PATTERN 01

Domain Champion

1. Domain champion

The Domain Champion is an expert in a particular area of the codebase. They know nearly everything there is to know about their domain: every class, every method, every algorithm and pattern.

In truth, they probably wrote most of it, and in some cases rewrote the same sections of code multiple times.

The Domain Champion isn't just "the engineer who knows credit card processing"; it's all they ever work on. It's their whole day, every day.

Some degree of job specialization is essential and often motivating. But even within specialized roles there can be 'too much of one thing.' Managers must balance enabling a team member to unilaterally own the expertise, and encouraging breadth of experience.

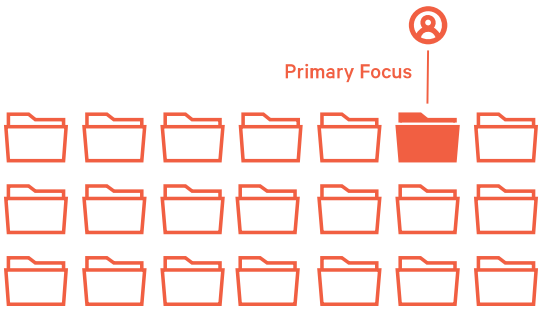
How to recognize it

Domain Champions will always work in the same area of code. They'll also rewrite their code over and over, and you'll see it in churn and legacy refactoring metrics as they perfect it.

Domain Champions are deeply familiar with one particular domain. As a result, they'll typically submit their work in small, frequent commits and will show a sustained above average *Impact*.

Because no one else knows more than the Domain Champion, there's usually very little actionable feedback that others can provide in the review process. As a result, Domain Champions will typically show low *Receptiveness* in incorporating feedback from reviews.

Domain Champions will seldom, if ever, appear blocked. Short-term, it's a highly productive pattern. But it's often not sustainable and can lead to stagnation, which of course can lead to attrition.



What to do

Assign tickets that focus on other areas of the codebase.

Of course, some engineers would prefer to stay where they are. It can be very enjoyable to do a task you're good at. And, it can be uncomfortable to take on work that requires information or skills you have less practice with. But effective managers will strive to challenge their team.

Start a new conversation in your next one-on-one:

1. Acknowledge their expertise and encourage them to share that expertise with others. Ask them who, if anyone, would benefit from participating in code reviews in the domain to learn best practices.
2. Ask them what they like to work on — first generally, then specifically.
3. Ask them if they are willing to take on a small assignment outside their domain in part to help share the best practices they've developed refining the code in their domain.

Inch the engineer out of their domain using small, low-risk tickets. A little bit of diversification can go a long way toward minimizing attrition risk and maximizing best practices.