Microspeak: Bug jail

October 11, 2011

Raymond Chen

Bug jail is not a place where bugs are sent as punishment for their crimes. Rather, it's a (virtual) place that developers are sent when they have too many bugs. Project management establishes some maximum number of bugs (known as a *bug cap*) each developer is permitted to have on his or her plate, and developers whose bug count exceeds the specified maximum are placed in *bug jail*. The precise triggers for bug jail vary from team to team, and it may vary based on the nature of the bug. For example, one trigger might be that any Priority 0 bugs more than 24 hours old will land you in bug jail. Once you land in bug jail, you are not allowed to do feature work, be it coding, writing specifications, whatever. The precise triggers for getting out of bug jail also vary from team to team, but one rule might be that you need to get your bug count back down to 50% of the bug jail trigger level before you are allowed to exit. Staying on top of your bug count is an important part of the software development process, and the primary motivation behind bug jail is not to punish developers, although I'm sure that's how most developers perceive it. Rather, it serves as an early-warning system to highlight things that are not going smoothly. There may be a bug farm developing in that feature area. Or the team needs to revise its idea of what it means to be "done" with the feature. And it's a signal to project management that they may need to scale back their plans so as not to compromise quality and the ship schedule.

When everybody understands the reasoning behind bug jail, you avoid lengthy discussions over boundary cases ("What if a developer is about to check in a feature but a low-priority bug comes in that pushes them over the limit?") and avoid discovering that people have been "gaming the system" (for example, by reclassifying bugs as feature requests so they don't count toward the bug cap). Like bug hugging, these sorts of games prevent project management from truly understanding how close the project is to being finished.

Raymond Chen

Follow