2016-04-16

What directs and sustains students’ early-stage software design work?

“To date,” wrote Petre, van der Hoek, and Baker in 2010, “not enough is known about the formative stages of program design….What do designers actually do during early software design? How do they communicate? What sorts of drawings do they create? What kinds of strategies do they apply in exploring the vast space of possible designs?” Even as we make efforts to answer those questions, the majority of those efforts focus on how \_experts\_ design software. In other words, our novice-expert model for design expertise in computing is very asymmetric; most of our data come from fine-grained analyses on the expert side (Petre & van der Hoek, 2014). And, we cannot faithfully model how novices might develop into experts without studying novices.

This paper tries to address that asymmetry by investigating what directs and sustains students’ in-the-moment activity when they program. Its focus is on early-stage program design; using a “revelatory case study” (Yin, 2009) to show how what students say, do, write, and gesture even before they type a line of code can help us improve theories of cognition and activity in computing education. Through gesture, inscription, and discourse analyses of interviews with two students (“Rebecca” and “Lionel”) in a university-level introductory programming course, we argue that:

1. Students’ early-stage design activity reveals patterns outside the explanatory scope of misconception-based accounts of cognition in computing education.

2. We can recast how students make progress in design and why they get stuck in terms of “epistemological coherences” (Rosenberg, Hammer, & Phelan, 2006)—students’ in-the-moment views on what constitutes knowledge and knowing in program design.

3. These dynamic epistemological models can offer a lens for rethinking aspects of assessment and instruction in computing education.