MAY ISSUE PERSPECTIVE

Computer’s May 2006 issue included several articles that require comment.

First, I take exception to “The Problem with Threads” (Edward A. Lee, pp. 33-42), which assumes that concurrent programming should be mainstream. Such a claim needs proof, not just assertions.

A better option would be an article on how to remove nondeterminism from systems where it is inherent, such as real-time communications, and make them predictable and bug free.

Second, concerning the fine summary of IEEE 1220 (Teresa Doran, Standards, “IEEE 1220: For Practical Systems Engineering,” pp. 92-94), which has come a long way, but is still short of being useful as claimed: The IEEE still needs to fill in the omissions and remove the ISO- and CMMI-type excesses.

Harmonizing with 15288 (or any other ISO standard) is a mistake. IEEE and ISO standards should all harmonize with what is right—not the least worst that can get approval by a committee. We should strive to do what is right, not compromise to avoid confrontation.

Finally, “Can We Make Operating Systems Reliable and Secure?” by Andrew S. Tanenbaum and coauthors (pp. 44-51) overlooks the real solution as hinted at in IEEE 1220. A total systems approach is needed. And note that system takes in a lot more ground than the software that has historically been the purview of most practitioners and their software-oriented journals.

The simple answer is, yes, we can build a secure system—if we view the OS from the larger perspective as a component of the actual system and also include the hardware in the initial systems architecture.

Note: A true systems architecture includes technical components for supporting areas including hardware, software, communications, information, security, operations, and so on, as well as defining the nonfunctional requirements including environment, performance, reliability, power, weight, and so on, along with context and genuine constraints.

Adding security after the fact is impossible. The hardware is one core requirement for security, as is the explicit architecting in of security from the beginning.

I solved this security problem for PCs in the 1980s. Amazingly, another group in our company independently came up with the exact same approach on a classified contract.

I am now working with a small consortium that will soon be offering a graphics interface OS and PC architecture to the Department of Homeland Security, NSA, and others, in case they actually would like the US to have a truly secure PC for use.

The only real risks are that politics would block the acceptance of our approach as it could blow Microsoft’s monopoly away or that the government would reject this concept because it actually wants insecure PCs so it can access them easily.

Finally, stepping back and looking at the larger perspective that seems to contain a common thread in this issue: The IEEE needs to bring systems considerations (systems thinking, systems science, systems architecture, and systems engineering) under its purview via an IEEE Systems Society to promote the proper approach to solving problems and instantiating workable cost-effective solutions without unintended consequences.

William Adams williamadams@ieee.org

Edward Lee responds:

I appreciate the thoughtful comments on my article. To address the issues William Adams raises, it’s helpful to cite some applications that require both concurrency and nondeterminism.

Consider any application that deals with stimulus from outside the system—Web servers; any embedded software system that accepts sensor inputs; user interfaces; databases; in fact, most of computing. Given such stimulus, we could avoid concurrency by batch processing, that is, react to each individual event and process it fully before moving to the next one.

But is this really a good idea?

If the application is an accounting system, each user would need to wait in line until no other user is accessing the system and would need to complete all entries before the next person can work. This eliminates the concurrency, but does it really eliminate the nondeterminism?

There is still contention for positions in line. Can we rigorously define the order in which people wait in line? What if two people arrive at the same time?

To completely eliminate nondeterminacy from this application, it seems that the “system” would have to include all its “users,” becoming completely closed and noninteractive.
Then we could specify the ordering deterministically. This would make computers much less useful, however.

I agree (and argue in my article) that nondeterminism should be avoided, with the caveat that for some applications, that isn’t possible. My case against threads is that they introduce enormous amounts of completely unnecessary nondeterminism.

Teresa Doran responds:
I would like to respond to Mr. Adams’s comments regarding IEEE 1220, standards harmonization, and the need for an IEEE Systems Society.

IEEE 1220 provides a strategy, process, and guidance that organizations and projects worldwide use in their practice of systems engineering. I have worked with several organizations that include IEEE 1220 in their suite of systems engineering standards and that have integrated IEEE 1220 with their organization’s processes. This would not be the case if IEEE 1220 was not considered useful.

It is not clear what Mr. Adams particularly finds excessive in the requirements of IEEE 1220, but the standard’s tailoring provisions allow organizations that claim conformance much leeway in creating tailored applications to suit their business needs.

Harmonization of the IEEE CS Software and Systems Engineering Standards Committee (S2ESC) collection with that of its ISO/IEC counterpart (SC7) benefits both communities. The article’s explanation of IEEE 1220’s alignment with ISO/IEC 15288 offers one example.

Defining what is “right” and acceptable for standard users on a global scale is no easy feat. Whether a standard is developed in S2ESC or SC7, a committee of subject matter experts typically performs the work. Different IEEE and ISO/IEC consensus-building and balloting approaches are coordinated. This harmonization initiative facilitates a standard’s multinational recognition and acceptance. Promotion of consistent and complementary collections minimizes contradictions among joint users.

Regarding Mr. Adams’s comment on the need for an IEEE Systems Society, an IEEE Systems Council was chartered in June 2005. Recognizing that many IEEE societies are home to system-related elements and efforts, the council is designed to promote synergism and cooperation across its member societies. The council’s broad field of interest associated with the engineering of systems should encompass the systems considerations Mr. Adams mentions. As of March 2006, the council included 15 participating IEEE societies.

Andrew S. Tanenbaum responds:
We are extremely pleased to learn that William Adams solved the PC security problem in the 1980s, and we look forward to learning what the solution is, since current PCs remain highly insecure.

We are certainly aware that if a single company controlled the hardware, software, and networking, it might be possible to employ a systems approach and perhaps find an optimal total solution. Perhaps a merger of Intel, Microsoft, and Cisco would make such a solution possible.

On the other hand, before its breakup in 1984, AT&T controlled the hardware, software, and networking of the telephone system, and most impartial observers would say that the phone system is much better now than when one company controlled everything and was able to globally optimize it.

In short, we do not believe that this proposal is practical in a competitive market with many players, each optimizing its piece of the puzzle. Thus, like it or not, companies making operating systems have to live with hardware produced by someone else, and they are trying to do their best under those circumstances.

We welcome your letters. Send them to computer@computer.org.