Seeds of a Computer Culture: An Archival Analysis of Programming Artifacts from a Community Technology Center

Yasmin B. Kafai, Kylie A. Peppler, Mabel Alavez, & Omar Ruvalcaba
University of California, Los Angeles 2331 Moore Hall Los Angeles, CA 90095
kafai@gseis.ucla.edu, peppler@gmail.com, m_1080@yahoo.com, indigo@ucla.edu

Abstract: We examined the genre and complexity of computer programs created by members of a community technology center. We collected 240 projects during a six-month period. The program genres reflected animations, games, and graphics in about equal numbers. While the number of computer programs decreased over time, the number of advanced programs remained constant suggesting that members either abandoned programs or paired up with more experienced programmers and continued to develop more complex programs.

About 25 years ago, Seymour Papert (1980) described the necessity of creating computer cultures rather than isolated experiences to learn with and about technology. He defined a computer culture as a place promoting access to technology fluency — in contrast to computer literacy — and by emphasizing technology production and personal expression as essential catalysts for learning. As the history of school classrooms has shown the creation of such cultures has proven to be a challenging enterprise, in particular what concerns programming activities. Even in informal learning environments such as community technology centers dedicated to technology fluency, computer programming has rarely become part of design activities.

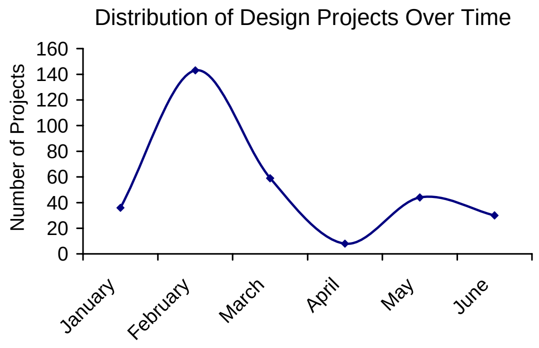
Our research was situated in a Los Angeles community technology center (CTC) visited by a predominantly Latino/a and African-American youth ages 8-18. The center, where children and youth are considered "members" (as opposed to students) of a learning community, encourages them to devise multi-media, multi-application activities that are founded upon their personal interests (Resnick, Rusk, & Cooke, 1998). We introduced Scratch, a new programming environment oriented towards media production (Resnick, Kafai & Maeda, 2003). In Scratch, programmers do not need to write program code; rather they select and manipulate blocks to create scripts that control objects or characters on the screen. These blocks also facilitate manipulation of existing media such as imported graphics from the Internet or creation of videos, animations, and music. For this poster we will focus on the programming projects created by CTC members over the course of the six months. We considered these projects to be potential seeds, or indicators, of a computer culture that would tell us about members' interest in programming and their development of programming skills.

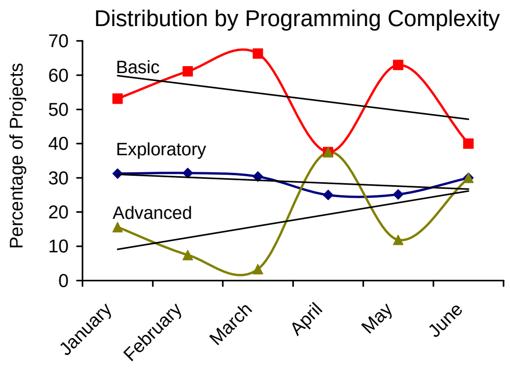
Methods

A member of the research team collected, on a regular basis, all of the Scratch projects stored on the central server by doing a file search for files with .scratch extensions. By collecting projects on a weekly basis we were able to track the number of projects and possible progress of individual projects, gathering multiple copies of a single project that had been reworked over a longer period of time. For the analysis, we took screenshots of program graphics and code and entered them into a spreadsheet along with short descriptions of content and functionality. In addition, we noted the name, gender, and age of programmer (if known) and possible collaboration with a mentor. Programs were coded into four categories based on project type (animation, game, story, graphics, and other) and classified into three levels of programming complexity: *exploratory* with no programming and only graphics, *basic* with simple programming scripts and of short length, and *advanced* with programming structures such as looping, conditionals, and random.

Findings

Over the course of six months, we collected 240 programming projects created by members of the CTC, some designed alone others with mentors. We found that 44% of these projects fell into the category animations with and without user manipulation, followed by 23% of graphics projects, and 15% of game projects focusing on fighting, sports and adventure; 14% or 34 projects escaped a clear categorization because they did not provide enough detail. In terms of program complexity, 30% or 72 projects were considered exploratory, 59% of the projects fell into the basic category, followed by 11% advanced projects. A longitudinal analysis revealed that over the time period of six months the total number of program projects decreased (see Figure 1a) while the number of advanced projects remained constant (see Figure 1b).





Figures 1a and 1b. The left graph illustrates the distribution of designed projects from the start of introducing Scratch (in January) with reaching a peak in February and a slow down in the second quarter from April to June. The right describes the distribution of projects in terms of programming complexity with the lines capturing general trends.

Discussion

We realize that this archival analysis of programming artifacts provides us only with a partial access to a computer culture for multiple reasons: to begin with, our archive while extensive did not capture all Scratch program designed but only those saved. The archive does not tell us what motivated CTC members to create their programs, what they value in their designs, and how they compare them to their other design projects. We also could not address the equally important social and local influences at work that contributed to the design of the programs. Notwithstanding these limitations, the large number of Scratch programs provides a compelling example that members were active in creating numerous programs over an extended period of time and that even without explicit curricular goals, grades or instruction. More importantly, the complexity of programs created remained constant while the total number of program projects decreased over time suggesting at least several explanations: members who generated exploratory programs dropped out of the CTC, members decided to team up with more experienced members or mentors to work on programs, or members developed their exploratory and basic programs into more complex projects. Our next steps will be to construct case studies for a select number of Scratch programs and to collect information from field notes about the design process and context and to conduct interviews with members about their projects and about programming in general.

References

Papert, S. (1980). *Mindstorms*. New York: Basic Books.

Resnick, M., Kafai, Y. B., & Maeda, J. (2003). *A Networked, Media-Rich Programming Environment to Enhance Technological Fluency at After-School Centers in Economically-Disadvantaged Communities*. Proposal (funded) to the National Science Foundation: Arlington, VA.

Resnick, M., Rusk, N., and Cooke, S. (1998). Computer Clubhouse: Technological fluency in the inner city. In D. Schoen, B. Sanyal, and W. Mitchell (eds.), *High Technology and Low-Income Communities*. Cambridge, MA: MIT Press.

Acknowledgments

We would like to thank Shiv Desai and Grace Chiu for their participation in research activities. The work reported in this poster is conducted in collaboration with Mitchel Resnick’s research group at the MIT Media Lab and supported by a grant of the National Science Foundation (NSF-0325828). The views expressed are those of the authors and do not necessarily represent the views of the supporting funding agency or the University of California, Los Angeles.