Scrum Teams, Swarming, and Hardware

Kevin Thompson, Ph.D. www.kevinthompsonphd.com

Swarming and *Cross-Functional Teams* are two of the most important concepts in the adoption of Scrum for software-development work. They play out very differently in the worlds of software and hardware development.

Swarming Defined

Swarming is a particular strategy for assigning the members of a Scrum team to "Product Backlog Items" (deliverables to be produced) in a Sprint. It optimizes for completing the maximum value in a Sprint, in the face of substantial uncertainty about how much work each item entails. In other words, it is a risk-mitigation technique for coping with uncertainty.

Suppose the plan for the Sprint entails completing five items, in a specified order (rank), as shown in the diagram below. This Scrum Team contains eight Team members who do the hands-on work of building and testing deliverables, and their associated Product Owner and Scrum Master.

At the conclusion of the Sprint Planning meeting, the Scrum Team members work out initial assignments of Team members to the items to be developed. In this discussion, they assign a "swarm" (temporary group) of Team members to each item, working from the top down, until every Team member has something to work on. They typically do not define assignments for the remaining items at this time, but defer those assignments until later.

Day by day throughout the Sprint, different Team members will finish their work on individual items, and move on by seeking out the highest-ranked items where their presence will be beneficial. This real-time work allocation is commonly facilitated by the Scrum Master, who can provide insight to each Team member about where it would make the most sense for that person to go.

The logic behind swarming is that the best way to ensure that the Scrum Team completes the greatest value in the Sprint is for them to finish the highest-ranked items as swiftly as possible. This way, they complete the top and most valuable subset of the planned items even if they cannot finish all planned items.

The size of each swarm is constrained by the nature of the work to be done. For each item, there is a maximum number of people who can truly work in parallel on it. Exceeding this limit simply slows down work, and creates idle time.

The following diagram shows how swarms form for various items the Team members work on in a Sprint.

Cross-Functional Scrum Teams and the Impact of Specialization

The second concept is that of the cross-functional team. The intent is that each Scrum Team builds some aspect of the product, and has the appropriate skills to do so. As a Scrum Team's typical deliverables require multiple skills to complete, we ensure that the Team members have all the requisite skills across their membership.

Some degree of specialization is inevitable, and accounted for in the swarming discussions. The degree of specialization of skills has a large impact on how swarming works.

The members of software-development teams commonly have skills such as coding in one or more languages, user-interface development, database development, test-plan development, test-case development, test automation, and so forth.

The members of hardware-development teams commonly have skills such as mechanical engineering, electrical engineering, firmware development, and occasional specialties such as antenna design.

Software-team members can often be characterized as *generalizing specialists*. Each person has a deep specialty in some area but can do things of modest difficulty in other areas.

Hardware-team members are not commonly generalizing specialists. The mastery of each single discipline is sufficiently challenging that no one person is likely to be able to work outside of his or her specialty.

This difference means that software teams have typical swarm sizes of 2—3, while hardware teams have typical swarm sizes of 1. In other words, the concept of swarming as a way to reduce risk and optimize the value delivered in a Sprint is not nearly as effective in a hardware context as it is in a software context.

The inability to swarm for hardware development causes some distinct shifts in Sprint Planning. While the items are still ranked in the Sprint, and the ranking does address priority, it is much less definitive in generating any pattern of work. The Sprint Planning meeting generally does have to provide a detailed allocation of each Team member's time to tasks associated with the Product Backlog items. Each such assignment does have to address the actual hours of work each person is likely to be able to contribute in the Sprint, based on personal availability. As a result, the Sprint Planning meeting can expand from, say, two hours for software teams to as much as three hours for hardware teams.

These shifts make it worthwhile to address the question as to whether swarming can be enabled by changing some other aspect of the Scrum Teams. The answer is, "Yes," but at a prohibitive cost.

If the ability to swarm was a goal, instead of a tactic, we could organize the Scrum Team so that it was not cross-functional. We could make one Scrum Team of mechanical engineers, another of electrical engineers, and so forth. The overlap in skills across members of the same team would make swarming possible for each team's work.

However, that achievement comes with a large cost. It means that individual Scrum Teams are not focused on building aspects of the product, but on engineering disciplines. This means that different Scrum Teams must collaborate closely and frequently in order to complete the product. In effect, we've pushed the cost of cross-skill collaboration into Program Management, instead of keeping it inside each team.

The cost of collaboration is much higher across team boundaries than within team boundaries, so we have increased the overall cost of developing products. We've increased the time spent on collaboration, made change more difficult to manage, and increased the risk of failed handoffs across skill areas by increasing the number of such handoffs.

Conclusion

In software development, organizing Scrum Teams to have cross-functional membership and allocating work via swarming work well. In hardware development, these two things are at odds, and only one can be accomplished. The goal is always to develop a successful product at minimal cost, as quickly as possible, so I favor having cross-functional skills on teams over the ability to swarm. Risk and cost increase if we insist on forming single-skill teams in order to enable swarming within them.