Table of Contents

Why Shouldn't We Set Up Dedicated Defect Teams?

Or “What are the benefits of having value production teams work on defects?”

Came across "Bug teams - Well Meaning Foolishness by Tim Ottinger" recently. Discussion is well-thought description of the downside of setting up a bug team but I thought it had a wider message than that - the idea that we need to treat production of bugs as something that you can analyze, run experiments to reduce, and increasingly create a team “immunity system” aimed at preventing bugs.

Starting with the idea that development has the following characteristics:

- Programming is not typing; it is a purely intellectual labor
- Code is a collection of decisions, not a collection of keystrokes.
- Good code is a set of decisions made well.
- A defect is a result of one or more decision(s) made poorly.
- Maintainability is the ability to arrive at good decisions quickly
- Poor code invites or limits decision-makers to compromised decision-making

It then takes you through the idea that if these are true, then have a team become expert on bug fixing actually won't help the problem, because the non-bug team is still producing all the bugs. We say this with the statement “we want the team to feel the pain of the bugs they produce”.

But that doesn't stop the discussion as we continue to talk about setting up bug teams. We have a problem in that a lot of capacity is being taken up by fixing defects, and it often overruns the capacity we though we were going to apply to the problem. The idea of a bug team would allow us to set a limit on the capacity we apply to the problem, since we can say “here is the team to do it, and you can only get whatever that team is able to produce.” In other words we are dealing with a human problem - we take on too much bug work and so we can deal with this artificially by setting capacity to the team.

The big thing that the article points out, and the point of this discussion, is that to deal with the root cause, we need to say more than “teams deal with the pain”. We have to encourage and help people on teams understand that they might want to do work so there is less pain in the future. The article talks teams working to ensure that bugs are not only addressed but that we increasingly create systems that reduce the chance of a defect getting into the work we do. The article calls this a team's “immune system”. This means both investing in automation (to ensure we get fast feedback), but also investing in time to really have the team understand why a bug was created and to take action which addresses this. It suggests working the bugs we have more scientifically, rather than just as they come in, do analysis on what bugs we have, what areas of code it came from, time work was done, who was involved, and then conducting experiments aimed at reducing the impact of problem areas that are discovered.

There may still be reasons we create a bug team to help deal with a capacity issue. But it is also important to understand that setting up a bug team won't necessarily improve the quality of the code produced. Irrespective of whether there is a team in place to do this, we still need to deal with the root cause issue, and really focus on understanding why we have defects. We need to set up our development environment so that we get feedback very quickly about the bugs, and then encourage real analysis back

to the coding that was done to understand how we can stop bugs from entering the system the same way. This is the only approach that has the potential to reduce the capacity that we allocate to working bugs freeing everyone up for doing more interesting, and more valuable work.

FAQ, Quality, Bugs, Defects, Team

~~LINKBACK~~ ~~DISCUSSION~~