## Technical Dimensions of Programming Systems: major revisions

We graciously thank the reviewers for their detailed feedback. We've made many minor changes to the paper and several major changes to both structure and content.

Two issues particularly stood out:

**Definition of "system".** All three reviewers requested clarification on what constitutes a "system", mentioning IDEs amongst other things as having unclear membership. Unfortunately, we truly don't have a definition behind the scenes which we're using to include or exclude examples. It's more of a "family resemblance" between things that generalize "programming languages" to "software which lets one program, yet doesn't have to be a language". Nevertheless, we thought it might be worth adding a new classification scheme for the examples we mention, which should get across the key "reference classes" from which we're drawing. In sum, these are: things that are like operating systems (O-type), things that are like domain-specific applications (A-type) and the systems that are based around a programming language (L-type). We also went into detail about how a programming language (Java) can be interpreted as a system in several different ways. These were all additions to Section 3.

**But how do you use it?** Two reviewers were concerned that it's hard to see how to use the framework. We've rectified this by including a new, concrete "Discussion" section with two halves: one using the framework for analysis, the other for synthesis. Specifically, we analyze Dark, a recent programming system, along many of the dimensions. Then, we plot two dimensions (self-sustainability vs. notational diversity) against each other to reveal an unexplored gap. The methodology we used to turn qualitative dimensions into numbers is given in a new Appendix. Various points earlier in the paper (e.g. the Abstract and Figure 1) have been updated to set up for this new section.

Other changes:

- We've significantly reworked Section 3 to be more to-the-point about the example systems and less of a historical narrative.
- We've argued for why we think "sociability" belongs as a technical dimension in the relevant section.
- We've reorganized and re-worded parts of Section 2. We've also made the part about how the framework relates to Cognitive Dimensions in Section 2, under "Already-known characteristics", more explicit.
- We've clarified some of the points about Direct Manipulation (primitive actions) and Immediate Feedback (pure code previews).
- Under "Addressing and Externalizability", we've made it explicit that CSS provides Additive Authoring for the Web programming system.
- To make room for the "Discussion" section, we've had to move more material to the existing Appendix with the bulk of the dimensions. Now, the main body only contains the Customizability dimensions.
- We've elevated the singleton Automation cluster to a doublet by adding a Factoring of Complexity dimension.