

Programming skills are a critical part of any modern interaction designer's education. Computational thinking and digital prototyping skills, both of which require some level of programming proficiency, are increasingly important for designing all manner of products and services. In an era of cross-functional teams operating in demo-or-die environments, the notion of an exclusively human-focussed HCI practitioner seems ever more obsolete.

Despite this, a substantial fraction of the interaction design students in our undergraduate program, the longest-running HCI-focussed design course in Australia, consider programming to be one of their biggest struggles. This low coding self-efficacy (Ramalingam et al., 2004) is associated with students perceiving themselves as "not a coder", or "just not able to think that way". This paper synthesizes what we have learned from a 3-year project to redesign the introductory programming subject within our design degree.

Educating emerging practitioners of human-centered design to also be competent programmers is not straightforward: design and software development require very different metacognitive strategies, particularly in how they handle ambiguity and abstraction. Computational thinking teaches how to resolve ambiguity using hierarchies of abstraction (Wing, 2008). By contrast, design thinking teaches acceptance of ambiguity and how to instead value and work with multiple competing perspectives (Tversky, 2015). It's not a stretch to see how the students each discipline tends to attract would favor one approach but struggle with the other. The human-centered design aspects of an HCI education have classically been confronting to traditional STEM cohorts (Cooper, 1999), and the opposite is also true: the system-centered nature of programming is confronting to students of design.

To effectively educate modern HCI practitioners, therefore, means to produce graduates equally adept at both the human and the technical. To do so will require—perhaps fittingly—both technology-led and design-led innovations, but also a greater understanding of the student experience of such a program than we have today. To that aim, this paper reports on a 3 year design-led project to explore how to more effectively introduce interaction design, HCI and user experience design students to programming.

Our approach combined creative coding (i.e., programming as a creative medium) (Reas and Fry, 2006) with open-ended learning (i.e., giving students greater agency in shaping their learning trajectories) (Hannafin, 1995) and student-centered learning (i.e., letting students play an active role in teaching) (De Volder et al., 1985). Specifically, we wanted to frame programming skills around small open-ended "making" activities and then invite students to create these activities for their peers. Our hypothesis was that this would create an environment where the flexibility and expressive capacity of programming was emphasized, appealing to students of design. Furthermore, we needed this approach to scale to classes of up to 500 students and be teachable by staff with a wide range of expertise, so we adopted a blended learning approach—a far more niche choice in 2018 than it is today!

Our design-led methodology was necessitated by the well-known challenge of scaling educational innovations from the laboratory to the curriculum (Cohen and Ball, 2007). Evidence-based practices, particularly those of a technological nature, are notoriously difficult to implement (Klingner et al., 2013), facing obstacles from students, educators, administrators and policy-makers alike. As an alternative to tarring any of those stakeholders as particularly ornery, "design-based implementation" approaches (Penuel et al., 2011) have been adopted as a way to bring stakeholders into the process of deciding how, when and where educational innovations should be applied. Familiar to any practitioner of HCI, this approach amounts to applying human-centered design to the process of implementing educational innovations. This paper presents research in this tradition of design-led implementation of educational innovations, combining interface design, service design, and learning design into a multi-year collaboration between researchers and educators. Critically, that means this research thus does not propose or evaluate any original technological innovations, but instead contributes classroom-tested understanding and principles to guide future similar implementation challenges.

Driven by this approach we engaged in iterative prototyping, evaluation and refinement, deploying our first prototype in 2018, running our first full course using in 2019, and teaching 400+ students each year 2019–2021. Each year we took the best parts of what worked and refined them into a new version of our open-ended blended learning platform. In 2021 students submitted over 5,800 responses to our online "challenges," which are open-ended making-focused learning activities. The course has increased student satisfaction and been enthusiastically well-received by the 20+ teaching team.

The project also, by virtue of featuring a blended learning platform that was already deployed at scale in 2020, collected insights on how our students navigated the educational disruption of the COVID pandemic. At the university where this study was conducted this disruption was severe: one semester transitioned to remote learning in its fourth week, one semester was conducted entirely online, and a third semester was run "hybrid," with small (¡20 person) face-to-face classes for the (approximately half) students who were able to get to campus. In one sense this disruption has made it impossible to report on the year-on-year quantitative improvements in student satisfaction over the life of the project. However, it also offered an opportunity for us to expand our exploration of student attitudes to cover a broad range of contexts. Given this opportunity, our research contributions can be expressed as follows:

a) an understanding of how open-ended blended learning impacted the experience of designers learning programming, including their attitudes toward self-directed and student-led learning, derived from a rigorous qualitative meta-analysis, and

b) a set of classroom-validated design principles for effective open-ended programming education, particularly for cohorts without a strong computational background.

We reflect on each major revision of our "Creative Coding Challenges" platform (CCCs), its focus and goals, the way we evaluated its success, and the insights gained from it. We then provide a thematic meta-analysis of the 63 student interviews conducted over the life of the project. We then derive a set of recommendations for how to teach programming to designers in future.

2. BACKGROUND

Open-ended and student-led pedagogies are particularly applicable to teaching designers due to the existing prevalence of collaborative, project-based learning in design (Wang, 2010). Our blended learning focus was by necessity: a technology platform was needed to implement our ideas about student-led teaching at the scale our courses required. To explain how we arrived at these notions, we present the four research fields in which this project is situated: programming education, design education, open-ended learning, and blended learning.

2.1. Programming Education

Our 2018 prototype was inspired by another successful multi-year experiment in online peer learning for creative coding (Carvalho et al., 2014). The motivation behind that platform was to explore the peer learning aspects of learning programming in a web context (Carvalho and Saunders, 2018). Another key idea in both projects is that teaching creative coding is a more effective and accessible method compared to a traditional "plain" programming course.

It is well established that learning to program is very difficult (Gomes and Mendes, 2007), although directly saying as much to students has been shown to disadvantage students from underrepresented groups (Becker, 2021). Introducing students to algorithmic thinking and complex problem solving is a challenging task. Students must also simultaneously learn complex syntax with high levels of abstraction, in languages typically not designed to be a student's first language. Introductory programming courses typically aim to teach programming generally, but must by necessity focus on a single language, a confusing distinction for many students.

For educators, it is often difficult to personalize lessons due to large class sizes in introductory courses. Learning programming well-known to require significant individualized feedback based on each student's progress, which becomes challenging as classes and courses scale up (McBroom et al., 2020). There is also the challenge of students' coding self-efficacy, which is associated with prior exposure (and thus typically lower in non-CS cohorts) as well as being linked to programming course outcomes (Ramalingam et al., 2004). If coding self-efficacy is a high predictor of coding success, and many HCI and design students are not from the kind of backgrounds where they have had a high exposure to programming before attending university, how can we best improve it in our courses?

Difficulty learning programming is linked to a nexus of highly related motivational, interest, and identity factors (Jenkins, 2002). This is particularly common in the increasing number of contexts, like our own, where introductory programming is a core component of non-computer-science courses (Guzdial, 2003). For many students in these contexts, the completion of the subject may be seen as an inconvenient obstacle to completing their degree: they are less likely to exhibit the critical intrinsic motivation to learn that programming so benefits from. These issues are known to be especially prominent in non-white, non-male, non-cisgendered, non-heterosexual, and non-native English speaking students students, as well as students with disabilities (Peckham et al., 2007; Charleston et al., 2014; Kargarmoakhar et al., 2020).

"Creative coding" is a computing pedagogy that offers some solutions to these problems. In creative coding approaches, programming is presented as a medium for creative (often visual) expression (Reas and Fry, 2006), providing a simple means for highly abstract concepts to be represented visually. This can often lead to the "flow" of a complex program—a common sticking point for students—being clearer and more easy to manipulate. Many languages for creative coding are specifically designed for people without strong technical backgrounds, such as the Processing family of languages (Reas and Fry, 2007), which are designed for artists and educators. The visual and interactive nature of creative coding provides instant feedback to students on what their program is doing, as the code typically revolves around drawing to the screen. In addition to being more popular among certain groups of non-traditional programming students (Guzdial, 2003, 2009; Greenberg et al., 2012), creative approaches to code are perfect for our HCI audience: our students identify as designers, and this approach lets them see code as a medium for design.

2.2. Design Education

Design education finds its foundations in the "atelier" or master-and-apprentices model common in the fine arts until the late nineteenth century. In this educational model a well-known artist would coordinate a small group of assistants to produce creative works, with the assistants learning on the job and then, ideally, going off to start their own practices. This evolved into what is commonly known as the "studio model," the cornerstone of architectural and industrial design education. Studio-based teaching shifts the focus of the class toward the students, as autonomous and curious practitioners-in-training. Structuring learning in this way is supported by research into design cognition, such as the notion of "reflective practice" (Schön, 1979, 1987). The reflective practitioner is one who can think and re-think their plans while acting, and thus can respond to the uncertainty, uniqueness and conflict involved in the situations in which designers (and other professionals) practice. Important to Schön's argument is that the knowledge required to know how to act is learned through intentional and critical practice, i.e., the repeated act of placing one's self in a situation in which they are required to make design decisions. Studio-based education is the pedagogical formalization of that notion, with a focus on repeated learning-by-doing, interspersed with feedback and reflection. The goal of design studios is to building the critical and tacit knowledge required to become a reflective design practitioner (Kuhn, 2001).

In the last few decades studio-based education has found purchase beyond the traditional design domains where it was dominant throughout the twentieth century. Successful applications have been applied throughout STEM (Kuhn, 1998; Adams et al., 2003; Reimer and Douglas, 2003; Carvalho and Saunders, 2018), in part because of the expanding attention on design thinking as a general model for solving under-specified problems involving people (Cross, 2011). As the scope of human-centered design has expanded to include interactive products and services of all kinds, the design studio has followed, and now forms a core component of design programs with focusses as diverse as game design, medical device design, information visualization, and visual communication.

While there is much potential in this approach, the design studio model is uniquely ill-suited to the modern university context of ever-expanding classes and ever-shrinking teaching budgets. Design studios are extremely expensive in terms of face-to-face time, and require a high level of educator expertise, not just in the design domain in question but in the practice of studio teaching itself. It does not, at least in its original conceptualization, permit easy scaling nor lend itself to educational technologies. This paper reflects on how elements of the studio model that would be familiar to our students—open-ended learning, self-directed learning, and peer learning in particular—might be applied to teach programming in a scalable, blended way.

2.3. Open-Ended and Student-Led Teaching

Open-ended learning, which has its roots in constructivism, refers to "processes wherein the intents and purposes of the individual are uniquely established and pursued" (Hannafin, 1995). It involves individual students having autonomy in determining what to learn, and how they learn it. This definition is by necessity broad, as the very essence of this approach requires that there is not one correct way. Arguably the main difference between open-ended learning and more traditional directed methods, is that students are at the center of the learning process (Land and Hannafin, 1997).

Open-ended learning is based on premise that effective learning involves fitting new information together with what students already know (Bada and Olusegun, 2015). It is also related to the idea that learning is affected by context, as well as by students' beliefs and attitudes (Bereiter, 1994). This paradigm views teaching as a process that helps learners to create knowledge through interactive, engaging and authentic learning experiences. Taking inspiration from constructivist theories, Chickering and Gamson (1987) published ta well-known set of principles for effective open-ended learning environments in higher education. They included the encouragement of both student-student and teacher-student co-operation, active learning, prompt feedback, high expectations, and a respect for functional diversity.

These open-ended principles have been integral in drawing attention to good teaching and learning practices (Vaughan et al., 2013), although primarily they have been used in face-to-face contexts. In HCI education specifically, these principles have been manifested through studio-style teaching which emphasizes student autonomy, collaboration, creativity, curiosity, and student-led feedback (Reimer and Douglas, 2003). On the other hand, programming education tends to focus on learning transferable skills through various kinds of problem-solving (Carbone and Sheard, 2002; Rajaravivarma, 2005).

Open-ended learning has been adopted in programming education (Carbone and Sheard, 2002; Blikstein, 2011), typically with a focus on computer science and software engineering students. Collaboration is often a key part of open-ended learning, and existing research has sought ways both pedagogical (Emara et al., 2017) and technological (Troussas et al., 2020; Emara et al., 2021) to support and sustain collaboration amongst teams of open-ended learners. Computational approaches to analyzing and grouping students, however, have largely been studied in the context of tasks in STEM with clear right answers: assessing collaboration styles and assigning appropriate tasks in creative design contexts is significantly more challenging. We are not aware of significant research to date on how these open-ended methods can be applied when teaching programming to non-STEM audiences, such as to students of interaction design. Our design students have existing familiarities with open-ended and collaborative ways of learning, and it's possible that their expectations and outcomes will differ.

Student-led or peer learning is a closely related strain of experimental pedagogy to open-endedness. In student-led teaching, the design and/or conduct of some learning activities is given over to one or more students, who lead their peers in (usually collaborative) learning (De Volder et al., 1985). This has been shown to increase learner engagement and achievement in some settings (Casteel and Bridges, 2007), particularly when involving students from under-represented groups (Rohrbeck et al., 2003). Student-led teaching can be demanding (Robinson and Schaible, 1995), but it benefits both the student-teacher and the student-learners. The "protégé effect" is the common name for how teaching something forces thinking critically about one's own understanding of it (Chase et al., 2009). Peer learning can be considered an extension of active learning, in which learning-by-teaching is an extreme form of learning-by-doing.

It should be noted that open-ended, student-led and self-regulated approaches to learning are well known not to always work for all kinds of students all the time (Land, 2000). Students sometimes retain prior misconceptions, fail to sufficiently monitor and self-regulate, or engage only shallowly, without analysis or self-reflection. Land refers to this as the "metacognitive knowledge dilemma," the problem of monitoring learning in the absence of domain knowledge. It's a fundamental principle of constructivist approaches to learning that effective educators extend students' capability by framing new knowledge in ways compatible with those students' existing understanding (Vygotsky, 1930–1934/1978). From that perspective it is then unsurprising that removing the educator from the process can lead to worse outcomes for some students, particularly those who require more support. It has been recommended that open-ended learning environments, particularly those rich in content, incorporate organizing frameworks to help guide learners'

metacognitive strategies and make their progress through the content explicit (Land, 2000).

2.4. Blended Learning

Blended learning is an innovative pedagogical approach to learning that seeks to use technology to improve the differentiation of instruction according to student needs and the facilitation of student interaction (Huynh et al., 2016). A common misconception with blended learning is that it is the transposition of physical classes transferred to a digital space. When misapplied, blended learning can leave students unengaged and isolated (Logan, 2015). According to Paniagua and Istance (2018), a blended learning environment utilizes technology to improve certain teaching and learning practices in order to focus more time on making the physical classroom more interactive, and the digital classroom more connected. Blended learning can make rapid, unscheduled shifts in the format of teaching (such as in response to public health orders instituted during a pandemic) simpler to facilitate (Nielsen, 2012).

Horn and Staker (2014) outline that in order for any learning environment to be effective, it must be student-centered. Student-centered learning is closely related to open-ended learning (see Section 2.3) and is defined as an instructional approach in which students influence the content, activities and pace of learning (Froyd and Simpson, 2008). This is consistent with constructivist approaches to learning, i.e., where students have the skills and opportunities to learn independently and from one another (Wilson and Lowry, 2000). Technologically facilitated flexibility in the time, place and pace of learning allows students more opportunities to influence the way their learning happens (Nassrallah et al., 2018).

Blended learning is often discussed in the context of facilitating active learning, learning activities that encourage students to "seek new information, organize it in a way that is meaningful, and have the chance to explain it to others" (Bransford et al., 2000). This form of instruction emphasizes interactions with peers and tutors, with a focus on applying knowledge and receiving rapid feedback (Freeman et al., 2014). Placing students at the center of learning promotes a learning environment that is more amenable to the metacognitive development necessary for students to become independent critical thinkers (Bransford et al., 2000). Critical thinking skills are crucial in the development of both successful programmers and designers (Jeong, 2017), making their encouragement central to quality HCI education.

3. MATERIALS AND METHODS

We present a reflective account of our iterative design process over the course of the project, supplemented by a summative thematic meta-analysis of the student experience as observed through over 60 interviews. The project consisted of three cycles of prototyping, evaluating and reflecting on our intervention, with each cycle yielding its own insights that may inform future projects. Education researchers might be most comfortable framing this iterative approach as action research (Armstrong, 2019), with each cycle being an opportunity to act with and then observe the students and teaching team. By contrast, HCI researchers might conceive of it as research-through-design (Zimmerman et al., 2007), with each cycle being an opportunity to iteratively refine and reflect on the intervention itself.

The truth, as with all interdisciplinary research, is likely in the middle somewhere—we contend that both apply equally here. To that end we describe our process as three iterative cycles (in the tradition of action research) of each of three processes: prototyping, evaluating, and reflecting, although (in the tradition of research-through-design) these are never as linear or separable as they might at-first seem. Each cycle contains one or more classroom-delivered prototypes, designed to build toward the project's goals, one or more periods of rich student-centered evaluation (typically thematic analysis of interviews and/or content analysis of platform data), and a series of reflections on the efficacy and implications of those prototypes and their analysis, in the tradition of reflective practice (Schön, 1979). Activities within each cycle typically occurred in parallel, and were undertaken by our interdisciplinary team of researchers and educators, including some graduate students who were both. Each cycle spanned approximately a year, or two semester-long iterations of our design programming course.

The design insights gained from each cycle of the project's life come from reflections of the educators, system designers and researchers—three groups that have significant overlap. Since 2018 the project has been the focal point for five undergraduate honors theses, each a 1-year interaction design project exploring and building on an aspect of the CCCs platform. All of those honors students have also been part of the teaching team, forming a unique coupling between teaching practice and research. HCI is one of the few domains where it's possible for there to be so much overlap between the developers of an educational technology, the front-line educators using it, and the researchers evaluating it. That integration was a significant strength for the CCCs project and one that we recommend that future HCI education innovations adopt.

The contributions presented in this paper are derived from a union of practice-based learnings (grounded in the experience of making and using the CCCs platform in the tradition of research-through-design) (Zimmerman et al., 2007), with ethnographic data (from a meta-analysis of over 60 interviews with students across the project's life). From these data we synthesize principles for how best to design for open-ended learning among HCI students in future.

3.1. Overview of the Creative Coding Challenges System

Running from 2018 to 2021, the CCCs project unfolded in three cycles: prototyping in 2018/19, adapting to an all-online environment in 2019/20, and finding a hybrid remote/face-to-face balance in 2020/21. Each cycle started with a set of goals, proceeded to design and development, in-class delivery, evaluation through interviews and platform data, and then reflection. The course is introductory programming in p5.js (McCarthy et al., 2015), taught to both graduate and undergraduate students in their first years of design programs.

The undergraduate course contained 250–400 students and ran once per year, while the graduate course is smaller (30–90 students) and ran every semester, for a total of over 1,250 students. Students approximately evenly split between Australian domestic and International students from all over the world, predominantly Asia. The undergraduate students were mostly (more than 95%) enrolled in an interaction design focused Bachelor's program, while the postgraduate students were enrolled in similarly-focused Masters or Postgraduate Diploma programs. Gender balance was approximately 55% female, 45% male, and <1% non-binary.

The initial design goal was a platform where students could both complete open-ended coding challenges as well as design and submit their own challenges for their peers to complete. Coding challenges were envisaged as extension exercises to help students apply their newly gained skills to creative problems of an appropriate skill level. Making new challenges was conducted as a form of self-directed learning in which we asked students to "create a challenge that would have helped you to learn something that you struggled with in the first 8 weeks of this course." We refer to this approach as "retrospective self-directed learning" (RSDL) and intended it as a way to trigger the protg effect and encourage mastery (Chase et al., 2009). Particularly high-quality student-authored challenges would be included in the platform in subsequent years in an asynchronous instance of student-led teaching. As originally envisioned, students would need to both complete and create challenges for grades in the course as part of an innovative social learning network (Carvalho and Saunders, 2018).

The first cycle of the project, detailed in Section 4.1, spanned 2018 and the first half of 2019. The team focussed on a user-centered approach to getting a minimal viable prototype (MVP) into classrooms, starting with technology probes (Hutchinson et al., 2003) and user interviews. The second cycle spanned the last half of 2019 and the first half of 2020, which would by necessity prove to be a turning point for the project (see Section 4.2). The UI was overhauled and a challenge recommender system developed, and then project pivoted to a platform for fully online learning in response to the COVID-19 pandemic and the closure of university campuses. The third cycle (see Section 4.3) expanded on the (somewhat rushed) transition to a fully online learning experience, exploring how to support remote learning through both formally assessed and informal peer learning experiences.

3.2. Thematic Meta-Analysis

We conducted interviews with staff and students as part of each of the three cycles of research, using thematic analysis to explore the impacts of our intervention. Each of these analyzes was contained within a particular research project, often led by an honors student, with its own specific aims, research objectives, and coding scheme. These varied qualitative perspectives all contributed to the iterative re-design of the CCCs platform, but we also wanted a broader and more unified perspective. At the conclusion of the project we conducted a meta-thematic analysis (Batdi, 2017) to explore the underlying student experience of blended open-ended learning in this context. To do this we revised, coalesced, and expanded the initial codes, sub-themes and themes from each of the studies conducted over the course of the project.

The goal of this meta-analysis (see Section 5 for the results) is to explore—independent of all the design revisions, new features, and pedagogical changes—the impact of open-ended and student-led learning on design students learning to program. The meta-analysis sits alongside the insights about open-ended learning that arose from the research-through-design process. The triangulation of multiple data sources, multiple collection methods, and multiple researchers (Campbell and Fiske, 1959) across the three research cycles, coupled with the process of the reflective meta-thematic analysis gives us a rich perspective on the complexity of student experiences (Banning, 2003).

4. ITERATIVE DESIGN OF THE CREATIVE CODING CHALLENGES SYSTEM

The Creative Coding Challenges platform was developed as a way to explore open-ended learning, blended learning, and student-led learning pedagogies in an HCI context. The platform's iterative design and development can be characterized as occurring in three cycles, each with its own goals, design revisions, and evaluations.

4.1. Cycle 1: Discovery and Prototyping

The initial (2018–mid 2019) phase of the CCCs project combined early probes into how the intervention could be structured with our first full-semester deployment. The initial probes in 2018 were accompanied by a process of stakeholder interviews exploring how students and teaching staff responded to open-ended, student-led, and blended learning. The first design and development cycle in early 2019 was focused on delivering an MVP for classroom use as quickly as possible. This was followed by another round of student interviews, this time to explore responses to the MVP. The broad findings of this cycle were that a) the challenge-based blended learning approach was valuable for extension material, b) asking students to create challenges was an effective learning activity, and c) offering students a choice of what extension challenges to complete was confusing, and tended to result in some students doing everything and the rest doing nothing.

Our student cohort was familiar with blended learning and creative coding approaches from the unit's existing learning activities, but we wanted to understand how they would respond to a more open-ended approach. Before developing a fully implemented platform, we first ran two small technology probes as part of our prototyping phase to validate our design concepts and obtain qualitative feedback from fifteen student interviews.

In the first probe, approximately 250 students used a simple web interface to complete three Javascript creative coding challenges in a single 2 h tutorial class. In this probe the challenges were conducted in order, with no branching or choice. The structure of challenges themselves would be familiar to anyone who has explored the web for software development tutorials: a blog-like rich media article with in-line editors in

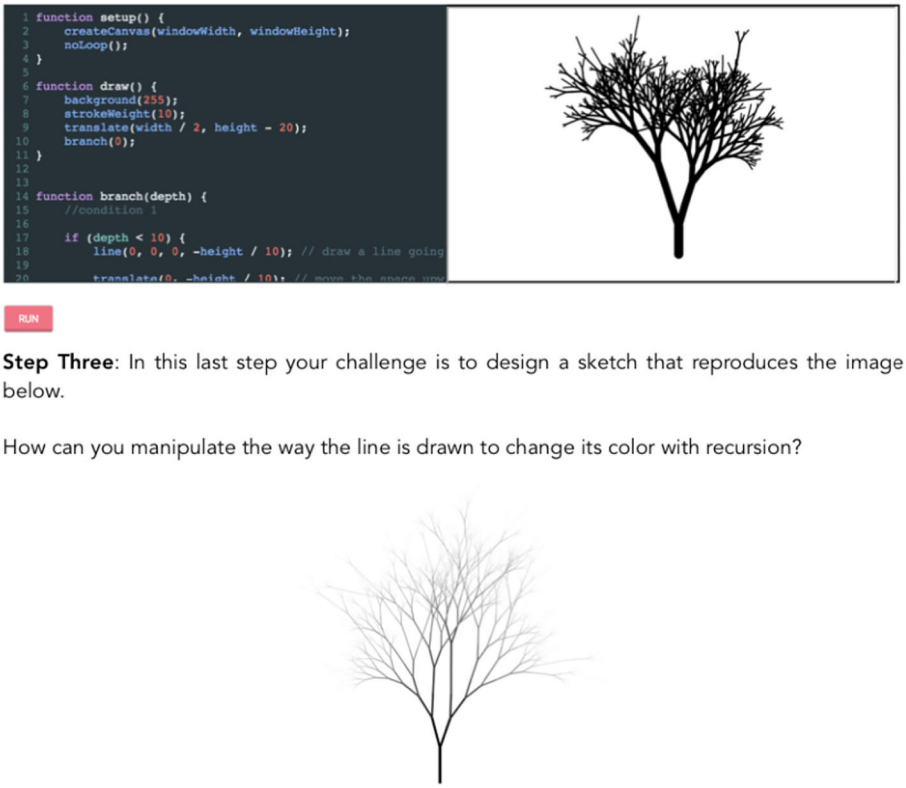

FIGURE 1 | An example challenge from our initial tech probes. Apart from minor advances to the editor (e.g., console access, stack traces) and some cosmetic updates, the structure of each challenge's page remained largely unchanged throughout the project.

which code could be written, saved, and run. See **Figure 1** for an example excerpt, in this case a challenge about learning recursion by drawing and styling a tree.

The second probe was conducted toward the end of the skills-focussed component of the class, before the pivot to project work for the final few weeks. In this probe the same cohort of 250 students were given choices as to which challenge to complete next. We employed a tree-like structure (seen in **Figure 2**) to show dependencies between challenges, ensuring that students would complete required prerequisites before moving on to more advanced concepts. Challenges were separated into a "trunk" of mandatory challenges with a branching series of optional "leaf" challenges for students to complete at their discretion.

Two rounds of semi-structured interviews were conducted, in order to evaluate these prototypes, one after each probe was used in-class. Student participation in the interviews was voluntary, conducted by researchers who were not in the face-to-face teaching team, and expressly disconnected from any suggestion that participation (or lack thereof) would impact grades. The first round focused on the challenges themselves, how they felt to do, what was fun, what wasn't, as well as how students searched for supplementary material to complement in-class activities. The second set of interviews (administered to a non-overlapping subset of students) focused on choice: how and why students chose to do the subset of challenges they completed. A speculative question concluded both sets of interviews, asking how the student would feel if a lot more of these challenges had been in the course, with the option to choose which ones to complete.

Students loved the challenges themselves, particularly the ones with clear multi-step instructions and well-crafted scaffolding. Opinions on open-ended learning were broadly positive but with some dissenters: perceived benefits included autonomy, more productive time with teaching staff, and increased engagement. Perceived disadvantages, however, included worries about whether their sub-set of challenges would be comprehensive, how much access to tutors they'd have if the course was heavily "blended," and how much motivation students would have to do anything that wasn't mandatory. Clearly just a taste of open-ended learning inspired both joy and fear.

We also implemented our first RSDL assessment, with 180 students in a follow-up course being asked to create a coding challenge that would have helped them learn a fundamental coding skill (like arrays or objects). Our hypothesis was that the protégé Effect would help solidify their knowledge, while simultaneously giving us a source of new, diverse content for our platform. The students performing this task had all participated in the two "probe" workshops in the prior semester. An additional round of (seven) interviews was conducted to

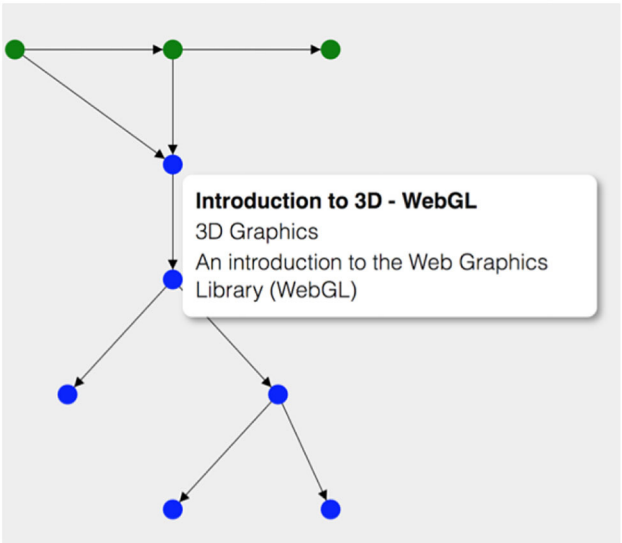

FIGURE 2 | The challenge "tree" used in the second 1-week technology probe that was the fore-runner for the CCCs platform. Mandatory challenges are in green, optional in blue.

explore the impact of this self-directed learning-by-teaching exercise. These questions focused on why students created the challenge they did, why they felt it would have helped them learn, and how they would feel if their work was used by other students.

The response among these (admittedly self-selecting for an interview) students was overwhelming positive, with the vast majority saying they'd created a challenge involving something they themselves had struggled to learn, that they had learnt more in creating it, and that they would feel positively about other students completing their challenge in the future. Of particular interest was the sense of "relatedness," or shared struggle: students making challenges felt that future students would "come from the same head space," or "understand [their] pain." However, the majority of the actual challenges produced by students were not of high quality, mostly lacking in appropriate scaffolding and/or being so disjointed from the course content that they could not have been used. Nevertheless, the benefit to their creators was apparent.

Following the success of these probes, we reflected on the feedback in the interviews to implement the first full version of the CCCs platform in the first semester of 2019. This first complete design had two main goals: to collect some survey data that could be used to improve the challenges, and to provide a whole semester of examples to the students creating challenges in the subsequent course. The branching "tree" interaction model was shelved for simplicity, with all challenges being presented as lists under each week, in approximately ascending order of complexity. After completing each challenge, students were asked to rate (on a five-point Likert scale) its level of difficulty and their level of enjoyment in completing it—this feedback let us quickly identify and revise challenges that were boring or too hard.

4.2. Cycle 2: From Blended to Online, Overnight

The second cycle spanned the last half of 2019 and the first half of 2020, which would by necessity prove to be a turning point for the project. The first goal of this cycle was to expand upon the MVP, both in terms of its interaction and educational design. A prototype educational recommender system was also deployed to assist students with their confusion about what challenge to do next. The second goal was to explore the quality of the student-created challenges and add our first batch of student-created content to the platform.

Interviews throughout the cycle evaluated student motivation to do challenges beyond the minimum required, finding (as hoped) that some students were intrinsically motivated to do additional creative coding tasks. However, other students were still struggling to find their footing, and a fraction of students were obsessively doing every possible challenge to ensure they didn't "miss out." At this time we started realizing that choice—the goal of our open-ended and challenge-based approach to the course—was a double edged sword, creating empowerment for some students but anxiety for others. Understanding the cause of this bifurcated experience and figuring out how to support choice positively became a major focus of the project.

The CCCs platform was deployed to around 95 students in the second half of 2019 with a fully re-worked user interface, which can be seen in **Figure 3**. This revision focused on bringing the interface to the professional standard expected by students familiar with the modern web—a task made possible by the fact that the teaching and research teams included professional interaction designers and web developers.

We also prototyped an educational recommender system (Bodily and Verbert, 2017) intended to provide support for those students having difficulty choosing which challenge to complete next. This used a hybrid knowledge-based and item-based recommendation approach (Ricci et al., 2011), combining data about students with data about challenges. The introduction of a recommender system brought aspects of guided learning models to our open-ended model, but it retained its open-ended nature as engaging with the recommendations was always voluntary. Metacognitively-aware personalisation is an established strategy in learner modeling (Bull and Kay, 2013), and has been applied in intelligent tutoring (Roll et al., 2007), and adaptive assessment (Krouska et al., 2018) in addition to content (Hidayah et al., 2018).

The logic for our recommendations, which appeared in a banner at the top of the UI, was as follows: If there was a mandatory challenge that had not been completed, the recommender would always suggest that first. This caught most of the disengaged or truly struggling students, who would be best served by engaging with something introductory (or, most likely, working on simpler exercises given out in class before tackling the challenges). If not, the system would use the number of challenges the student had completed as well as their average difficulty and enjoyment ratings to place the student into one of two categories: "striving," or "thriving."

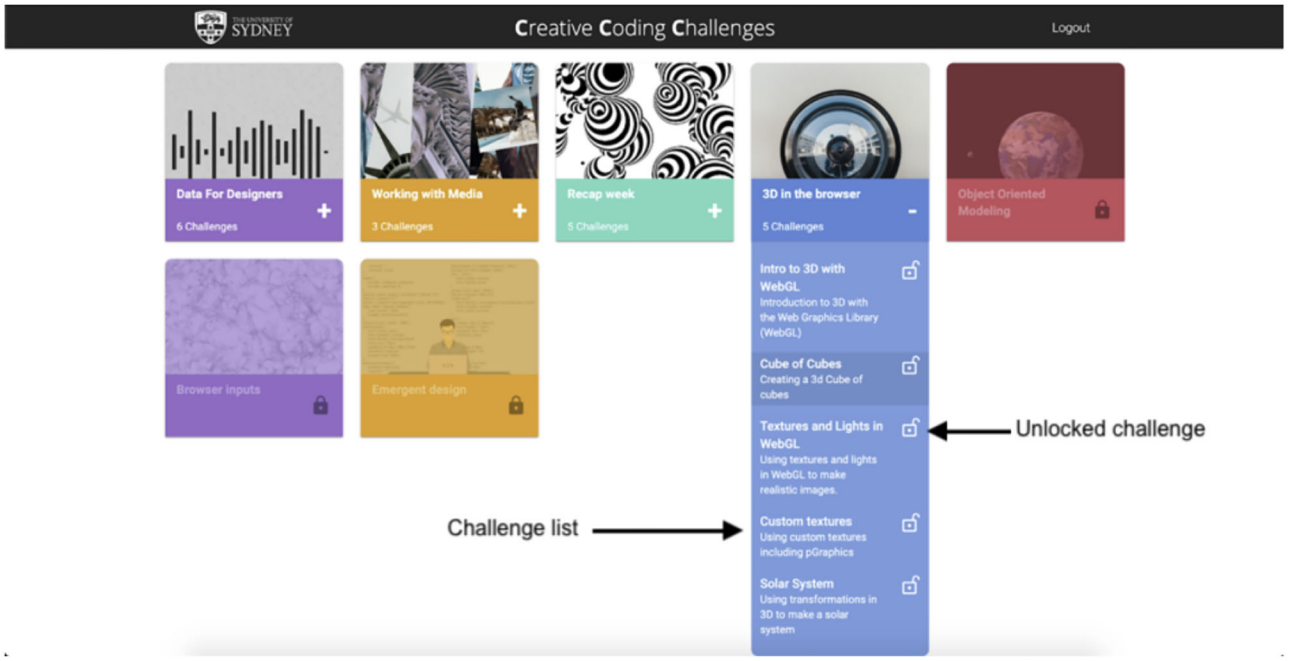

FIGURE 3 | The second version of the CCCs platform interface, showing challenges grouped into "modules," each corresponding to a week of the course.

"Striving" students, those who had rated challenges more difficult than average, were recommended challenges that contained topics that were precursors to those in their most recently completed challenge. A directed graph of programming topics and their dependencies was constructed from the challenge tags to support this. For example, understanding loops depends on understanding conditional statements, and understanding vector-based character movement depends on understanding both arrays and co-ordinate systems. By contrast, "thriving" students, or those who had rated challenges less difficult than average, were instead recommended challenges that similar students had enjoyed, a collaborative filtering approach based on the Singular Value Decomposition algorithm (Su and Khoshgoftaar, 2009). The goal was to try to empower those who felt that choice was an opportunity, while offering support to those who found choice anxiety-inducing.

12 students and six tutors were interviewed during this cycle, primarily to establish the effectiveness of the recommender system, but also (in the case of the students) to continue exploring how they choose challenges and what improvements they might want in the platform. A thematic analysis was conducted on both cohorts together, with ideas around progress, difficulties, communication and motivation emerging as important factors to both staff and students. The vast majority of students were positive about the CCCs platform and its challenges, for reasons that can be broadly characterized as a preference for active learning (Freeman et al., 2014). "Striving" students (we obtained permission to retrieve each interviewed student's record from the platform) worried that there were things they were missing, and often found challenges to not explain concepts in sufficient detail: they needed more basic learning material than the recommender could provide. "Thriving" students were more likely to view the recommendations positively, but found that there weren't enough truly open-ended challenges in the system yet, and so opportunities for truly serendipitous discovery were limited.

A significant fraction of students did not trust the recommendation system's ability to teach them what they would need to pass the unit, and didn't see how its suggestions would directly lead to improved grades. Interestingly, a number of students in both categories also wanted to re-do challenges as revision, which the team had explicitly excluded from recommendations. The recommender had helped start to address the gap between those empowered by and fearful of choice, but (and this attitude was prominent in both tutors and students) there was still clearly a need for structured, teacher-led learning. The challenges, even the mandatory ones, could only build on top of that.

Also during the second semester of 2019, the first class of students who had used the full CCCs platform completed the RSDL task in the follow-up course. In-class observation and informal discussion revealed that this cohort of students also found the learning-by-teaching component of the task helpful for reinforcing their knowledge. A small number of student submissions—six in total, out of almost 180 submissions—were judged to be of sufficient quality to be incorporated into the CCCs platform after significant editing. These challenges were labeled as "student contributed," and our intent was to continue iterating on this formula year-on-year. We planned to refine the recommender system, continue working on how to empower student choice without triggering

anxiety, and to keep integrating exceptional student-contributed challenges.

It was at this point, however, that the COVID-19 pandemic forced the course entirely online, and the role of the CCCs platform—as well as the scope of this project—changed significantly. Instead of a platform for what were effectively "extension" exercises, CCCs had to become practically the whole course, supplemented only by pre-recorded lectures and video-conference tutorials. The notion of a "challenge" expanded overnight to cover all tutorial exercises, which ran counter to some of our findings but was the only feasible way to run the course during the crisis. In addition to its enormous impacts on the mode of delivery, it also had resourcing impacts, as in the rush to pivot online, further developing the recommender system was not feasible and that component of the project was shelved.

It was always our intention that the research questions would evolve as demanded by both the needs of the classroom and the capabilities of the technology. However, the unexpected pivot to fully-online learning caused our research to diverge to a degree that we could not previously imagined. We were no longer able to really assess (either qualitatively or quantitatively) whether our year-on-year refinements were delivering improvements to the student experience, because the contexts were now so inconsistent with each other that such comparisons were meaningless. COVID-19 also impacted the quantity and quality of available challenges: instead of refining our open-ended creative challenges and adding a few exceptional student contributions, we had to rapidly shift the entire course online. However, this offered a unique opportunity to study a different question: how could we design effective open-ended and student-led learning in a fully remote context?

4.3. Cycle 3: Pivot to the Protégé

The third cycle was all about consolidating the use the CCCs platform as the main focal-point of the course. We remained in remote-only mode for the second half of 2020 before returning to a hybrid model with some face-to-face classes in the first half of 2021. Throughout this cycle we focussed on further support for remote students in the form of pair programming for open-ended creative challenges, with very positive feedback. RSDL was also implemented within the programming subject itself, rather than as a component of the next semester's follow-on subject, to highly polarizing feedback.

Our experience rapidly pivoting online taught us that much more structure was needed for effective remote-only learning. To address that we abandoned the recommender system, which was at its best extending in-class learning, since we now had to focus on the course as whole. The "tree" structure from the first cycle was re-introduced in a new UI. Each week of the course starting with a pre-recorded lecture, then a tree of challenges, some introductory (and mandatory), and some more advanced, creative, and optional. This interface, which was used throughout the third cycle, can be seen in **Figure 4**.

By the second COVID-affected semester we had made the decision to pivot away from the notion that students could choose their own path through the challenges by branching out in directions that interested them. This "open-ended direction" approach proved both difficult to support in remote-only learning and difficult for students. As the platform now featured "challenges" for every tutorial exercise rather than just extension material, the ratio of mandatory-to-optional challenges increased substantially. With the role of the platform as a place for open-ended extension material no longer clear, the proportion of students perceiving the platform's open-endedness as anxiety-inducing increased. The open-ended tasks where students could choose how to solve a proscribed problem, however, were still among the highest-rated challenges on the platform. This suggested that the "no right answer so long as you make something intereting" task structure inspired by creative coding was still viable in remote learning contexts.

We also developed additional scaffolding for the RSDL task in the form of a walkthrough to help students create their own challenge. This approach framed the task as making a "puzzle," the solution to which required understanding something something about one of the concepts in the course (e.g., arrays, objects, nested loops, etc). This framing—which we had used internally for a number of the well-regarded challenges—was developed through a series of co-design workshops with students and then evaluated in focus groups after students had submitted. Student responses to the scaffolding were very positive, although still only a fraction of student-submitted were of sufficient quality to be included.

By 2021, with about half of our students back on campus, it was clear that students were experiencing significant anxieties during remote learning. A prominent source of student anxiety appeared to be how their emerging grasp of programming concepts compared to the course's expectations. This was true among both in-person and remote students, but stronger among those not coming to campus. To explore this we conducted 39 interviews exploring students' satisfaction with the platform and course as a whole, the latter to capture some of the sentiments around learning during the pandemic. The major findings were that students needed more connection, they needed more support, and they needed more motivation. It was clear that learning programming, which was an unfamiliar discipline for many of our design students, was an isolating experience.

To address these needs we developed a remote, creative-coding focussed version of the pair programming approach (Wiebe et al., 2003) and piloted it in several tutorials. The pilot was intended to add elements of peer learning (van Popta et al., 2017) to our unit, a familiar experience for design students used to working in creative teams. The goal was to introduce programming in pairs as a middle-point between the tutor-led walkthroughs of material and students working individually. This created a three-layer "I do it, then we do it together, then you do it yourself" approach based on the notion of gradual release of responsibility (Pearson and Gallagher, 1983). Students conducted these pair programming sessions remotely, completing creative coding challenges together using video-conferencing (Zoom) and a collaborative visual workspace (Miro) to structure their challenge responses.

Five interviews were conducted with students who participated in the pair programming pilot, with a thematic analysis revealing that the process had helped them overcome

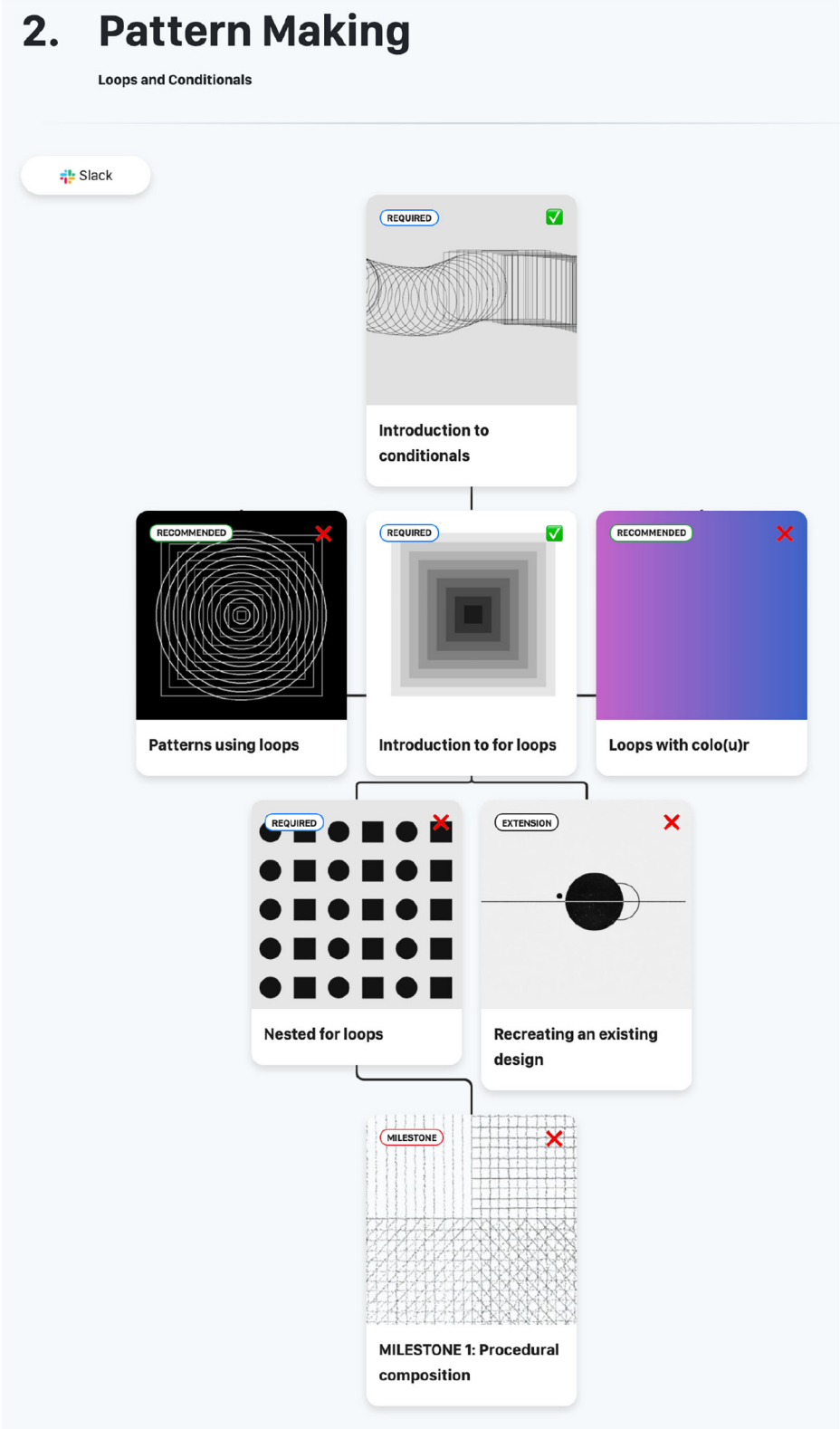

FIGURE 4 | The final UI used in the CCCs system, after the pivot to fully remote learning. By this point the notion of "challenges" had been expanded to cover all learning activities, not just open-ended extension material.

isolation, develop better coding self-efficacy, and be more pro-active with their learning. These social benefits of remote pair-programming were actually more universal among the interviewees than the benefits traditionally associated with the method (i.e., learning from each other and holding each other accountable). Most existing studies of pair programming were face-to-face, which suggests that an additional benefit of pair-programming for remote and isolated cohorts is the simple opportunity for much-needed socialization.

We also moved the RSDL task into the programming unit itself for the first time, with students in the last few weeks of the course creating a challenge that they personally would have benefitted from earlier in the course. Even though this exact assessment had been completed as a "refresher" in the first few weeks of the next semester's course for 3 years now, this particular version produced very different results.

The integrated RSDL task was the single most polarizing assessment any of our teaching or research teams had ever seen. Students either absolutely loved it, saying things like "I found it was a turning point in my learning where I actually could freely explore" or utterly hated the very idea of it, saying things like "in industry they pay us for our work, we don't pay them!." Five students were interviewed about their experiences with the RSDL task, with another seven offering anonymous feedback via a survey. One particularly negative group of responses exhibited the sentiment that students felt they were not getting their money's worth: they felt that asking them to teach was asking them to do our jobs for us. One hypothesis is that the student-led teaching exercise may have become a trigger point for broader student concerns about the value-for-money of remote education, particularly among students who also expected a more traditional mode of delivery.

In fact, the detractors of the "create a challenge" task were almost entirely remote students studying from overseas due to the ongoing pandemic, while the supporters of the task were almost all in the face-to-face tutorials. Language issues may have also played a part, as a portion of students appeared to misunderstand the task and produce a completed puzzle without any scaffolding or steps. Several of those students became hostile when they received poor marks for these submissions, asking why they should have to break their work into "baby steps" to help other, struggling learners. To speak freely for a moment: an actual flame-war broke out between supporters and detractors of the assessment on the class discussion board, complete with an ugly undercurrent of anti international-student sentiment. None of us had ever seen anything like it—and it underscores the challenge of effective open-ended learning in diverse student cohorts.

5. RESULTS

The thematic meta-analysis of interviews conducted throughout the life of our project revealed student attitudes toward open-ended blended creative coding fell into seven broad themes: *learning as a skill*, *learner technology*, *learner autonomy*, *social learning*, *learning support*, *content complexity*, and *learner struggles*.

5.1. Learning as a Skill

In tertiary education, particularly in HCI, a more student-centered approach to learning is encouraged. This means that instead of instilling knowledge into students, we as educators facilitate their learning by giving them the tools to develop their own learning strategy. **Table 1** shows the sub-themes that made up this theme. The CCC platform encouraged students to build their metacognitive learning skills through a more reflective and introspective approach, with students agreeing that the questionnaire at the end of each challenge allowed them to *"really reflect"* on how much they have improved. This reflection also allowed students to see the benefit of this subject outside the scope of semester, *"in this course I feel like I'm investing into learning a new skill."*

The ability to reflect on one's work also had an interesting impact on students' desire to push themselves, one student admitted that "*It was ok for me not to finish the advanced challenges*" because *"I know I pushed my limits and can see that it was my best attempt."* Students also acknowledged the difficulties faced working independently, *"I struggled a lot working through some of the assignment challenges by myself"*. But upon reflection, one student observed that *"I'm glad I struggled on my own... even though I felt so stressed during that time. It helped later, just because you knew you had to struggle for it"*. The initial difficulty of the challenges seemed to encourage students to develop their own protocol for solving them, with students being able to reflect and "*identify their own weaknesses*", and prioritize accordingly.

5.2. Learner Technology

Unsurprisingly, the technology that facilitates learning for students significantly mediated their experiences, as reflected in our interviews (see **Table 2**). The CCC platform was initially designed to be used in partnership with physical tutorial classes, however, due to the COVID-19 pandemic, online learning resources were prioritized far more than originally planned. This resulted in pandemic-affected students describing the online tutorials and CCC platform as only "*somewhat interchangeable*", with others describing how they "*couldn't get enough information from the [CCC] platform to do challenges by themselves*". This physical/online learning disconnect was further exacerbated by some innate limitations of online learning whereby the restrictive nature of a virtual classroom *"doesn't allow [students] to feel comfortable asking questions*" with one student noting that they *"don't trust who they don't know - why would I want to talk to my peers or tutors if I haven't met them?"*.

This negativity was in stark contrast to our pre-COVID data collection, where students often expressed comfort seeking clarification or help, stating, *"I do not have an issue calling a tutor over or messaging on slack, I feel quite supported in that regard."* Despite these limitations and the challenges of pandemic-impacted semesters, students did discuss how motivating and impactful the CCC platform was. P5.js artworks are *"really inspiring"* for students, and immediate visual feedback *"drives [them] toward a goal"*. During scenarios where the challenge outcome was not clearly communicated, students expressed frustration, stating there's *"no answer for us to know what our goal*

TABLE 1 | Sub-themes within the "Learning as a Skill" theme.

Sub-theme	Example quotes
Learning Reflection	...you have to actually like to rate how you felt about that particular challenge and whether you liked it before you can move on. And I think it also has that kind of reflective aspect to it, which I guess most other courses don't really do.
	yeah well now that we're week 11 or 10 or whatever and now I look back to when it was week 2 and I'm like "you have no idea." I appreciate being able to reflect on my process, a motivator for sure.
Proactive Learner	I feel like, I'm glad I struggled on my own, even though, like, I feel so stressed during that time. Because it helped later, just because you knew you had to struggle for it, and you had to go really deep, maybe outside of traditional resources to understand how to solve the problem.
	In that time, I might have worked through it myself, maybe that's a good thing. But also, it's like, it's helpful to have that help as well.
Value of Learning by Doing	I really like how there's the structure, they introduce, you know, maybe a few features or a few functions and whatever, and then you put them into practice straight away.
	I like the way that the lectures are broken down into challenges like we're not sitting there looking to learn about theory, like we are doing something practical. I think that's how you learn.
Perceived Future Benefit	In this course I feel like I'm investing into learning a new skill.
	[Will you use programming after this semester for anything, not necessarily P5, but programming in general?] - Probably. I will...I think for certain I could use this one to design some interesting program for my career or university, so I probably will.
Student Workflows	So that's probably how I study, I identify my own weaknesses based on what I think the quiz will be about.
	It was a bit daunting at first, but then I started to make myself structure. So now I like drawing out a picture and I've chosen, like, I'm going to draw a sunflower for my final thing. And hopefully as time progresses, it's going to have interactive elements. It is just all about breaking down the elements.
Recognizing Own Competencies	Some of the challenges, I didn't finish them, but I felt good about it, I mean I didn't feel good but like I was relieved that I like it. It was okay for me to not finish "advanced" challenges cause honestly that was really hard for me.
	If I do my best, I do not care if I don't get a HD because I know I pushed my limits and can see that it was my best attempt, not everyone can get a HD.
Independent Learning	[on learning concepts] by ourselves maybe a little bit helpful is to make sure that everybody understood it
	It's like I need that hand holding. I need a basis because I feel like I can't build anything from scratch. But I think potentially for the weeks beyond week 6, week 7 when we're getting closer to like the stage now where we're building our own thing, potentially be good to maybe hide that prefilled text so that you can kind of have a go yourself at how you might build it from scratch.

is", and nothing for them to "go back and have a look" to see "how off they are".

TABLE 2 | Sub-themes within the "Learner Technology" theme.

Sub-theme	Example quotes
Disconnect Between Tutorial and Platform	I think the lecture content and the challenges are quite disjointed and don't really help each other. I think they can be incorporated better.
	I didn't have the knowledge to actually do that one and I accidentally missed the bit where he was explaining it in class, so I was just like, " shit. I guess I just have to submit the not finished one." I didn't get very far with that.
Online Learning Limitations	I just don't like speaking on calls, when there's a lot of people. That's just how I am quite an introverted person. Yeah, I don't find it comfortable to ask the questions. So I honestly just wouldn't ask.
	When I'm working on the challenge, I'm stuck. Like, I know there are people to ask, but there's 70 people in the one session, so I feel a little bit bad sometimes asking and also, I guess since we're in that zoom group, I can't really go to the person next to me and ask because there's no one there.
Creative/visual Code allows for Instant Feedback	I really liked the method where you can test your code and then you can immediately see the result.
	I find it helpful when I can visualize what the outcome will be, it is motivating!
What Students Need From Learner Platforms	I like the way that the lectures are broken down into challenges like we're not sitting there looking to learn about theory, like we are doing something practical. I think that's how you learn.
	If I like the picture I want to do the exercise. If the picture isn't attractive I feel less willing to do the exercise.
Platform Design	I know we have choice, but I still feel I need to follow the structure
	I think the content on the website (ccc platform) is too limited and maybe more examples would be better.

5.3. Learner Autonomy

Feelings of both autonomy and a lack thereof arose regularly in our interviews with students, as can be seen in **Table 3**. Students had a strong desire for different types of choices. Some students liked the ability to choose what they learnt, *"It's giving the student or myself autonomy and agency to kind of learn core foundational concepts that are essential across the whole unit"*. However, this wasn't a clear majority. When asked whether they felt they had a choice in what challenges they could complete, many students we interviewed expressed that they thought they *"were just required to do everything to do well"*.

Fear of failure was a prominent reason for students not feeling like they had any choice, mostly relating to the mid-semester exam: *"I'll get to the exam and with my luck, the random question will be the area that I didn't choose to learn more about"*. Other students instead appreciated the autonomy to choose how they complete the challenge, rather than what they learn. Students appreciated *"the opportunity to explore and do your own thing"*, noting that *"it's more personal driven, which I like. You get to come up with a design that you imagined, not what was given to you as a brief."*

TABLE 3 | Sub-themes within the "Learner Autonomy" theme.

Sub-theme	Example quotes
Desire for Choice	I like the freedom of choice. Like, you know, if you're not making us do everything, like some people just don't have time or, and some people just want to learn more, so it's up to them.
	It's giving the student or myself autonomy and agency to kind of a learn core foundational concepts that are essential across the whole unit
Inspiring Self-Directed Learning	It's good to have help with other people. But you also need to be challenged individually to go deeper.
	I don't think me and P3 really use the driver thing anyway, we just kind of did it on our own, and worked out how to collaborate and solve problems, all right.
Freedom to Explore	I also like the opportunity to explore and do your own thing, like the challenges in the first week. It's more personal driven, which I like. You get to come up with a design that you imagined, not what was given to you as a brief.
	This is the thing I've already got in mind that I want to do. I already know, I like to come into this class. I have all these ideas and I'm thinking how can I best use this class to realize these ideas? So I'm already gravitating toward things that I think are more relevant.
Feeling Comfortable Being Challenged	Challenging my own thinking around problems that I would have originally just disregarded, that I had solved in the first place.
	There are some things like, I feel like, I'm glad I struggled on my own, even though, like, I feel so stressful during that time. Because it helped later, just because you knew you had to struggle for it.
Satisfaction of Visible Progress	I also like the ability to kind of go back and look at what challenges you've done and haven't because you can't complete a challenge unless you've completed the previous one.
	I think it's just like that bit of representation that this is what we're focusing on now and you'll build up to be able to complete these future interactions, these features concepts, which I think is really encouraging.
Motivated by Marks	To be completely honest. I will only focus on what is testable on the exam
	Just because for me as a student, I want to optimize what will get me the most marks in my limited time. Um, and if I know that the criteria would be like looking at these elements and um, creating a novel idea and if the challenge is related directly back to that criteria, then I would prioritize them first.
Conquering Individual Goals	Well, you instantly fall in love with the challenge that you struggle with at first and then you conquer yourself.
	I think there's a sense of satisfaction in being able to, like solve problems
Sense of Accomplishment	My main motivation is to just get all the greens over here when I finished one challenge and I completed properly. Gives me like what motivation to go to the next one and finish that one too
	[the challenges] the reason why I think is enjoyable is that I am doing what I want to, during this challenge I can feel a kind of achievement or when I can, solve the problem myself.

5.4. Social Learning

Social factors played a big role in student attitudes toward open-ended learning (see **Table 4**). Feelings of isolation and detachment from peers predated the pandemic, seen in sentiments like *"Yeah, no, they [students] don't really help me. I don't know. I don't really know how to ask anyone. I haven't made that sort of connection with anyone yet"*. A fear of judgement by their peers was also present: *"I wouldn't ask a peer to help me cause I would be worried they are smarter than me"*. Students who completed the course during the pandemic definitely experienced enhanced feelings of isolation: *"Just naturally being virtual and away from people, you just don't feel as connected."* When we introduced pair programming to our virtual classroom, students acknowledged that there was a major improvement to learning, *"even just the practice of explaining, or pretending that you know what you're explaining catalyses learning"*.

Students also appreciated the support from their peers, and having someone there they can vent to: *"I think it was nice to, like, mutually support each other."* The online classroom also presented barriers to language accessibility, with some students feeling that *"the context of physical space and classrooms is very important to help us understand English"*. The ability to actively converse with peers also was hindered, with one student expressing a need for *"tutors to teach us how to ask questions"*, with students with English as a second language expressing that *"It's not a problem about listening, it's about talking"*.

5.5. Learning Support

A very prominent desire amongst students was additional learning support (see **Table 5**). This was often expressed through students vocalizing their concern over *"minimal revision opportunities"*. With some students agreeing that *"week to week when you come to class things progress based on what you've learned previously. And if you don't go back and revise and do it, you struggle."* This can be attributed to the issue of autonomy (see Section 5.3), with students feeling that open-ended learning makes it difficult to know what knowledge will be critical in future tasks. Students also expressed discomfort researching additional resources unless promoted or encouraged by the tutor, noting that tutors have a *"sense of authority"* and *"if it worked for the tutor then it should work for us."* This also caused some initial hesitancy with pair programming, with some students agreeing that *"sometimes with students, you can't be sure they are right. With tutors, it is their job, so you trust them more."* Regardless of when students completed the course (pre or during the pandemic), they expressed that access to tutors was something that they really craved: *"one-on-one time with the tutors is absolutely the most valuable thing. Right. But it's kind of limited to class time."*

5.6. Content Complexity

The perception that learning to code is inherently difficult was a common thread amongst the cohorts of students, as seen in **Table 6**). For some, that initial fear deterred them from the beginning: *"I was so nervous coming into this subject, and it just made my experience worse"*. For others, the pace of the course was stressful, with some students surprised that the *"difficulty*

TABLE 4 | Sub-themes within the "Social Learning" theme.

Sub-theme	Example quotes
Improvements to Communication	Some of the tutors' replies will be easy to understand and some tutors' answers will make us more confused.
	But in the grading standard, the definition of originality is very vague.
Effective Partner Matching	I think if we were on different levels in programming knowledge, that would be frustrating probably for both parties.
	I think it was nice to, like, mutually support each other.
Learning from Others	I think peer learning is really, really important.
	Because there's a lot of times where I'm pretty good at something. But others, I just need another perspective. And I can't always just do that in the classroom, because that's spent teaching us like the content and everything.
Encouragement from Peers	Even just the practice of explaining, or pretending that you know, what you're explaining catalyses learning.
	I am initially quite happy that I get to be mingling with other people, I think because of all the remote learning at the moment. It's nice, just any opportunity to kind of work with others.
Judgement by Others	I asked like on the second of [or] third week, one of the students in like, the explanation was just like, Oh, like how come you don't know this? And so I was like a little bit taken back by it.
	I feel like when they say ask the general chat in slack sometimes it might be stupid questions.
Willingness to Engage	if I had a question about what they were teaching in class, I would just act straight away.
	We can divide into groups and work together. I think that will be great for me and can help us. So some questions can be asked and answered.
Language Accessibility	questions to my tutors. But if tutors are explaining to me I can, I understand. It's not a problem about listening, it's about talking.
	If I just watch the CCC, I cannot code anything, well not anything, but a lot of things that I cannot understand including the english explanation.
Feeling Detached from Tutors/Peers	Just naturally being virtual and away from people, you just don't feel as connected.
	Yeah, no, they [students] don't really help me. I don't know. I don't really know how to ask anyone. I haven't made that sort of connection with anyone yet, because we only had 3 weeks together. So, it's been difficult.

TABLE 5 | Sub-themes within the "Learning Support" theme.

Sub-theme	Example quotes
Revision Opportunities	[cont.] - because I guess week to week when you come to class things progress based on what you've learned previously. And if you don't go back and revise and do it, you struggle.
	("extra challenge" challenges) even if it's not compulsory, I like how it just helps you test your skills more
Resource Availability	[cont.] - If I don't get the idea of why this function works, I'm checking YouTube from the coding train. All students use that.
	I feel like I don't learn things very well. Right. Um, and I've been struggling to find resources that will help me to just practice.
Trouble Applying Tutorial Content	Ook, to be honest, I don't, I don't love the, um, creative challenges. Um, I don't, I have actually just not found them very useful, especially without guidance, especially out of the context of the classroom.
	Sometimes it's really exhausting because I can't figure out what to do.
Tutor Accessibility	The one-on-one time with the tutors is absolutely the most valuable thing. Right. But it's kind of limited to class time.
	But also with this online model, it's a little bit harder to access help. It pushes you toward self learning a bit more.
Desire for Credible/Reliable Sources	Whereas, like I like the sense of authority of you guys, I don't know, I assume you guys thought about the best way to give us this information.
	[on preference between help from tutors v students] I think teachers, because then you know as a fact that the answer is right.
Establishing Expectations	It'll always be a concern in the back of my head as to how much I'm supposed to learn to do well in the course.
	I just don't know what exactly the tutor wants and the rating is relatively subjective.
A Boost of Learning Support	[the platform green indicators] I think it's really helpful because it tells me which one I need to work on.
	So I think if that recommendation model can really guide students into focusing on what's really important, not just for assessments as well, but just in general, like as a designer or as a developer, like what are the core things you need to get right and what are you struggling with and filling in that gap. I think that'd be really good cause I think a lot of students kind of like give up really early with programming because they can't really get the basics. And if you don't get the basics you can't really get the bells and whistles.

increased so much", or that there was not a lot of time to "*reinforce your learning*". This unexpected difficulty was often the cause of students struggling to learn transferable skills, with one student noting that they *"get very confused as to how to apply different techniques"* and that they know *"how it is done, and how it is useful, but if you asked me to use it in a challenge I couldn't."*

The rapid expansion of the platform's role in 2020 also created some issues around content quality. Students expressed that at times the challenges were verbose or overly complicated, *"why do you need that much text for a challenge that takes a couple of minutes?"*. To at least one student the text descriptions that were intended as scaffolding added *"more anxiety than if they weren't there"*. Conversely, students suggested that some of the harder challenges *"were not explained at all"*, with students feeling like they were *"left in the dark"*. Some challenges that were well-received pre-pandemic evoked these responses once the course switched to remote learning, suggesting that the levels of

TABLE 6 | Sub-themes within the "Content Complexity" theme.

Sub-theme	Example quotes
Unexpected Difficulty	[cont.] - the challenge, I mean, uh, in the beginning, in the beginning challenge is easy and uh, I can easily solve it, but the second challenge is just like, look, difficulty improved so much.
	I think it moves, uh, very quickly. Uh, I think all of the challenges are challenging, which makes them very interesting. Um, but I think it would be, um, you there's not a lot of time to reinforce your learning.
Knowledge Confirmation	I feel like we need a way to make sure that we understand, we know what we've been taught.
	Um, maybe the marks we give to each challenge and we can know which we didn't do well, so we can we really again, yeah. And I found that we usually need to ask for a resolution in Slack. Maybe you can after 1 week or something like that. Put the, say the answer. Maybe some of the solutions to each challenge.
Trouble Generalizing Concepts	Um, however, when I get the feedback on the code, sometimes I don't understand the thinking behind what I've done wrong, so I get the change in the code, but I've still gone, wow, I never would've thought of that. I don't know what to do. So I still feel a little bit like I'm not quite learning my own mistakes as much as I do one on one.
	I get very confused as to how to apply different techniques, such as the mapping. What does it mean? I see how it is done, and how it is useful, but if you asked me to use it in a challenge I couldn't.
Fundamental Difficulty of Computational Thinking	I think it depends on the challenge, how difficult it is because in the beginning I thought the basic class was very easy, so just out of class I wouldn't usually use the platform, but after weeks 4 and 5 the challenge became very difficult.
	Some of my friends are finding it a bit difficult, especially because it's the first time doing a programming related unit

scaffolding required for complex and open-ended content is very environmentally dependent.

5.7. Learner Struggles

Lastly, but expressing a critical component of the student experience, particularly among a portion of the cohort, were sentiments relating to the struggle of learning to program (see **Table 7**). In particular, catering for different learning styles presented itself as a major barrier. Some students struggled to adapt to the open-ended and student-led way of learning: *"how am I supposed to determine for myself when I have learnt or done enough to be confident?"* Some felt quite overwhelmed by the freedom: *"I think one of the biggest things for me is like, sometimes I'll get the answer, but I don't think I'm doing it right or in the right order."* A very interesting theme that came to light during the pandemic courses was the cultural learning differences. Students from outside of Australia stated that in previous semesters they could *"pick up the culture a lot quicker, which made it easier to adapt". When learning from their home country however, this was "a lot harder".*

Open-ended learning was a big adjustment for some international students: *"coming from an Asian learning background, it's been ingrained that like everything that's presented to you is testable."* Other students, mostly those who came from HCI or design backgrounds appreciated open-ended learning, *"I think doing everything online, being forced to do everything online, made it a bit more transparent in different ways that we can learn."* Overall, and regardless of background, study fatigue played a big role in inducing anxiety amongst students, some stating that they *"had had enough"* and just submitted what they had because *"they were sick and tired of getting things wrong"*. Continuous practice seemed to be exhausting for students learning remotely, *"practicing is much harder than normal studying, my brain cannot copeâ"*.

6. DISCUSSION

As a research-through design project paired with a summative meta-evaluation, the findings arising from this research come in two parts: the meta-analysis of our student interviews, and the design insights arising from almost 4 years of iterative interaction design. Here we present both, starting with what we have learned from our students and then putting it all together into a set of recommendations for future open-ended learning in HCI contexts.

6.1. Understanding Student Attitudes Toward Open-Ended Blended Learning

As in any meta-analysis of a long-running project, student attitudes were extremely broad, covering the content, the delivery methods, the teaching team, their emotional responses, their learning needs, and more. Within the seven themes that we identified, however, is a common thread by which we intuit student attitudes toward open-ended learning can be understood: a tension between open-ended blended learning as a source of *empowerment*, and as a source of *anxiety*. Over and over, the same educational innovations produced both responses in different students, and through the lens of our meta-analysis we think we can begin to explain why.

The freedom to self-direct learning was appreciated by some students, and from our analysis we know that those students tended to be more motivated. While we don't know causality of that relationship (did motivation cause open-ended learning to be empowering, or did empowerment cause open-ended learning to be motivating?), we can leverage existing studies of learner motivation to make some educated guesses. The self-determination theory of motivation (SDT) (Deci and Ryan, 2012) is widely used in education contexts (Lavigne et al., 2007) and states that motivation requires autonomy (the capacity for impact), competence (the perception of ability, i.e., self-efficacy) and relatedness (the feeling of being in a community). Our open-ended creative coding model was designed, from the SDT perspective, to maximize autonomy, since it let their programs produce compelling and elaborate visual output that they could directly manipulate in code. Teaching during the pandemic highlighted the importance of relatedness (and its absence,

TABLE 7 | Sub-themes within the "Learner Struggles" theme.

Sub-theme	Example quotes
Start Paralysis	So most of the class I'm left with, like I don't know what I'm doing, or where to start. So that'd be sitting there doing nothing cause I didn't get the beginning part of it.
	Most people don't really understand what you're supposed to do with this challenge because it just shows a static image of what it's supposed to look like. They didn't understand that you loop over the circles and show a different position at each time. So that was kind of confusing
Different Learner Styles	I think doing everything online, being forced to do everything online, made it a bit more transparent in different ways that we can learn.
	Like, the tutors are great, but then to teach the knowledge that you have to someone else is very different for every student. Like, I learn better in different ways to other people.
Desire for more Engaging Instructions	The second one I had a bit more trouble with. I found that there was a lot of text dump up front, so there were lots of blocks of texts and like, I found that reading through that my brain just kind of mush and couldn't pass it quite that well.
	[cont.] - I think it's missing like a punch in, in its delivery. So, um, like summarizing it more might be applicable
Cultural Learning Differences	[If there were a lot more challenges available in CCCs, and you were able to choose which ones to do, specializing in different areas or techniques, how would you go about choosing?] - I think, like firstly, that would stress me out. Um, because like in my head, especially like coming from an Asian learning background, you've just had to- it's been ingrained that like everything that's presented to you is testable. And then I would feel like I would need you to go through all of that.
	Creative coding stressing the importance of solid practice is somehow not working for me. Practice does not necessarily equate to no-brain copying.
Lack of Confidence	P4 was sort of saying "it's okay. Like, I don't know what to do you, you can do it", whereas I feel like he had the ability to do it was probably a lack of confidence.
	I think one of the biggest things for me is like, sometimes I'll get the answer, but I don't think I'm doing it right or in the right order.
Fears Related to Failure	[on the platform] if I always fail at first, I, I don't want to begin yeah. I don't want to continue.
	[on the platform] So I think I like in order, um, like from some easy things to begin so I can get out of fear if I fail
Anxiety Over Open-Ended Platforms	I would just assume I would have to learn all of them. Because I'll get to the exam and with my luck, the random question will be the area that I didn't choose to learn more about. So I just assume I have to learn everything.
	I guess in lectures you just sit there and consume an hour's worth of information and then the tutorial, they just kind of regurgitate that information again and like you might do an activity that's almost unrelated to the lecture somewhat.
Insufficient Scaffolding in Open-Ended Tasks	There needs to be more detail and more step by step because this is the most useful for people who haven't understood code before.
	There are never enough guidelines
Study Fatigue	[On when to submit assignments] After I was done, honestly, I felt like it didn't really match the grading criteria, but one of the reasons is because I was sick of it and tired. I didn't really have enough energy to go further on
	The only thing I would say about the class time is that in the 3 h slot, like I feel like because it's so, I'm not, I haven't, I don't have background in programming, so I use so much cognitive power at the beginning that like I'm kind of, not bad, but like I'm a bit tired and foggy toward the end. And then generally toward the end is the more complicated part of what we're learning
Feeling Overwhelemed	So like I'm getting, falling further and further behind because I still don't understand a couple of weeks ago.
	I guess for me I didn't feel like I had a choice. Because I felt like we were just required to do everything to do well, if that makes sense.

isolation) on learning, and our peer learning exercises helped address this. But it was the third attribute, the perception of competence, that our analysis suggests drove the central tension between empowerment and anxiety.

We found that those students who knew where they stood, and who were comfortable being challenged, felt empowered. Those that were uncertain about their standing felt anxious, either because they were used to having "right" answers to judge their own performance, because they had a fear of failure, or because they felt they had to do everything because it might be "on the test." Choices created anxiety not because of the perceived difficulty of challenges themselves, but because they obscured traditional markers of progress or attainment that less-confident students rely on. A key takeaway from our project is that *open-ended learning can make it hard for students to understand where they are at relative to their peers or their instructors' expectations.*

The competency that we observed was not only in terms of prior programming skill: if that were the case, then perhaps our courses progressed too quickly, requiring prior exposure to succeed. Instead we saw a significant fraction of students talking in interviews about their metacognitive strategies for approaching the unit, and how those skills *in learning itself* were critical to success in our open-ended learning unit. Freedom to choose—and its inverse, the fear of not knowing where you stand—are dependent not only on your prior mastery of the

material but on your mastery of your own learning. Student-driven learning requires students to lead, and many are not equipped to do so, particularly when forced into remote learning environments.

The desire to "know where I stand" was a powerful theme throughout all our cycles of interviews. Whether it related to the choice in open-ended learning or the isolation of being a remote student during the pandemic, students struggling with learning outside their comfort zone had significant fears of failure. This can be thought of as a kind of "hierarchy of needs" for learners. If progression in your degree is at stake, you're not going to focus on enriching experiences, or to put it another way: learning doesn't matter if don't think you're going to pass.

Our recommender system prototype was a key example of this dichotomy at work: from a content appropriateness perspective our recommender was very successful, suggesting challenges that would have helped students master concepts they were struggling with. However, many students—the exception being those who were confident in their performance—did not trust that the personalized content could help them meet course-wide objective standards: in that moment they were not primarily concerned with learning, but with meeting learning objectives! It's too easy to dismiss these "grades first" attitudes as reflective of students with extrinsic motivations, but SDT suggests that intrinsic motivation can only arise after those fears of failure are addressed. These issues are not insurmountable, we feel that good design—both of learning activities and platforms—can provide support to those who are not yet possessed of the necessary confidence, while still opening up choices to those who are.

The empowerment/anxiety dichotomy we discovered aligns with prior research in the domain of self-regulated learning (SRL), where a meta-review showed that metacognitive strategies, motivation and emotional regulation were three common themes across many SRL models (Panadero, 2017). Past studies of open-ended learning environments have demonstrated similar failure cases, including the resilience of prior misconceptions (Land and Hannafin, 1997) and the inability to deploy effective information retrieval strategies (Oliver and Hannafin, 2001). This suggests a complex self-reinforcing relationship between metacognition, motivation, and competence in open-ended learning. We suggest a possible connection to similar positive feedback loops observed in studies of learner self-efficacy (Schunk, 1995), which can be reinforced by authentic positive mastery experiences (i.e., "small wins").

Given the uneven efficacy of open-ended learning, particular when classes turned remote, we found ourselves pivoting over the course of the pandemic toward supporting our students to feel confident and capable. Open-ended choice motivated students with high self-efficacy, but created anxiety among those without. Creative coding, with its open-ended design tasks, helped some students reach the self-efficacy required for them to succeed by promoting the kind of highly visible "little wins" that contribute to the enactive mastery experiences that are so critical for effective open-ended learning (Land, 2000). Asking students to design educational activities for their peers helped yet more students—particularly those from design backgrounds who were used to thinking about human-centered design tasks—but alienated those unused to thinking in that way. It was pair programming during lockdown that was the most positively-received intervention in the project, perhaps because it offered a human touchpoint. Knowing that even a single other student was struggling with the same concepts seemed to provide a sense of relatedness absent in remote learning.

The seven themes that emerged from our meta-analysis are all tied to this central tension. Furthermore, our themes explore the relationship between the empowerment/anxiety dichotomy and complicating factors like remote learning and learners operating outside of their comfort ground (like designers learning to code). Our qualitative findings support both the overall positive efficacy of open-ended learning and its failure modes in students with insufficient metacognitive strategies and motivation. In the next section we present the implications of our findings in the form of three principles for designing effective open-ended programming activities for non-CS students.

6.2. Designing Effective Blended Programming Pedagogies for Designers

The three and a half year research-through-design process we followed for this project has yielded three design insights that we think are valuable for future open-ended learning projects in HCI, particularly for non-computing students.

Where possible, design open-endedness **within, not between** learning activities. We found that creative coding challenges, where students had to apply a particular technology to an open-ended problem, to be much more effective than offering choices of which activities to do. Students with low coding self-efficacy (even those who were getting reasonable grades) found the choice of activities anxiety-inducing, especially the notion of recommended-but-not-mandatory activities. Would content in those activities be tested in the exam? Would it be necessary for the final assignment? In all cases the answer was no—otherwise it would have been a mandatory exercise—but students did not trust that, possibly due to previous educational experiences where "everything could be on the test." Particularly when dealing with students early in their degrees, the use of mandatory activities containing open-ended problems added choice while largely avoiding this phenomenon. Examples of activities with embedded open-endedness include "create a visual composition using nested arrays" or "create a design that merges stylistic elements of these two stimulus images." Care must be taken when such activities are graded, that their open-ended nature is supported with clear grading rubrics, such that even a struggling student should know when they are "done."

Even **a single other student makes remote learning better**, allowing students of all ability levels to share their struggles and achievements. Pair programming is a well-known methodology in computing education and software development practice, but it seems particularly apt to an HCI and design context. Students of design are likely to be both proficient at and receptive to collaboration, and their positive response to our synchronous peer learning exercises suggests this translates to effective learning in pairs. Working in pairs, even on challenging

tasks, was found to be more tolerable and less likely to trigger the anxiety and fear of failure we observed in students attempting our challenges alone. As proposed in the "Lightweight Teams" approach (MacNeil et al., 2016), students working together need not imply group projects that are worth a significant percentage of students' grades. We applied pair programming on in-class activities of little or no grade impact, and found that student motivation among participants in our pilot was very high. While the benefits we observed were likely magnified by the effect of the pandemic on students social lives more broadly, even outside of such extreme events many students suffer social isolation and a lack of support networks (Wu et al., 2015).

Student-led learning is human-centered design, or at least it can productively be framed as such to HCI students. Students in HCI and design degrees, especially professional degrees aimed at producing human-centered designers, are likely to respond positively to the idea of making something that helps someone else solve a problem. Where tasks can be framed as human-centered design, doing so may improve student self-efficacy. The major benefit we observed was for the student in the teaching role, confirming the "Protg effect" notion that one of the best ways to learn (or at least to master) something is to teach it. We used student-led teaching in an asynchronous and retrospective mode, with students being asked to make something that would have helped them learn, effectively designing for their past selves. Student-led teaching was also observed occurring naturally in the peer learning sessions, with (we hypothesize) similar effects.

7. LIMITATIONS OF THIS RESEARCH

Like all research-through-design, care must be taken when generalizing our findings, as they are the result of an iterative reflective practice attuned to a specific context, rather than an empirical attempt to observe population-wide facts (Zimmerman et al., 2007). Our findings should be read in the context they were generated, and the insights and design principles we draw from them are intended as suggestions for future practitioners and researchers, rather than conclusive objective truths.

Beyond the epistemological limitations, however, our study also has a number of specific scope limitations that bear mentioning. Our student population was drawn from a large Australian comprehensive research university, with about 60% of our students being Australian citizens and 40% international, primarily from Asia. The sociocultural expectations of our cohort may not align with those at other institutions worldwide. Furthermore, our course was delivered to first-year undergraduate and first-year coursework masters students, so both cohorts were in their first year of study, which may have had implications for their level of metacognitive development. Finally, this study overlapped with the second-worst pandemic in living memory, a period during which significant disruption to the tertiary education sector occurred, including border closures, stay-at-home orders and widespread layoffs in many of the industries where students work part-time. While irredeemable on a global scale, COVID-19 was a mixed blessing for this study, as while it prevented any year-over-year comparison of the efficacy of our approaches it did let us study our approaches in both blended and fully-remote contexts.

8. CONCLUSION

This project has been a unique opportunity to study the effects of different levels of technology-enhanced learning on open-ended learning pedagogies. Without the COVID-19 pandemic, we would have continued focussing on our platform to support open-ended learning a traditional face-to-face context. In that less-tragic timeline we would have likely designed both the interaction model and the learning activities of our intervention to minimize the anxiety felt by some students during open-ended learning. Instead we explored the notion of open-endedness in a much more broad set of educational contexts: face-to-face pre-pandemic, fully remote during the first wave, and hybrid after. With that exploration has come a rich understanding of the ways that open-ended learning can both empower and impair design students when they are learning to program.

At their most broad, our findings can be summarized as "open-ended learning helps some students some of the time," but to do so elides nuance. It's tempting to say that some students can "handle" freedom, while others are too focussed on their marks and grades to appreciate it, but this too is reductive: the real question is which students and why. Our findings suggest that at least one major cause for the anxiety that can arise from open-ended learning is a lack of understanding of one's own skills relative to expectations, leading to a fear of failure. Once that fear sets in, anything not directly and obviously connected to the exams or major assignments is likely to be discarded. This contrasts with the empowerment felt by the majority of the cohort when open-ended learning is successfully employed, but in order to be inclusive with our pedagogies these fears need to be addressed. We have outlined three design principles that might help do so, at least when teaching programming to design-focussed HCI students: adding open-endedness within rather than between activities, using pair programming, and appealing to students' human-centered design skills with student-led learning. These principles were derived from the iterative research-through design process we used during the CCCs project and codified through the meta-analysis of student attitudes conducted thereafter. We hope that they can help direct HCI educators in the critical task of teaching students from design backgrounds to program.

DATA AVAILABILITY STATEMENT

The datasets presented in this article are not readily available because our ethics approval prohibits sharing even de-identified interview transcripts outside of the research team. Requests to access the datasets should be directed to KG (kazjon.grace@sydney.edu.au).

ETHICS STATEMENT

The studies involving human participants were reviewed and approved by the University of Sydney Human Research Ethics Committee. The patients/participants provided their written informed consent to participate in this study.

AUTHOR CONTRIBUTIONS

AE-P developed the system and wrote the associated sections of the manuscript. BK performed the thematic meta-analysis and wrote the associated sections of the manuscript. LB co-ordinated the units into which our intervention was deployed and assisted with the writing of the manuscript. KG initiated the project, wrote the balance of the manuscript, and provided mentoring throughout. All authors contributed to the article and approved the submitted version.

FUNDING

We acknowledge the support, both financial and advisory, of the Education Innovation Team at the University of Sydney, particularly Dr Jessica Frawley. They were instrumental in the founding of the Creative Coding Challenges platform in 2018. The University of Sydney supported this research internally in several ways. The Deputy Vice Chancellor for Education supported the initiation of the project through a 2018 Strategic Education Innovation award. The School of Architecture, Design and Planning, where the authors work, supported this article's processing fees.

REFERENCES

Adams, R. S., Turns, J., and Atman, C. J. (2003). Educating effective engineering designers: the role of reflective practice. *Design. Studies* 24, 275–294. doi: 10.1016/S0142-694X(02)00056-X

Armstrong, F. (2019). "Social constructivism and action research: transforming teaching and learning through collaborative practice," in *Action Research for Inclusive Education* (London: Routledge), 5–16.

Bada, S. O., and Olusegun, S. (2015). Constructivism learning theory: a paradigm for teaching and learning. *J. Res. Method Educ.* 5, 66–70. doi: 10.9790/7388-05616670

Banning, J. (2003). *Ecological Triangulation: An Approach for Qualitative Meta-Synthesis*. What Works for Youth with Disabilities Project: US.

Batdi, V. (2017). Smart board and academic achievement in terms of the process of integrating technology into instruction: a study on the McA. *Croat. J. Educ.* 19, 763–801. doi: 10.15516/cje.v19i3.2542

Becker, B. A. (2021). What does saying that 'programming is hard' really say, and about whom? *Commun. ACM* 64, 27–29. doi: 10.1145/3469115

Bereiter, C. (1994). Constructivism, socioculturalism, and popper's world 3. *Educ. Res.* 23, 21–23. doi: 10.3102/0013189X023007021

Blikstein, P. (2011). "Using learning analytics to assess students' behavior in open-ended programming tasks," in *Proceedings of the 1st International Conference on Learning Analytics and Knowledge, LAK '11* (New York, NY|: Association for Computing Machinery), 110–116.

Bodily, R., and Verbert, K. (2017). Review of research on student-facing learning analytics dashboards and educational recommender systems. *IEEE Trans. Learn. Technol.* 10, 405–418. doi: 10.1109/TLT.2017.2740172

Bransford, J. D., Brown, A. L., Cocking, R. R., and Others (2000). *How People Learn, Vol. 11*. Washington, DC: National academy press.

Bull, S., and Kay, J. (2013). "Open learner models as drivers for metacognitive processes," in *International Handbook of Metacognition and Learning Technologies* (Ann Arbor, MI: Springer), 349–365.

Campbell, D. T., and Fiske, D. W. (1959). Convergent and discriminant validation by the multitrait-multimethod matrix. *Psychol. Bull.* 56, 81–105. doi: 10.1037/h0046016

Carbone, A., and Sheard, J. (2002). "A studio-based teaching and learning model in IT: what do first year students think?" in *Proceedings of the 7th Annual Conference on Innovation and Technology in Computer Science Education, ITiCSE '02* (New York, NY: Association for Computing Machinery), 213–217.

Carvalho, L., Goodyear, P., Wardak, D., and Saunders, R. (2014). "Peep: peer support for programing," in *The Architecture of Productive Learning Networks* (Sydney, NSW: Routledge), 97–111.

Carvalho, L., and Saunders, R. (2018). Coding, designing and networking: fostering learning through social connections. *Res. Learn. Technol.* 26:1–18. doi: 10.25304/rlt.v26.2006

Casteel, M. A., and Bridges, K. R. (2007). Goodbye lecture: a student-led seminar approach for teaching upper division courses. *Teach. Psychol.* 34, 107–110. doi: 10.1177/009862830703400208

Charleston, L. J., George, P. L., Jackson, J. F., Berhanu, J., and Amechi, M. H. (2014). Navigating underrepresented stem spaces: experiences of black women in us computing science higher education programs who actualize success. *J. Divers High. Educ.* 7, 166. doi: 10.1037/a0036632

Chase, C. C., Chin, D. B., Oppezzo, M. A., and Schwartz, D. L. (2009). Teachable agents and the protégé effect: increasing the effort towards learning. *J. Sci. Educ. Technol.* 18, 334–352. doi: 10.1007/s10956-009-9180-4

Chickering, A. W., and Gamson, Z. F. (1987). Seven principles for good practice in undergraduate education. *AAHE Bull.* 3:7.

Cohen, D. K., and Ball, D. L. (2007). Educational innovation and the problem of scale. *Scale Educ.* 1, 19–36.

Cooper, A. (1999). "The inmates are running the asylum," in *Software-Ergonomie'99* (Wiesbaden: Springer), 17–17.

Cross, N. (2011). *Design Thinking: Understanding How Designers Think and Work*. Oxford: Berg.

De Volder, M. L., De Grave, W. S., and Gijselaers, W. (1985). Peer teaching: academic achievement of teacher-led versus student-led discussion groups. *Higher Educ.* 14, 643–650. doi: 10.1007/BF00136502

Deci, E. L., and Ryan, R. M. (2012). "Self-determination theory," in *Handbook of Theories of Social Psychology, Vol. 1*, eds P. A. M. Van Lange, A. W. Kruglanski, and E. T. Higgins (Thousand Oaks, CA: Sage Publications), 416–437. doi: 10.4135/9781446249215.n21

Emara, M., Hutchins, N. M., Grover, S., Snyder, C., and Biswas, G. (2021). Examining student regulation of collaborative, computational, problem-solving processes in open-ended learning environments. *J. Learn. Anal.* 8, 49–74. doi: 10.18608/jla.2021.7230

Emara, M., Tscholl, M., Dong, Y., and Biswas, G. (2017). *Analyzing Students' Collaborative Regulation Behaviors in a Classroom-Integrated Open Ended Learning Environment*. Philadelphia, PA: International Society of the Learning Sciences.

Freeman, S., Eddy, S. L., McDonough, M., Smith, M. K., Okoroafor, N., Jordt, H., and Wenderoth, M. P. (2014). Active learning increases student performance in science, engineering, and mathematics. *Proc. Natl. Acad. Sci. U. S. A.* 111, 8410–8415. doi: 10.1073/pnas.1319030111

Froyd, J., and Simpson, N. (2008). "Student-centered learning addressing faculty questions about student centered learning," in *Course, Curriculum, Labor, and Improvement Conference, Vol. 30* (Washington DC), 1–11.

Gomes, A., and Mendes, A. J. (2007). "An environment to improve programming education," in *Proceedings of the 2007 International Conference on Computer Systems and Technologies* (Ruse), 1–6.

Greenberg, I., Kumar, D., and Xu, D. (2012). "Creative coding and visual portfolios for cs1," in *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education, SIGCSE '12* (New York, NY: Association for Computing Machinery), 247–252.

Guzdial, M. (2003). "A media computation course for non-majors," in *Proceedings of the 8th Annual Conference on Innovation and Technology in Computer Science Education* (Atlanta, GA), 104–108.

Guzdial, M. (2009). Education teaching computing to everyone. *Commun. ACM* 52, 31–33. doi: 10.1145/1506409.1506420

Hannafin, M. J. (1995). "Open-Ended learning environments: Foundations, assumptions, and implications for automated design," in *Automating Instructional Design: Computer-Based Development and Delivery Tools* (Berlin; Heidelberg: Springer Berlin Heidelberg), 101–129.

Hidayah, I., Adji, T., and Setiawan, N. (2018). "A framework for improving recommendation in adaptive metacognitive scaffolding," in *2018 4th International Conference on Science and Technology (ICST)*, (Yogyakarta: IEEE), 1–5.

Horn, M. B., and Staker, H. (2014). *Blended: Using Disruptive Innovation to Improve Schools*. San Francisco, CA: John Wiley & Sons.

Hutchinson, H., Mackay, W., Westerlund, B., Bederson, B. B., Druin, A., Plaisant, C., et al. (2003). "Technology probes: inspiring design for and with families," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (College Park, MD), 17–24.

Huynh, D., Zuo, L., and Iida, H. (2016). "Analyzing gamification of "duolingo" with focus on its course structure," in *Games and Learning Alliance* (Asahidai: Springer International Publishing), 268–277.

Jenkins, T. (2002). "On the difficulty of learning to program," in *Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences, Vol. 4* (Citeseer), 53–58.

Jeong, C. (2017). Effects of pair programming in an introductory programming course for college students: academic performance and student satisfaction. *J. Korean Assoc. Inform. Educ.* 21, 537–545. doi: 10.14352/jkaie.21.5.537

Kargarmoakhar, M., Lunn, S., Zahedi, L., Ross, M., Hazari, Z., Weiss, M. A., et al. (2020). "Understanding the experiences that contribute to the inclusion of underrepresented groups in computing," in *2020 IEEE Frontiers in Education Conference (FIE)* (Uppsala: IEEE), 1–9.

Klingner, J. K., Boardman, A. G., and McMaster, K. L. (2013). What does it take to scale up and sustain evidence-based practices? *Except. Child.* 79, 195–211. doi: 10.1177/001440291307900206l

Krouska, A., Troussas, C., and Virvou, M. (2018). "Computerized adaptive assessment using accumulative learning activities based on revised bloom's taxonomy," in *Joint Conference on Knowledge-Based Software Engineering* (New York, NY: Springer), 252–258.

Kuhn, S. (1998). The software design studio: an exploration. *IEEE Softw.* 15, 65–71. doi: 10.1109/52.663788

Kuhn, S. (2001). Learning from the architecture studio: implications for project-based pedagogy. *Int. J. Eng. Educ.* 17, 349–352.

Land, S. M. (2000). Cognitive requirements for learning with open-ended learning environments. *Educ. Technol. Res. Dev.* 48, 61–78. doi: 10.1007/BF02319858

Land, S. M., and Hannafin, M. J. (1997). Patterns of understanding with open-ended learning environments: a qualitative study. *Educ. Technol. Res. Dev.* 45, 47–73. doi: 10.1007/BF02299524

Lavigne, G. L., Vallerand, R. J., and Miquelon, P. (2007). A motivational model of persistence in science education: a self-determination theory approach. *Eur. J. Psychol. Educ.* 22, 351–369. doi: 10.1007/BF03173432

Logan, B. (2015). Deep exploration of the flipped classroom before implementing. *J. Instruct. Pedagogies.* 16, 1–16. Available online at: https://www.aabri.com/jip.html

MacNeil, S., Latulipe, C., Long, B., and Yadav, A. (2016). "Exploring lightweight teams in a distributed learning environment," in *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (Charlotte, NC), 193–198.

McBroom, J., Yacef, K., and Koprinska, I. (2020). "Scalability in online computer programming education: automated techniques for feedback, evaluation and equity," in *Proceedings of the 13th International Conference on Educational Data Mining (EDM 2020)* (Sydney, NSW), 802–805.

McCarthy, L., Reas, C., and Fry, B. (2015). *Getting Started With P5. js: Making Interactive Graphics in JavaScript and Processing*. San Francisco, CA: Maker Media, Inc.

Nassrallah, Z., Frankfurt, M., and Hill, R. V. (2018). A student' centered, active learning approach to teaching spinal cord anatomy. *FASEB J.* 32. doi: 10.1096/fasebj.2018.32.1_supplement.lb510

Nielsen, L. (2012). Five reasons i'm not flipping over the flipped classroom. *Technol. Learn.* 32, 46–46.

Oliver, K., and Hannafin, M. (2001). Developing and refining mental models in open-ended learning environments: a case study. *Educ. Technol. Res. Dev.* 49, 5–32. doi: 10.1007/BF02504945

Panadero, E. (2017). A review of self-regulated learning: six models and four directions for research. *Front. Psychol.* 8, 422. doi: 10.3389/fpsyg.2017.00422

Paniagua, A., and Istance, D. (2018). *Teachers as Designers of Learning Environments: The Importance of Innovative Pedagogies. Educational Research and Innovation*. OECD Publishing. OECD Publishing. 2, rue Andre Pascal, F-75775 Paris Cedex 16. Available online at: http://www.oecd.org.

Pearson, P. D., and Gallagher, G. (1983). The gradual release of responsibility model of instruction. *Contemp. Educ. Psychol.* 8, 112–123.

Peckham, J., Harlow, L. L., Stuart, D. A., Silver, B., Mederer, H., and Stephenson, P. D. (2007). Broadening participation in computing: issues and challenges. *ACM SIGCSE Bull.* 39, 9–13. doi: 10.1145/1269900.1268790

Penuel, W. R., Fishman, B. J., Haugan Cheng, B., and Sabelli, N. (2011). Organizing research and development at the intersection of learning, implementation, and design. *Educ. Res.* 40, 331–337. doi: 10.3102/0013189X11421826

Rajaravivarma, R. (2005). A games-based approach for teaching the introductory programming course. *SIGCSE Bull.* 37, 98–102. doi: 10.1145/1113847.1113886

Ramalingam, V., LaBelle, D., and Wiedenbeck, S. (2004). "Self-efficacy and mental models in learning to program," in *Proceedings of the 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (Leeds), 171–175.

Reas, C., and Fry, B. (2006). Processing: programming for the media arts. *Ai Soc.* 20, 526–538. doi: 10.1007/s00146-006-0050-9

Reas, C., and Fry, B. (2007). *Processing: A Programming Handbook for Visual Designers and Artists*. Cambridge, MA: Mit Press.

Reimer, Y. J., and Douglas, S. A. (2003). Teaching hci design with the studio approach. *Comput. Sci. Educ.* 13, 191–205. doi: 10.1076/csed.13.3.191.14945

Ricci, F., Rokach, L., and Shapira, B. (2011). "Introduction to recommender systems handbook," in *Recommender Systems Handbook* (Negev: Springer), 1–35.

Robinson, B., and Schaible, R. M. (1995). Collaborative teaching: reaping the benefits. *College Teach.* 43, 57–59. doi: 10.1080/87567555.1995.9925515

Rohrbeck, C. A., Ginsburg-Block, M. D., Fantuzzo, J. W., and Miller, T. R. (2003). Peer-assisted learning interventions with elementary school students: a meta-analytic review. *J. Educ. Psychol.* 95, 240. doi: 10.1037/0022-0663.95.2.240

Roll, I., Aleven, V., McLaren, B. M., and Koedinger, K. R. (2007). Designing for metacognition–applying cognitive tutor principles to the tutoring of help seeking. *Metacogn. Learn.* 2, 125–140. doi: 10.1007/s11409-007-9010-0

Schön, D. A. (1979). *The Reflective Practitioner*. New York, NY: Basic Books.

Schön, D. A. (1987). *Educating the Reflective Practitioner: Toward a New Design for Teaching and Learning in the Professions*. San Francisco, CA: Jossey-Bass.

Schunk, D. H. (1995). "Self-efficacy and education and instruction," in *Self-Efficacy, Adaptation, and Adjustment* (New York, NY), 281–303.

Su, X., and Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Adv. Artif. Intell.* 2009, 421425. doi: 10.1155/2009/421425

Troussas, C., Giannakas, F., Sgouropoulou, C., and Voyiatzis, I. (2020). Collaborative activities recommendation based on students' collaborative learning styles using ANN and WSM. *Interact. Learn. Environ.* 1–14. doi: 10.1080/10494820.2020.1761835

Tversky, B. (2015). "On abstraction and ambiguity," in *Studying Visual and Spatial Reasoning for Design Creativity* (New York, NY: Springer), 215–223.

van Popta, E., Kral, M., Camp, G., Martens, R. L., and Simons, P. R.-J. (2017). Exploring the value of peer feedback in online learning for the provider. *Educ. Res. Rev.* 20:24–34. doi: 10.1016/j.edurev.2016.10.003

Vaughan, N. D., Cleveland-Innes, M., and Randy Garrison, D. (2013). *Teaching in Blended Learning Environments: Creating and Sustaining Communities of Inquiry*. Alberta: Athabasca University Press.

Vygotsky, L. S. (1930–1934/1978). *Mind in Society: The Development of Higher Psychological Processes*. Cambridge, MA: Harvard University Press.

Wang, T. (2010). A new paradigm for design studio education. *Int. J. Art Design Educ.* 29, 173–183. doi: 10.1111/j.1476-8070.2010.01647.x

Wiebe, E., Williams, L., Petlick, J., Nagappan, N., Balik, S., Miller, C., and Ferzli, M. (2003). "Pair programming in introductory programming labs," in *Proceedings Submitted to American Society for Engineering Education Annual Conference and Exposition, Vol. 2003*. Raleigh, NC: Researchgate.net.

Wilson, B., and Lowry, M. (2000). Constructivist learning on the web. *New Dir. Adult Contin. Educ.* 2000, 79–88. doi: 10.1002/ace.8808

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.* 366, 3717–3725. doi: 10.1098/rsta.2008.0118

Wu, H.-P., Garza, E., and Guzman, N. (2015). International student's challenge and adjustment to college. *Educ. Res. Int.* 2015, 202753. doi: 10.1155/2015/202753

Zimmerman, J., Forlizzi, J., and Evenson, S. (2007). "Research through design as a method for interaction design research in HCI," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '07* (New York, NY: Association for Computing Machinery), 493–502.

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.