Software Engineering Revolution

I fully endorse Bertrand Meyer’s observation that, if not done right, requirements engineering, documentation, configuration management, quality assurance, contracts, and management practices become project killers (“The Unspoken Revolution in Software Engineering,” The Profession, Jan. 2006, pp. 124, 121-123). Technologies such as J2EE, DB2, and C# are important, but using them to full advantage requires the sound foundations that Meyer refers to.

According to the “NASSCOM Hewitt Total Rewards Study” (http://nasscom.org/artdisplay.asp?catid=737), there are seven levels in the Indian IT services sector:

• software engineer,
• senior software engineer,
• team leader/module leader,
• project leader,
• project manager,
• program manager/senior project manager, and
• head—software development/large business unit.

As this list indicates, only two levels require creative programming skills— the other five levels focus on management.

Most Indian engineers working in the IT field are software engineers (mainly programmers) who work on a project for one or two years. After the project closes, they are promoted to become senior software engineers, and from there, they move on to other levels.

In India, the undergraduate computer science syllabus includes more than 60 subjects. Among these, 40 are related to technology, and five others include software engineering, software quality management, engineering economics and management, marketing management, and total quality management. There is no coverage of topics such as project management, documentation, or contract engineering.

I advise my students that most software engineers spend less than five years working as programmers, while the rest of their professional life will focus on project management, quality management, contracts, and general management.

Most of a software engineer’s education is oriented toward research, not development. For every computer journal, we have more than 20 transactions. Naturally, there is a large gap between theory and practice. Yet, we are all surprised that 70 percent of our projects fail.

R.T. Sakhidaran
sakhidaran@computer.org

The Applied Mathematics and Computer Science Schism


It is common knowledge, at least where I reside, that the demand for university placements in engineering, science, and mathematics steadily dwindled throughout the past decade. This suggests that high school graduates are, as a whole, losing interest in these fields.

Let’s face it: Mathematics is a subject that many people find inherently tedious and difficult. The decline in demand for these courses, as well as possible concerns over high failure rates and not wanting to make them even less attractive, have led to a general lowering of the admissions standards. The students who enroll in computer science-related courses are hence increasingly less well-equipped to deal with the mathematical rigors that ideally would characterize such courses.

During my university studies—I graduated in 2003 in an engineering discipline—it became evident to me that many of my peers did not see mathematics as being relevant to the sort of IT or engineering work they envisaged themselves doing. I believe this is a shame because such an attitude considerably restricts career options.

Quite simply, it isn’t possible to foster aptitude if the student has no interest or motivation. Even with increased exposure to mathematics, those who aren’t really interested in the discipline will learn the content at a superficial level, rote-memorize some appropriate formulas, and hope for the best. These students will be content with “just passing” the subject and never having to do it again.

A fundamental shift in the attitudes of young people toward mathematics and science is required for the benefits of interdisciplinary training to manifest themselves on a grander scale—and this needs to start long before students enter university lecture halls.

Quang Thai
Sydney, Australia
qlthai@gmail.com

Professional Precision

I thoroughly enjoyed reading “In Praise of Professional Precision” by Neville Holmes in Computer’s April 2006 issue (The Profession, pp. 100, 101-102). However, I take issue with one point: 1k2, 2k3, 2m2, and so on already have well-established meanings in engineering. If applied to electronic resistors, for example, they mean 1.2 kilohms, 2.3 kilohms, and 2.2 megohms.

This notation is seen primarily in Europe, and I first noticed it while working on some European medical equipment. It is my understanding that inserting the metric prefix in the position of our decimal point was to avoid the challenge of little dots spontaneously appearing and disappearing after multiple rounds of photocopying or faxing.

The existing metric prefixes are just fine the way they are, thank you.

Principle: Respect tradition.

Dan Landiss
St. Louis, Mo.
dan@landiss.com
The revelation of the “well-established meanings in engineering” of \(1k2\), \(2k3\), and so on comes as a complete surprise to me. Certainly they were not in use back in the early 1950s when I was studying electrical engineering, so, unless the usage in the US was quite other than that in Australia, they can hardly be traditional.

I suspect that the notation arose because the widespread adoption of the deplorably deficient 7-bit telegraphic code, ASCII, made it extremely difficult or even impossible to use, for example, the traditional rendering of \(\mu\) and \(\Omega\) for microhm, just as it all but killed off the traditional and useful spellings of words like café and façade.

The notation you describe seems most unfortunate in conflating the operation of scaling with the marking of the end of the integral portion of the value. This is only less damaging than the conflation by mathematicians of the operation of subtraction with the property of negativity in the hyphen because the mathematical malconvention is so much more widely used. For your students, as for students of elementary arithmetic, the resulting confusion may well be leading to mistakes and misunderstanding, or at least difficulties in understanding, which their teachers don’t easily recognize.

As to the metric prefixes, I had no expectation that they would be abandoned. My suggestion of a scaling notation with base 1,000 was to supplement the so-called engineering notation, where the scaling base in 12e3 or 12E3 is 10, to start with in program code. This would make using the metric prefixes more amenable as the units of measures with their prefixes typically can’t be added to such code. I also had it in the back of my mind that, whereas in values like 2m3 and 4K5 the scaling base would be 1,000, in values like 2M3 and 4K5, the scaling base could be 1,024, which would be useful in computing applications.

I appreciate having the opportunity to clarify and elaborate this point.

I read the April 2006 The Profession column with great interest, as always. However, I would like to point out that one of the redundant phrases Neville Holmes cites is, in fact, not quite redundant.

I submit that “ballistic missile” is not redundant, owing to the existence of missiles that are not ballistic in nature. I cite three examples:

- the Beech Aircraft AQM-37A “Jayhawk” target drone,
- the Northrop Aviation N69 “Snark” cruise missile, and
- the Boeing IM-99A “Bomarc” remotely piloted interceptor.

Each of these definitely qualifies as a missile in the traditional sense, but all of them had wings and exhibited behavior that was decidedly not ballistic.

Notwithstanding that, I agree with the premise that principles of good writing are too often violated in technical writing and reporting.

Mark S. Johnson
Wichita, Kan.
mark.s.johnson@worldnet.att.net

Neville Holmes replies:

Traditionally, and in ordinary dictionaries, a missile is something that is thrown, either by hand or machine, and something that is thrown is ballistic. Relatedly, both the ballista and the arbalest were machines for throwing things. Because I believe we should use ordinary words ordinarily, even in technical writing, I maintain that “ballistic missile” should be regarded as tautological.

If this is accepted, then the term “cruise missile” must be seen as an oxymoron, unless the missile in question starts its journey from an arbalest.

In any case, names such as doodlebug and buzz bomb are much more specific and picturesque, so why have we dropped them?

A “Vocabulary of Information Processing” might be a helpful resource for professional formality if it was generally available, but having it hidden in some catalog (for a price, I assume; to be honest, I couldn’t actually find it) makes it useless.

As to using G for billion, maybe it’s safer to say 10° instead of having the reader try to figure out if it means American or British billion.

Michael Hunter
mpb@acm.org

Neville Holmes replies:

It’s not surprising that you couldn’t find the vocabulary, as its location seems to have shifted since we checked it just before publication. The URL is now www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=7229, where the reference is to ISO/IEC 2382-1:1993. This differs only slightly from what we published, but that’s all it takes to kill a URL.

I quite agree that the vocabulary should be freely available, at least to computing professionals. It now costs 120 Swiss francs, more than US$80, incidentally.

I concluded “The Great Term Robbery” (The Profession, May 2001, pp. 96, 94-95) urging the Computer Society to negotiate with the ISO to have a copy online free for members and for the public. Even if it were only free to members, that would be an important contribution to professionalism.

As for the confusion of billions, the British one has long been swamped in Australia, and it would also seem to be in Britain (guardian.co.uk/notesandqueries/query/0,5753,61424,00.html). In fact, I would be surprised if there are any English-speaking countries where it has survived.

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