Self-Organizing Teams Are Not Put Together Randomly

by Mike Cohn •

22 Comments

Image not readable or empty
/media/images/2005-04-Self-Organizing-Teams-Are-Not-Put-Together-Randomly.png
Self-Organizing Teams Are Not Put Together Randomly

The ability for a team to self-organize around the goals it has been given is fundamental to all agile methodologies, including Scrum. In fact, the Agile Manifesto includes self-organizing teams as a key principle, saying that “the best architectures, requirements, and designs emerge from self-organizing teams.”

As part of deciding how best to achieve the goal given them, some teams will decide that all key technical decisions will be made by one person on the team.

Other teams will decide to split the responsibility for technical decisions along technical boundaries: Our database expert makes database decisions, and our most experienced Python programmer makes Python decisions.

Still other teams may decide that whoever is working on the feature makes the decision but has the responsibility of sharing the results of the decision with the team.

Not Every Agile Team Will Self-Organize in the Same Way

There are two key points here:

- First, not every agile team will choose to organize themselves the same way, and that’s OK.
- Second, making use of the collective wisdom of the team will generally lead to a better way of organizing around the work than will relying solely on the wisdom of one personnel manager.

The benefit of allowing a team to self-organize isn’t that the team finds some optimal organization for its work that a manager may have missed. Rather, it is that by allowing the team

to self-organize, it is encouraged to fully own the problem of performing its work.

Can We Really Expect Agile Teams to Self Organize?

A common criticism of self-organizing teams is, “We cannot just put eight random individuals together, tell them to self-organize, and expect anything good to result.”

Well, I don’t know if we can put eight random people together and expect anything either, but an agile team should not be a random collection of people. In fact, those in the organization responsible for initiating a Scrum project should expend a lot of effort in selecting the individuals who will comprise the team.

Management Exerts Subtle Control over Self Organization

In the original paper describing Scrum, Takeuchi and Nonaka identified “subtle control” as one of its six principles. They list staffing decisions as a key management responsibility.

> Selecting the right people for the project team while monitoring shifts in group dynamics and adding or dropping members when necessary [is a key management responsibility]. “We would add an older and more conservative member to the team should the balance shift too much toward radicalism,” said a Honda executive. “We carefully pick the project members after long deliberation. We analyze the different personalities to see if they would get along.

Getting the Right People on the Agile Team

If you are a personnel manager or otherwise influence team composition in your organization, here are some of the factors to consider.

Include all Needed Disciplines on Agile Teams

As a cross-functional team, it is important that all skills necessary to go from idea to implemented feature be represented on the team. Initially this may mean that team size is slightly larger than desired.

But, over time, individuals on a Scrum team will learn some of the skills possessed by their

coworkers. This is a natural result of being on a Scrum team. As some team members develop broader skills, other individuals can be moved onto other teams.

Mix of Technical Skill Levels on Agile Teams

Subject to considerations of team size, you should strive to balance skill levels on the team. If a team has three senior programmers and no less-experienced programmers, the senior programmers will need to code some low-criticality features that they could find boring.

Not only might a junior programmer have found such features enjoyable to work on, that programmer would also benefit from learning through association with the senior programmers.

Balance Domain Knowledge on Agile Teams

Just as we strive to balance technical skills, we should strive for a balance between those with deep knowledge of the domain in which we are working or the problem we are attempting to solve.

This is not to say that if we have the opportunity to assemble a team entirely of domain experts we shouldn't take it. Rather, we should consider the long-term goals of our organization.

One of those goals is likely the build up of domain knowledge throughout the organization. You'll have a hard time achieving that if you put all of the domain experts on one team.

Seek Diversity on Agile Teams

Diversity can mean many different things—gender, race, and culture being just three among them. Perhaps equally important can be how individuals think about problems, how they make decisions, how much information they need before making a decision, and so on.

Research shows that homogeneous teams reach consensus more quickly than do heterogeneous teams, but they do so by failing to consider all options.

Consider Persistence When Forming Agile Teams

It takes time for agile team members to learn to work well together. Strive, therefore, to keep

team members together who have worked well together in the past. When forming a new team, consider how long members will be able to work together before some or all are dispersed to other commitments.

Some Common Objections to Self-Organizing Teams

Let's consider some common objections to relying on a team to self organize.

One Dominating Person Will Make All Decisions

A common concern is that one dominating personality, such as a tech lead, will decide that self-organization means he or she gets to make all decisions.

Or one dominating personality will force his or her will on the team before the team even has a chance to discuss an issue.

If you notice this starting to occur, take the dominating personality aside and inform her of the issue. Let her know that even in situations where she may know the "right" thing to do, she should sometimes refrain from voicing her opinion before others have a chance to express their thoughts.

Ask her if she thinks the team would make the right decision if she were to present her thoughts as an opinion rather than as an unchallengeable decision.

Enlist her assistance as a mentor to the others—her job should be not just making sure the right decisions are made but that team members grow such that they will make the right decisions on their next projects, where she may not be there for them.

My Team Expects the Scrum Master to Lead

A second common objection is that the team is too passive to self organize and will instead look to its Scrum Master or coach to lead and make important decisions for them.

If this happens, make sure team members know that the Scrum Master's job is to support them, not to make decisions for them. If you are in a role as a combined Scrum Master / team member, you do not need to suppress your opinions and keep quiet all the time.

However, you should look for ways to engage others by not making the decision in all cases. For example, try asking questions of others before giving your opinion.

The Team Is too Junior to Self Organize

A third concern is that team members are too junior to self organize.

I have never met a team too junior to self organize. If team members have enough experience to build a software product, they probably have enough experience to figure out how to organize themselves.

If not, provide the team with training or coaching.

Often, this objection really masks the objection of, "I don't trust the team to self-organize in the way I want them to." Too bad. Exert subtle control over the team in who you put together to form the team and the goal you give that team, not in how it self organizes to do its day-to-day work.

What's Your Experience?

Has your team embraced the idea of self organizing around their work? If not, what problems have you experienced? How have you overcome them? Please share your thoughts in the comments below.

Posted: June 5, 2018

Tagged: teamwork, self organizing teams, large teams

About the Author

Mike Cohn specializes in helping companies adopt and improve their use of agile processes and techniques to build extremely high-performance teams. He is the author of User Stories Applied for Agile Software Development, Agile Estimating and Planning, and Succeeding with Agile as well as the Better User Stories video course. Mike is a founding member of the Agile Alliance and Scrum Alliance and can be reached at hello@mountaingoatsoftware.com. If you want to succeed with agile, you can also have Mike email you a short tip each week.