Brady Field

Software Engineering

**Important points**

“Studies have shown that for every six new large-scale systems that are put into operation, two others are cancelled. The average software development project overshoots its schedule by half; larger projects generally do worse. And some three quarters of all large systems are ‘operating failures’ that either do not function as intended or are not used at all.” [63]

“In fact, a research innovation typically requires 18 years to wend its way into the repertoire of standard programming techniques. By combining their efforts, academia, industry and government may be able to hoist software development to the level of an industrial-age engineering discipline within the decade.” [64]

“Like handmade muskets, several programs may perform similar functions and yet still be unique in design. That makes software difficult to modify and repair. It also means that attempts to graunch systems together often end badly.” [65]

“Even the best-laid designs can go awry, and errors will creep in so long as humans create programs. Bugs squashed early rarely threaten a project’s deadline and budget, however. Devastating mistakes are nearly always those in the initial design that slip undetected into the final product.” [66]

“Although the demand for more sophisticated and reliable software has boosted some large-scale programming into the commercial stage, computer science (which is younger that many of its researchers) has yet to build the experimental foundation on which software engineering must rest.” [67]

“Secondly, formal methods can guarantee only that software meets its specification, not that it can handle the surprises of the real world.” [68]

“Then again, the industry has heard tell many times before of ‘silver bullets’ supposedly able to slay werewolf projects. Since the 1960s developers have peddled dozens of technological innovations intended to boost productivity – many have even presented demonstration projects to ‘prove’ the verity of their boasts.” [68]

“The fact of the matter is that no one really knows how productive software developers are.” [68]

“After 25 years of disappointment, with apparent innovations that turned out to be irreproducible or unscalable, many researchers concede that computer science needs an experimental branch to separate the general results from the accidental.” [69]

“But as international networks sprout, and large corporations deflate, India, Hungary, Russia, the Philippines, and other poorer nations are discovering in software a lucrative industry that requires the one resource in which they are rich: an underemployed, well-educated labor force.” [70]

“The combination of industrial process control, advanced technological tools and interchangeable parts promises to transform not only how programming is done, but also who does it.” [71]

**Disagreements**

“A quarter of a century later software engineering remains a term of aspiration. The vast majority of computer code is still handcrafted from raw programming languages by artisans using techniques they neither measure nor are able to repeat consistently.” [63]

I will concede that the processes of software engineering, and especially the measurements, are imperfect. However, software engineering usually follows a development process that does solve a great many problems inherent in the “code and fix” way of doing things. A good process also does enable producing quality software consistently. Apart from these points, the author here uses computer code and software engineering as if they are the same thing; this disregards the reality that software engineering is more than just coding.

**Questions**

I have no questions. I understood everything in the article.