Reading 1 Response

Liam Strand

Wirth cites the introduction of the personal workstation in 1975 as critical to the development of computing. Give two examples of post-1975 computing innovations that you think also profoundly affected computing, and explain why.

One obvious example is the ubiquitous modern smartphone. Today, it is common for a person to be *in physical contact* with a powerful computer for their entire waking day. This would be an incredibly foreign concept to any computer scientist before 2000.

Another example is the Python programming language. With Python, it has become very easy to write "quick-and-dirty" programs to automate tasks. The combination of the Python Interpreter's REPL and Python's very high-level syntax allows programmers (and even non-programmers) to "play" very easily. The broad, deep ecosystem of Python packages, along with the incredible package management system lets programmers focus on solving *new* problems, rather than re-implementing a directed graph for the thousandth time!

Wirth is very focused on certain aspects of software engineering. Based on your experience, what are two software engineering challenges or solutions that Wirth entirely ignores?

While Wirth understands that language choice has a big impact in the educational environment, he only discusses how the languages nudge students to program, rather than what concepts the languages expose students to. For example, C and C++ can encourage students to develop terrible habits in structuring their programs, but they also allow students to learn about computer architecture, using the low-level memory access afforded by these languages.

Wirth also only briefly mentions the problems associated with managing concurrent and distributed systems and does not discuss any solutions, like those afforded by functional programming languages like Erlang.