IT Failure Rates— 70% or 10–15%?

Robert L. Glass

… in which I question the unquestionable Standish Chaos findings

This is a closed-book quiz. Quick, now— what percentage of software projects fail?

Let me hazard a guess as to your answer. Odds are, you said something like "70 percent." Am I right?

Now, I want to analyze why you said whatever you said. A friend of mine, Nicholas Zvegintzov (the Chief Guru of software maintenance), likes to say that, if you track down the sources of the things we "know" in our field, you'll find that what we thought was a plethora of sources for that knowledge generally boils down to only a precious few.
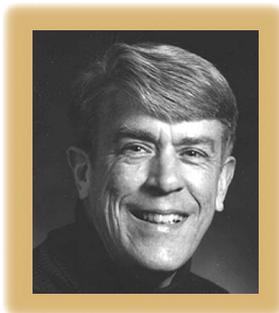
That raises the question "Where do we get our information on software failure rates?" I've seen software failure trumpeted from so many academic research papers that I had to quit counting (they tend to see a "software crisis" and then say that the research work they're describing will help eliminate it). However, if you closely examine the citations they use to support the claims of crisis, over and over again the citations boil down to one primary source, the Standish Chaos Reports. And, if you take matters one step further, the papers generally cite the 1994 report.

I want to question the unquestionable status of that Standish report. That's because, you see, my own observations lead me to believe that something is terribly wrong with those Standish findings.

Success depends on your viewpoint

First, there's the matter of the 1994 report. If you go back a decade and look at what it actually said, it was this—31 percent of software projects were cancelled, 53 percent were "challenged" (had difficulty meeting goals), and 16 percent were successful. Most people tend to add that 31 and 53, concluding that an astonishing 84 percent of software projects are deeply troubled. It's interesting to question whether those two numbers should be added, but at the same time, the finding that only 16 percent of software projects were successful is very damning to the field.

Those academic researchers who quote those original 1994 figures have ignored the fact that Standish has updated the Chaos Report periodically. The 2000 report, for example, saw a drop in cancelled projects to 23 percent, a small drop in challenged projects to 49 percent, and a corresponding increase in successful projects to 28 percent. Still not figures to be proud of, of course. But an improvement, nevertheless.

Then why do academics (and, to be honest, popular-press periodicals) continue to quote those 1994 figures?

The only reason I can come up with is lazy research. It's easier to copy the figure that they've seen so often in previous publications than to seek out the newer figures. (In addition, Standish charges something like $5,000 for the reports, a figure that many researchers are undoubtedly unwilling to pay.)

Pardon me for looking the gift horse of Standish improvements in the mouth, but I think there's something bogus about all those Standish findings. My own view of the software field, bolstered by the fact that most of my experiences as both a software developer and a software user have been largely successful, is that software projects succeed far more often than they fail. As I've said many times before, I see this as the Computing Age, an era that simply wouldn't be possible if we didn't have astoundingly successful software to make all those computers do the wonderful things they do.

So what could be wrong with the Standish findings? I've gone to their Web site and reviewed the questions they ask, which seem perfectly objective. I don't know whom they send their questionnaires to, but it's hard to imagine that they've accidentally and consistently sent them only to software losers, organizations that can't program their way out of a paper bag. But there are at least these possibilities:

- "Failure" and "success" are tricky words in the software business. How do you categorize a project that is functionally brilliant but misses its cost or schedule targets by 10 percent? Literalists would call it a failure, realists a success.
- People tend to focus on organizations that fail. For example, an early US Government Accounting Office study of projects that failed, which came up with an astounding failure rate, turned out to have examined projects already known to be in trouble when the study began.

When I put together the special issue of *IEEE Software* on the State of Software Engineering Practice back in November 2003, I invited Standish to include a report on their findings as part of that issue. It bothered me that they declined—and declined and declined, as I repeated my invitation several times—and eventually I had to give up on presenting their viewpoint. A viewpoint, as you can imagine, that I thought would be pretty darn relevant to a discussion of the state of software's practice.

Other dissenting voices

Time has passed since then, and I've continued to mull over this dilemma. I even put an appeal in one issue of *Software* asking for input on how Standish does its work. I received only one response.

Magne Jørgensen of the Simula Research Lab in Norway and his coauthor Kjetil Johan Moløkken-Østvold shared my doubts regarding the Standish studies' accuracy. [An unrelated article by Jørgensen, "Practical Guidelines for Expert-Judgment-Based Software Effort Estimation," appears in this issue.—Ed.] So, Magne tells me, they studied one particular aspect of the Standish findings: that (at least in the early Standish reports) software projects suffered from an average 189 percent cost overrun.

Looking at other research studies of software failures, they discovered something interesting. Whereas Standish reported those 189 percent overruns, three other studies reported a consistent 33 to 34 percent cost overrun. Clearly, something was at best inconsistent between what Standish was doing and what the other three studies had learned. (To check out the Jørgensen findings, go to www.simula.no/publication_one. php?publication_id=711. Jørgensen is trying to get those findings published as we speak, and there might well be a much more comprehensive source by now for continuing this discussion.)

Others are beating the drum that something might be fishy in the Standish data world. For example, in a special issue on the "Great Myths of IT," *InfoWorld* (16 Aug. 2004) included as its Myth 5 "Most IT Projects Fail." They cited, then openly doubted, the Standish findings in their discussion of that particular myth.

So what should you believe?

Meanwhile, whatever the truth of the matter, most people seem to have seen and tend to believe the Standish findings. A particularly interesting and indicative case of this came from a discussion in the 5 Nov. 2003 *Cutter IT E-Mail Advisor*. That article reports that a speaker asked an IT audience how many of them agreed with the data that showed an "over 70% failure rate of IT projects." Everyone agreed. Then, the speaker asked them what the failure rate was in their own companies. "85% of the audience reported 5–10% failure rates for their companies." The article concluded with the speaker's tongue-in-cheek comment, "I was lucky to have an audience from such high-performing companies," implying that the audience wasn't entirely truthful about its own failure rates.

Perhaps. But isn't it also possible that the 70 percent failure rate that everyone accepted, probably because they had seen it published so often in analyses that included the Standish findings, is really the figure that should be doubted?

We haven't seen the last of this issue. When and if the Jørgensen report is published, the issue will arise again bigtime, as well it should. Whatever the true failure rate of IT projects, our field should spare no effort in seeking the

> **My own view of the software field is that software projects succeed far more often than they fail.**

truth of the matter. In many ways, the future of the software field is at stake.

I repeat my invitation to Standish. Dear Standish folks—if you would like to rebut, explain, or supplement the views I've presented, I'd be happy to publish your response in a future Loyal Opposition column. 🆂

Robert L. Glass is editor emeritus of Elsevier's *Journal of Systems and Software*, the publisher and editor of the *Software Practitioner* newsletter, and someone whose "head is in the theory of software engineering, but whose heart is in its practice." Contact him at rglass@indiana.edu; he'll be pleased to hear from you.

IEEE Software

IEEE Computer Society
10662 Los Vaqueros Circle
Los Alamitos, California 90720-1314
USA
Phone: +1 714 821 8380
Fax: +1 714 821 4010
http://www.computer.org
advertising@computer.org