What’s Your Fault Feedback Ratio?

Posted by johanna on Nov 1, 2002 in Articles | 0 comments

Most of us track faults (also called defects, problems, issues, bugs) during the system test part of the project. However, many project managers don’t track how many of our fixes are successful and how many fixes are bad — either introducing a new defect or not completely fixing the original defect. If you’re looking for better scheduling of your system test and project completion, start measuring your Fault Feedback Ratio, the FFR.

Here’s how to calculate your fault feedback ratio:

\[
\text{Fault Feedback Ratio} = \frac{\text{Fixes that require more work}}{\text{All the fixes}}
\]

When I’ve measured successful projects, the FFR stays under 10% throughout the project, meaning that no more than one in 10 fixes are problematic as the work progresses. (Don’t be deceived by a low FFR and a high overall fault count. With a large total count, the defects can be too overwhelming to successfully manage.) A low FFR and a not-too-high overall defect count also implies that the developers have a relatively easy time finding and fixing the problems. The testers don’t have too much trouble keeping up with testing the fixes, because the fixes haven’t broken other pieces of the software. The project team is progressing.

On the other hand, an FFR of 15% or higher means that people are spending significant time finding and fixing problems. A higher FFR and a low defect count may mean that the ct counts, may mean the developers may have trouble detecting where to fix the problems, and the testers may have trouble verifying the fixes are good.

Once the FFR hits 20%, you’re making dubious progress on the project because your team is spending too much time re-fixing problems they thought were already fixed and retesting those fixes.

On one project, the FFR started at 18% in the implementation phase, when the project team started to track their defects. Because the developers had completed the design before they started coding, they had trouble fixing the defects quickly. To make up the time, they took shortcuts for fixing the defects, and stopped doing peer reviews on the new code.

By the time the project was supposed to start system test, the FFR was up to 23%. The project had met its previous milestones, but the team was unable to predict the start of system test. Their previous progress was an illusion because uncorrected defects remained in the code. With an FFR of 23%, developers had to take the time to understand each problem and how it affected the rest of the code base, to reduce the fix time for each problem. The project looked stalled, even though the developers were now successfully fixing the problems.

Bug Bash

The technical lead suggested a “bug bash,” where everyone tries to find and fix problems, but the project manager suggested an alternative: a two-week period where every fix required peer review. At the end of the two-week period, the FFR was down to 6%, a dramatic improvement. The team decided to peer review all fixes from them on, an effective technique for keeping the FFR low. You may not catch all of the problem-fixes using this approach, but you’ll catch many of them.

Measure your FFR to see if you’re spending too much project effort on fixing fixes. FFR is not linear with effort, so an FFR of 10% does not mean that you spend 10% of your time fixing fixes. You may well be spending more than 20% of your developer time on those 10% of the fixes. In one organization I consulted to, the FFR was 22%, but the developers spent almost 80% of their time fixing fixes.

Track FFR as soon as you start tracking defects, as early in the project as possible. If you start measuring the FFR only at the end of the project, you’ve missed an opportunity to see how your defects are affecting your project’s progress. Calculate the FFR for the entire project each week, not by developer or by code area, making sure the data can’t be associated back to a specific person. Your project staff is allowed to make mistakes when they fix problems; your job is to see whether the bad fixes are causing other problems in the project. Use the FFR data at your project team meetings as an early warning sign, to see if your project progress is an illusion or real.

© 2002 Johanna Rothman. This article was published on Computerworld.com, November 2002.