HCI education reflects the continual evolution of HCI, embracing the changing landscapes of technology, infrastructure, and technology use. This forum aims to provide a platform for HCI educators, practitioners, researchers, and students to share their perspectives, reflections, and experiences related to HCI education. **— Sukeshini Grandhi, Editor**

Coding as a Social and Tangible Activity

Olivia L. Tabel, Jonathan Jensen, Martin Dybdal, and **Pernille Bjørn,** University of Copenhagen

The centrality of technology places great responsibility on HCI education researchers. It's our duty to explore strategies for teaching children the skills, knowledge, and expertise necessary to continue shaping our society through technology. As part of this effort, teaching kids how to code is fundamental.

Coding Pirates (https://codingpirates.dk) is a Danish nonprofit social organization dedicated to doing just that. While Coding Pirates is a successful initiative, it is difficult to attract participants with diverse preferred-learning strategies. Take this example: In early fall 2016, one of the authors brought her 13-year-old daughter to a Coding Pirates event. During the two hours, she developed her first computer game in the language Scratch, which she proudly displayed. However, when asked to join the Coding Pirates event the following week, she replied that she did not find it fun to sit alone, but instead preferred to collaborate with others. In other words, even though she'd participated in a group event, she did not interpret what she'd done as collaborating with others.

While you could claim that this was an individual case, our previous experiences as volunteers during Coding Pirates events confirmed that children primarily worked individually. Thus, our research began to focus on how to design a workshop format for the Coding Pirates that would explicitly foster collaboration and interaction across participants so as to attract and retain new types of inclusive participation—and cater to children who prefer socially organized learning strategies.

Here we present our educational initiative for introducing newcomers to programming, which we designed and conducted for Coding Pirates in the fall of 2016. Our initiative is grounded in HCI strategies for social interaction and tangible experiences. Based upon our findings, we argue that reframing coding as a social and tangible activity fundamentally transforms the coding activity in a way that could attract young children from various backgrounds to the field of computing, thus fostering increased inclusion through early introduction to programming.

CODING PIRATES

Coding Pirates seeks to foster a creative environment for children between 7 and 17 years old across ethnicities, gender, and culture to learn programming through engagement. Programming is taught by volunteers in 65 branches located in schools, universities, and libraries across Denmark. There are close to 700 volunteer teachers from diverse backgrounds and approximately 1,340 Coding Pirates, children who attend weekly activities across Denmark. In 2015, computer science graduates from the Department of Computer Science at the University of Copenhagen (DIKU) established a Coding Pirates branch; today DIKU Coding Pirates runs weekly workshops for an entire school year (August to June). The workshops each typically have 30 participants with 11 volunteer teachers; 340 children are currently on the waiting list.

Initial observations of the Coding Pirates workshops took place in October 2016, first at a public library in Copenhagen, where we observed a workshop with 11 children and four teachers, and then at the DIKU branch, with 44 children and 10 teachers. These observations confirmed that children mainly worked individually, with little to no social interaction with peers during workshops. Children were paying attention to their own devices, the overhead projector, or the step-by-step guide pamphlet created by Coding Pirates teachers. While the volunteer teachers have excellent technical expertise and an amazing commitment to sharing their technical knowledge, they lacked pedagogical training and thus paid less attention to the social dynamics of the workshop.

Insights

- → Conceptually approaching programming as a social and tangible human-computer activity is an opportunity to develop new ways of teaching programming and technology development, potentially fostering inclusion.
- → When designing teaching activities, we must consider how the social context we create matters for the learning experience.

IMAGE COURTESY OF CODING PIRATES

WHAT DID WE DO?

In close collaboration with DIKU Coding Pirates, we designed two experimental workshops. The first focused on introducing coding as a social activity, while the second introduced coding as both a social and a tangible activity. The first workshop ran on November 22, 2016, and the second workshop ran a week later, on November 29. Both workshops took place between 5 p.m. and 7 p.m. on a weekday at the university campus in Copenhagen. For each workshop, 10 participants were recruited from the DIKU Coding Pirates waiting list. In total, 20 children participated. By recruiting from the waiting list, we ensured that no participants had previous experience with Coding Pirates events. The recruitment did not take into account, however, whether children had prior experience with coding from other situations (school, home, etc.). Our participants were between 8 and 14 years old; they were 60 percent male and 40 percent female. The workshops were led by four teachers: the first, second, and third author, and one additional DIKU Coding Pirates volunteer.

Programming as social interaction. The programming curriculum of our workshops was the same as in the regular introductory Coding Pirates workshops. It included an introduction to Scratch, a programming language and graphic programming interface known to help novice programmers learn to code [1]. However, we used it with two distinct innovations: We introduced pair programming, which articulates programming as a social activity (workshop 1), and we extended the curriculum beyond Scratch (workshop 2) by adding tangible programming experiences using Makey Makey [2].

Pair programming was our major intervention. We used it to demonstrate coding as a social activity. By introducing the distinct roles in programming—the navigator and the driver [3]—we were able to manifest coding as a socially embedded activity supporting collaborative learning [4]. The navigator was responsible for analyzing the situations and making suggestions on how to code, while the driver was responsible for typing and writing the code. To demonstrate the differences between the roles, we created a car-front mock-up used to role-play pair programming. To further distinguish between roles, two sets of hats (10 yellow and 10 black hats) were distributed to the teams, which meant that all drivers were wearing yellow hats and all navigators were wearing black hats (Figure 1).

We used pair programming to demonstrate coding as a social activity.

Participants were divided into five teams of two, with different-colored hats. The hats made it easier for the participants to know their role; it also made the role division visible across the teams. During the event, the teachers asked the pairs to swap roles and hats, which made the shift in roles clear.

The task was to create a game. However, instead of spending time introducing all the features of the programming environment, we designed the task using the fading example, a scaffolding technique in

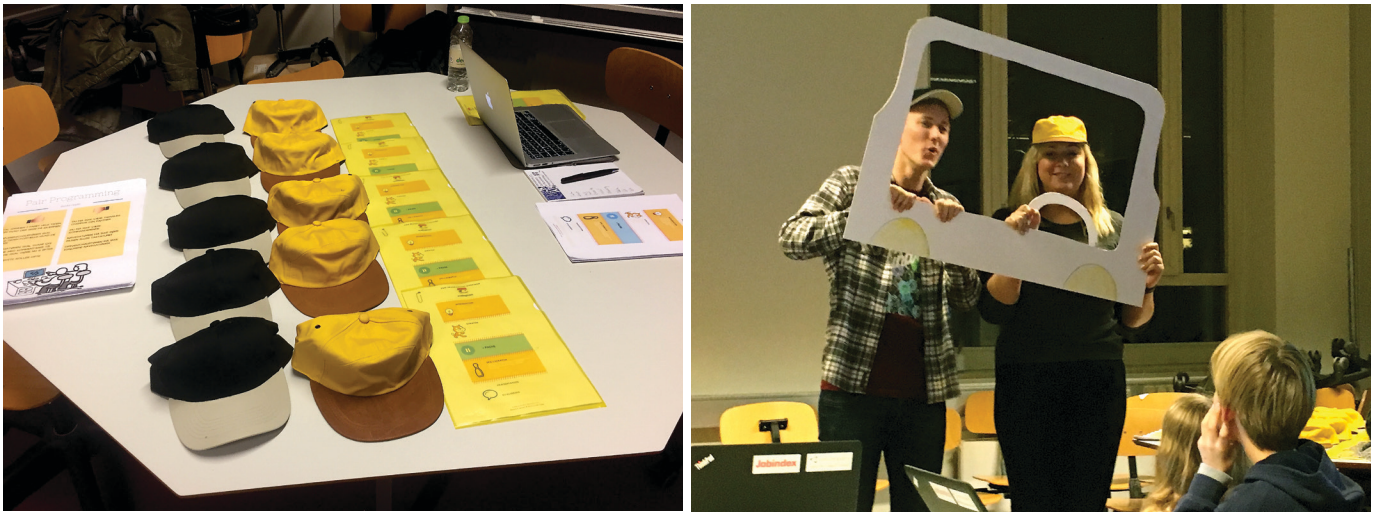

Figure 1. Introducing pair programming using yellow and black hats to represent drivers and navigators.

which tasks are designed to gradually advance as the task is resolved [1]. A plenum brainstorming activity was done to foster creativity across all participants, identifying game-theme ideas. After brainstorming, each team decided on a theme and created an appropriate background on the template game. All tasks during the workshop were open-ended yet had precise guidelines for how to proceed.

They were given instructions on how to measure coordinates, as well as how to move digital artifacts on the screen to identify the correct placement according to the theme. As the workshop continued, further functionality such as sound and "coding blocks" was touched on, and participants were then asked to edit their games accordingly. For example, they were asked to create an "enemy" that matched their theme, which, when clicked on, would remove points from the player. All teams managed to create a fully working game, which they showcased to each other at the end of the workshop (Figure 2). All teams went through the complete curriculum.

Programming as tangible interaction. We wanted to increase the complexity of the curriculum by asking participants to alter not only the game design but also the functionality, by adding motion through dynamic modifications. In the second workshop, we invited 10 new participants. Instead of providing only the game template, we added three arrow buttons to the template (up, left, and right), with a missing down arrow. After deciding on the theme, background, and so on, participants were asked to create the missing arrow by exploring the code from the other arrows, creating modifications within the coding script.

We also extended the curriculum to include programming as a tangible experience. We introduced participants to Makey Makey as a way to extend their experience of computation. What makes Makey Makey ideal in this situation: It functions as an extension to the keyboard, and so by plugging the Makey Makey into the computer, a Coding Pirate can design and build a physical controller for their game without changing any code. The arrows are standard in the Makey Makey design, so the arrows within the game could be directly connected to the physically designed controllers. By manifesting programming as tangible interaction, we challenged the participants to experience programming in concrete ways. And by introducing physical manipulation into the world of

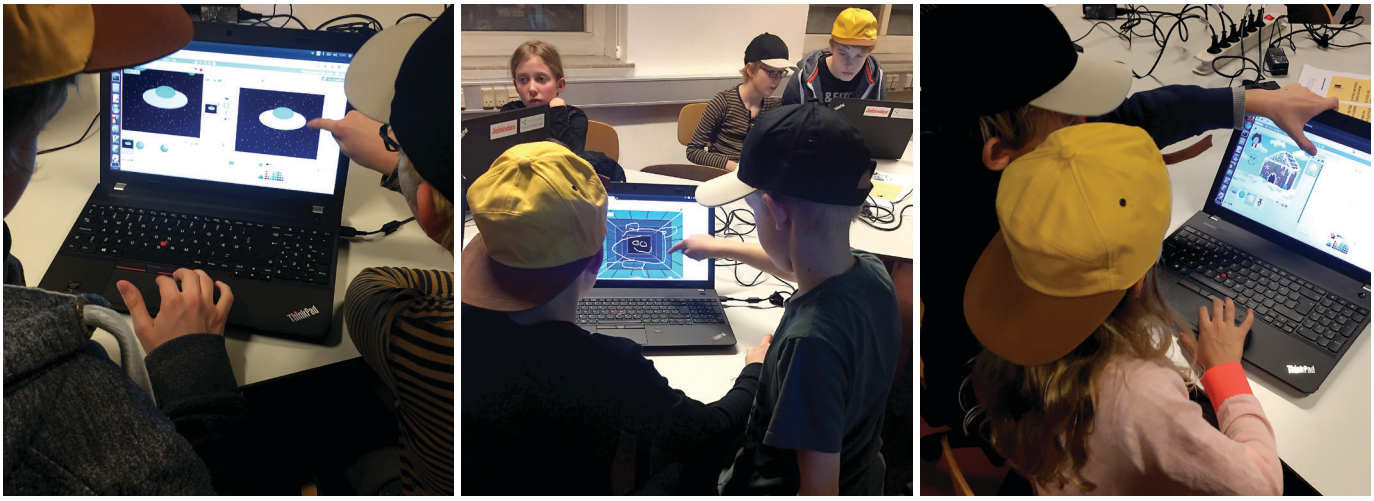

Figure 2. Students developing their games. From left: UFO game, Maze game, Xmas game.

programming, we allowed them to experience a connection between the physical world and digital world, exploring new creative interactions using programming beyond the computer screen. What's more, by applying wires with clips, they experienced programming in "real life." As in the first workshop, all teams managed to create their games, complete the extended curriculum, and present their games to other workshop participants.

WHAT DID WE LEARN?

Our interventions, introducing programming as a social and tangible activity, changed the dynamic of the Coding Pirates workshops. We saw how the social dynamic was distinctly different in our workshops compared with previous workshops. Including social interaction as part of the workshop design may hold potential for attracting and retaining more diverse participation (e.g., diversity in gender and social background), as it may change the perception of what programming entails, thus potentially retaining participants such as the 13-year-old girl mentioned earlier.

Our findings go beyond the agenda of inclusion. One of the challenges experienced by Coding Pirates teachers is that participants feel anxiety and are reluctant to present and share their ideas and accomplishments in front of the whole class. Indeed, it can be demanding to stand in front of others and present what you have created. Interestingly, this was not an issue during our workshops. Instead, our participants expressed a desire to share and present their accomplishments to fellow participants and referred to their work as a shared and joined task instead of an individual performance. For example, they said, "I wouldn't mind presenting in front of everybody in the class" and "You have someone to back you up if you forget something," which accentuates their perception of coding as collaborative and social. The social and tangible approaches applied to programming seem to accelerate and stimulate participants' eagerness to display their digital products and help others. In particular, a quote from the second workshop stands out. A participant was frustrated and expressed giving up, somewhat loudly, when another participant yelled back in encouragement, "You just need to continue! It gets much more fun soon!" The first participant managed to overcome his frustrations and succeed.

Both workshops achieved great success with regard to collaboration within teams and within the whole class. While the hats were originally designed to explain the roles in pair programming, they turned out to have a far stronger impact than first anticipated. The hats became a manifestation of the social coherence across the class. Expressions such as "Doesn't your navigator listen to you either?" being shared in a playful manner across the room, addressing participants wearing similar hats, demonstrated how each participant was part of a team, a sub-group, and the class simultaneously. Being able to monitor how other participants collaborate, while experiencing visual similarity, facilitated social interaction. The experience of belonging further supported participants in having confidence in sharing ideas and helping out others. In both workshops, the social interaction was central, not only because of pair programming, but also due to the inclusive social characteristic of the tasks and exercises. Two participants exchanged information after the workshop and agreed to continue their work. Finding new ways to create age-appropriate computational thinking [5] is important. Our interventions facilitated the experience of programming as socially embedded, and Coding Pirates were able to transform everyday objects into computational artifacts.

When conceptually approaching introduction to programming as social and tangible human-computer activity, we have the potential to develop new ways for teaching programming that foster inclusion. We argue that strategies of moving beyond the keyboard and screen, including considerations of the socially embedded context, are relevant for programming at various levels of education and participation—high school, college, or university. By creating new classroom experiences that foster engagement for computational thinking in the long term, we can promote increased participation in designing digital technologies and shaping society for all.

Endnotes

1. Resnick, M., Maloney, J., Monroy-Hermandez, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., and Kafai, Y. Scratch: Programming for all. *Communications of the ACM 52*, 11 (2009), 60–67.
2. Beginners' Mind Collective and Shaw, D. Makey Makey: Improvising tangible and nature-based user interfaces. *Proc. of the Sixth International Conference on Tangible, embedded and Embodies Interaction*. ACM, New York, 2012.
3. Werner, L. and Denning, J. Pair programming in middle school. *Journal of Research on Technology in Education 42*, 1 (2009), 29–49
4. Bjørn, P. and Hertzum, M. Project-based collaborative learning: Negotiating leadership and commitment in virtual teams. *Proc. of the 5th Conference on Human Computer Interaction in Southern Africa*. ACM, 2006.
5. Wing, J.M. Computational thinking. *Communications of the ACM 49*, 3 (2006), 33–35.

Olivia L. Tabel is finishing her bachelor's degree in communication and IT at the University of Copenhagen, currently at UC Santa Barbara.
→ lindertabel@gmail.com

Jonathan Jensen is finishing his bachelor's degree in communication and IT at the University of Copenhagen.
→ jona2412@gmail.com

Martin Dybdal established the DIKU branch of Coding Pirates and is a Ph.D. student in the Department of Computer Science (DIKU) at the University of Copenhagen.
→ dybber@dybber.dk

Pernille Bjørn is a professor of computer-supported cooperative work in the human-centered computing section in the Department of Computer Science (DIKU) at the University of Copenhagen.
→ pernille.bjorn@di.ku.dk

DOI: 10.1145/3137099