PATTERN 06

Over Helping

6. Over Helping

Collaboration among teammates is a natural and expected part of the development process. Over Helping is the pattern whereby one developer spends unnatural amounts of time helping another developer to get their work across the line.

Engineer One submits. Engineer Two cleans it up, over and over again. This behavior can be normal on small project-based teams. But when that 1-2-1-2 pattern doesn't taper off, it's a signal that should draw your attention.

The problem is threefold: (1) always cleaning someone else's work takes away from one's own assignments, (2) it impairs the original author's efforts toward true independent mastery, (3) it can overburden the helper and leave the original author in a continuous unnatural waiting state.

How to recognize it

You'll notice this pattern in the same way you'd realize "Heroing" (Pattern #5) in Pluralsight Flow's Review and Collaboration reports and the Help Others metric. Look for reoccurring, last-minute corrections between the same two people.

In the Review and Collaboration and Operational reports, you'll notice these two consistently review each other's work. One engineer will have a high Help Others, but it's not reciprocated. The "load-bearing" engineer will also show high levels of Influence and Review Coverage. The other engineer will not. One engineer will have a high Impact; the other won't.

This behavior can be perfectly healthy and expected when in a mentorship-type situation. But beyond a certain point, rotation is in order.

What to do

Bring additional engineers into the code review process. A side effect of this solution is that by increasing the distribution of reviews, you're strengthening the team's overall knowledge of the codebase (see Knowledge Sharing).

Cross-train and assign both engineers to different areas of the codebase.

Assign the senior engineer a very challenging project. The idea here is to give them challenging projects where they don't have the time or energy to review their colleague's work.

Lastly, the stronger of the two is showing natural leadership and coaching tendencies. Look for opportunities to feed this more broadly to the whole team.

One note of caution: be mindful when the two engineers are friends or were colleagues at a former employer. Making light of a friendship or teasing them can be incredibly damaging and hurtful. Go the extra mile to keep it professional.

And, as always, be transparent. You're not trying to split up friendships. It's the manager's job to ensure that knowledge of the codebase is distributed evenly across the team and to ensure that people are honing their craft and growing their careers.