Education

Human-Centered Computing: A New Degree for Licklider’s World

Combining computing and psychology, J.C.R. Licklider’s prescient ideas are being applied in contemporary educational settings.

In the 1960s, J.C.R. Licklider described his vision for the future of computing, which is remarkably like today’s world. He saw computing as augmenting human intelligence, and for communications among communities. He foresaw cloud computing and the Semantic Web. Licklider’s background was different than many of the early computer scientists. He was not an electrical engineer or primarily a mathematician—his degrees were mostly in psychology.

To predict today’s world took a combination of computing and psychology. It is not surprising that understanding today’s world of ubiquitous computing requires a blend of computing and social science. The phenomena of social computing are not primarily about technology. What is interesting about our modern computing milieu is the blend of technology, humans, and community. Human-centered computing is a new subdiscipline of computer science that prepares students for studying our socio-technical world.

Georgia Tech, Clemson University, and the University of Maryland, Baltimore County all offer graduate degrees in Human-Centered Computing (HCC). Students in Georgia Tech’s HCC Ph.D. program work in areas like human-computer interaction (HCI), learning sciences and technologies, and cognitive science and AI. They use methods from social and behavioral sciences as well as engineering. While HCI focuses on the boundary (the interactions) between computing and humans, HCC places humans (as individuals and in societies) at the center of the research. HCC might lead to designs for new software, but it can also help us to understand what emerges from the world that Licklider predicted.

Georgia Tech’s HCC degree program prepares students to design technology, to understand humans and societies, and to study what emerges when that technology is ubiquitous. HCC at Georgia Tech has a core of three courses. The foundational course gives students theories that can be applied to understand human behavior with technology. The technology course ensures all HCC students can build prototypes of interactive systems to demonstrate their ideas. The third course ties together the threads to establish research themes. An annual seminar...
engages students in discussion about recent literature that relates to HCC.

For me, HCC is an excellent way to prepare computing education researchers. People have always developed theories of how their worlds work, from ancient mythology to modern science. How are people explaining their computerized world to themselves? How can we help them develop a better and more useful understanding? What gets in the way of that understanding? Answering these kinds of questions requires knowledge of computer science (if only to recognize correct from incorrect understandings), but also of how people learn and how to study humans and their learning.

Not all HCC students address issues of computing education research, but even when that is not the explicit focus, HCC research often offers lessons about how people learn about computing. People always learn, and in a world filled with computing, that is often what they are learning about—though not always well, clearly, or efficiently. Here are stories of three HCC graduates whose dissertations inform us about how people learn about computing.

Reframing the Computing

Betsy DiSalvo (now an assistant professor at Georgia Tech) starts from an interesting observation. Many computer scientists (who are mostly white or Asian, and male) say they became interested in computing because of video games. No demographic group plays more video games than African-American and Hispanic teenagers and men. But few African-American and Hispanic males become computer scientists. Why was that?

DiSalvo explored her question with ethnographic methods. She observed African-American teen males playing video games and talked to them about how and why they played. She found they were playing video games differently than white teen males. Her participants never used “cheat codes” or modified their games in any way—they used video games like athletic competition. Manipulating the football or the field is cheating, so why would you change the video game? She used design research activities to explore how different ways of describing computing would make the technology more salient while still appealing to the audience she wanted to attract.

DiSalvo built the Glitch Game Testers project. Glitch successfully engaged African-American teen males in computer science by training and hiring them as game-testers. Game-testers must see video games as a technology with flaws. Glitch students learned computer science, motivated to become better testers. Glitch attracted students who loved video games, and kept them involved because it was a paying job. Most of her students went on to post-secondary computing education.

DiSalvo designed Glitch through a human-focused design process. She did not design a technology. She designed a new way for her students to think about computing. Her design research activities explored different ways of describing computer science with different lenses. Through that iterative design process, she found a reframing that could change who builds computing in the future.

Impact of Not Understanding Computing

Unlike science or mathematics, undergraduates often come to computer science with a poor understanding of what computer science is. Mike Hewner (now an assistant professor at Rose-Hulman Institute of Technology) wanted to know what the impact of that misunderstanding of computer science had on students who chose to major in computer science. Hewner interviewed 33 students at three different universities. He used a social science method called grounded theory to identify themes, create abstractions, and eventually come to a well-supported understanding of how CS majors make educational decisions.

Hewner found plenty of misunderstandings of computer science, like the two sophomore CS majors who told him computer graphics was the study of Photoshop. He also found more subtly different conceptions of computer science. There were the students who saw that computer science was the study of theory, and software engineering was “lower tier.” There were the students who saw that programming was “the end goal behind computer science,” and theory was in support of programming.

The biggest surprise in Hewner’s study was that these different conceptions did not significantly influence students’ educational choices. Few students that Hewner interviewed could even predict what would be in the next classes they took. Even when faced with choosing among specializations within the degree, students told Hewner the choice did not really matter, prompting the comment “I think I would have the same number of jobs available if I took either of them.” The students did not have a clear understanding of what jobs looked like in computer science, and consequently, they did not make choices in order to prepare themselves for a future job.

Hewner found his students used enjoyment as a proxy for affinity for a subject. Students would explore their interests through their classes. When they found something they really enjoyed, they chose that as a sign of their affinity for the subject. Hewner found students did not reason deeply about why they enjoyed a course (or not). A bad teaching assistant or an 8 a.m. lecture might lead the students to dislike a course, and that would be enough to cause students to switch majors to a course they enjoyed more. Once students committed to a major, they simply trusted the curriculum—and were willing to persist through difficult classes, once they had made the commitment.

Hewner drew on his deep understanding of computer science to make sense (and identify nonsense) in the students’ conceptualizations. His methods were drawn directly from social sciences. His findings help us to
The modern computing world is not just designed. It emerges from people working with it.

Did you know that you can now order many popular ACM conference proceedings via print-on-demand?

Institutions, libraries and individuals can choose from more than 100 titles on a continually updated list through Amazon, Barnes & Noble, Baker & Taylor, Ingram and NACSCORP: CHI, KDD, Multimedia, SIGIR, SIGCOMM, SIGCSE, SIGMOD/PODS, and many more.

For available titles and ordering info, visit: librarians.acm.org/pod

Better understand the process of preparing the students who will one day create future computing.

What Should Laypeople Understand About Computing?

Even the HCC students who study HCI topics can inform us about computing education. Erika Poole (now an assistant professor at Pennsylvania State University) is one of those. Poole studied how families attempting a variety of technology-related challenges, as a way of discovering how they sought out help.

The families in Poole’s study did much worse than I might have guessed. For example, only two of 15 families were able to configure their wireless router. Some of the reasons were because of awful user interfaces (for example, a virtual keyboard that was missing some keys). But other tasks were challenging for more subtle reasons.

One of the challenges in Poole’s study involved editing a Wikipedia page. When Poole’s participants edited a Wikipedia page for the first time, without an account, they saw this warning message: “You are not currently logged in. Editing this way will cause your IP address to be recorded publicly in this page’s edit history. If you create an account, you can conceal your IP address and be provided with many other benefits. Messages sent to your IP can be viewed on your talk page.”

Poole’s participants had to decide if “recording publicly” their “IP address” was a problem. One participant told Poole the process of editing the encyclopedia and reading this warning message made her “feel like a criminal.” She canceled her changes.

Another participant contacted his “computer savvy” friend to help interpret the message. The friend warned him that having your IP address recorded was dangerous. This participant also gave up.

Editing Wikipedia is a natural act in Licklider’s world. Should we expect people who edit Wikipedia to know what an IP address is? Is it a user interface mistake to expect Wikipedia contributors would understand the implications of publicly recording an IP address? While the Internet is in common use in the homes of all of Poole’s participant families, the families (and even some of their “computer savvy” friends) clearly did not understand some of the basic terms and concepts of the technology they use.

Poole’s study provides some concrete examples of what people understand, and misunderstand, about the computing in their lives. Certainly, she informs HCI designers, in thinking about expectations of user knowledge. On the other hand, an IP address is part of our modern world. Is it better to hide it, or explain it? Her study informs general education designers in thinking about what everyone needs to know about computing.

Understanding Licklider’s World

J.C.R. Licklider was able to predict much of our modern computing world because he combined his understanding of computing with his understanding of people. To understand the world he predicted, we need researchers who understand computing and who can use the methods from Licklider’s psychology (and sociology and other social sciences, too).

The modern computing world is not just designed. It emerges from people working with it, trying to understand and use it (typically, in ways different than originally designed), and interacting with millions of others doing the same things. We have to study it as it is, not just as how we meant it to be.

References


Mark Guzdial (guzdial@cc.gatech.edu) is a professor in the College of Computing at Georgia Institute of Technology in Atlanta, GA.

Copyright held by author.