FIRST, PERMIT ME to express my gratitude to Forrest Shull for the sacrifices he has made to be an excellent editor in chief for *IEEE Software*. The entire software community owes him a debt for the high standard to which he has created and holds the content of the magazine. Thank you!

This message is in response to the From the Editor column in the Jan./Feb. 2014 issue. I don't know objectively whether we are progressing, regressing, or standing still. Within the context of the special issue on software quality, I think the assessment is probably pretty dismal, as nearly all remnants of high-quality software have been removed from the marketplace and any processes that created high quality—in training, in education, and among developers— is also gone. How many courses are there on "preventing software errors"? Especially compared with the number of courses on a specific language, such as JavaScript, that is error-prone by design? Or, in the acquisition community, how many courses are they are on how to buy high-quality software?

From my perspective, a turn came in the mid-1990s. For example, Digital Equipment Corp. wrote some of the highest-quality operating and database systems. In 1998, it was acquired by Compaq, a commodity equipment provider that wrote no software. The year before that, Compaq bought Tandem Computers, a Silicon Valley enterprise famous for its fault-tolerant NonStop hardware and software line that were used, for example, by the New York Stock Exchange. Compaq disbanded both DEC and Tandem, and nowadays it's very difficult to identify any major software providers that optimize product quality. In fact, it appears that we have gone the other way, as the major software providers—for example, Microsoft, Google, Oracle, and Nintendo—have horrible product quality.

They're optimizing other things—primarily time to market. They each have incredible technical debt. Organizations like the Software Engineering Institute, NASA Goddard Space Flight Center, and ISO were independently trying to improve software product quality. Which of the enterprises are still in the business of quality improvement and which are not? To respond to one of [Editor] Shull's questions on a deeper level, I, along with many others, wonder whether we're undertaking assignments for which the complexity exceeds our abilities. Like everyone else, I'm influenced by the last thing I read, and at this time it's *X-Events: The Collapse of Everything* by John Casti. The volume was recommended to me by Barry Boehm because he knew of my interest in requisite variety, which can be seen as the matching of how we organize ourselves to achieve a result inside some environment, some part of the real world. So, one gloss on software product quality is that we've undertaken such complex operations because we view them as necessary to control or respond to the complexity we see in the world, and our ability to organize ourselves to match the world's complexity is less than what's required.

Some observe—and I owe this to James Thompson, director of major program support of the Office of the US Assistant Secretary of Defense for Systems Engineering—that governments do indeed acquire high quality software, so there can be a match. It just seems that we need to understand better how to rate problems, rate development organizations, and rate acquirers, and then see whether a sufficient match exists. I'm working on it, but the problem is hard, possibly wicked.

Keep up the good work!

Stan Rifkin
Master Systems