Position Statement

“Is Software Reliability an Oxymoron?"

Panel Session at the Technology Business Council Software Roundtable,
Richardson Chamber of Commerce,
October 26, 2000

Danny Faught, Cigital, Inc.
faught@cigital.com
http://www.cigital.com/~faught

Software reliability is both difficult to define, and difficult to achieve. But a couple of different angles on the subject give us hope.

What is reliability?

According to the IEEE, “reliability” is “The ability of a system or component to perform its required functions under stated conditions for a specified period of time.” This implies that failures tend to be catastrophic, such that it’s easy to tell whether something is working or not. With software, catastrophic failures are certainly common. But just as common are failures that fall into a gray area, where the application is still mostly working, but some small feature is malfunctioning, or perhaps a user interface flaw prevents the user from utilizing the application properly. Also, catastrophic failures often start with small localized failures that compound eventually into a big train wreck of problems. It’s hard to define when the application actually stopped working.

Factors

Two important factors come into play in a software user’s perception of reliability. First is the concept of “Good Enough” quality. The focus is not to create defect-free software, but to create something that the customer will perceive as a good value. A software development organization will balance the reliability that is expected for their product with the cost that the market will bear, and the time frame in which it is needed. In a few cases, ultra-high reliability is necessary, and this leads to ultra-high prices. Usually lower prices are required in order get market penetration, and the customer can tolerate lower reliability. So you may hear software users complain about less-than-stellar reliability, but they may not be willing to pay for anything better, or they may not be willing to wait for it any longer.

The second factor is the prevalence of risk. “Risk is everywhere,” says Cigital CEO Jeff Payne. Software projects frequently get surprised by problems. Often the problems are big hairy problems that drive right into the company’s bottom line. Risk isn’t necessarily bad; it’s a natural occurrence. Companies take big risks with the hope of reaping big rewards. But risk should be managed. We need to understand what all our risks are, how likely it is that they will become problems, and how much damage they will cause to the business if they do.

We can reduce the likelihood of getting bitten by a risk, and we can reduce the potential damage. We can also put contingency plans in place, so that if a problem does bite, we’re ready to start the damage control immediately. Too many companies don’t make the effort to identify and mitigate their risks. They end up using their resources to fight fires, and overall reliability suffers. The development organization is surprised by the problems, which leads to the customer being surprised by poor software reliability, if the software ever even sees the light of day.

Where do we go now?

The industry is putting tremendous amounts of energy into improving the software development process, so that instead of creating software that is just barely Good Enough, we can get to a point where it’s cheaper not to put the bugs in in the first place. I believe that the craft of software development is still in its infancy as an engineering profession, and we have many decades of growing up ahead of us.

Risk management gives us the tools to deal with the realities of the software development process, wherever it is in its evolution. You’ll find many resources describing risk management at the project level. We need to incorporate the process at all levels of the company, from the corporate risk manager and other executives down to the project manager and engineers. This will reduce the surprises that so often cause a compromise in the level of software reliability.

Is “software reliability” an oxymoron? Not really, when we understand the market realities, and if companies learn how to deliver on their promises with no surprises.