Putting frequent, short meetings to work for your team **by Linda Rising**

AGILE MEETINGS

I KNOW WHAT YOU'RE THINKING: "OH, BOY, another article on meetings. Just what I need!" Believe me, I hate meetings as much as you do. In fact, I consider meetings the biggest time sink in organizations today. I can't believe I'm writing an article about having *more* meetings. But please bear with me. Let me share some stories about how frequent, short meetings helped a few teams solve significant problems. Suspend judgment for just a bit...

My first experience with having frequent, short meetings came as part of our organization's trial with agile processes. In particular, we were using Scrum. Although there is considerable overlap among the many agile processes practiced today, one component that set Scrum apart—the one that provided the most benefit for our teams—was the frequent, short meeting. You may or may not be interested in learning more about Scrum, but the meetings can be used by anyone, whether you are following an agile process or are CMM Level 5. Any team working toward a common goal will benefit because the practice improves communication and integration effort.

No one likes meetings, least of all those of us in the software community who are already under considerable schedule pressure. I believe this resistance to meetings exists because they are usually poorly run and often scheduled for no reason other than because it's time for the weekly/monthly staff report. The only reason our teams agreed to sign up for a trial with Scrum was sheer desperation. They knew things weren't working well, and they were astute enough to know that communication was a contributing factor.

QUICK LOOK

- Rules for short meetings
- 3 questions to always ask
- Success stories

The Meeting

As an integral part of the Scrum methodology, teams meet frequently—daily or every other day. The meeting usually lasts fifteen minutes, occasionally longer, but never more than thirty minutes. This provides enough time to address obstacles, but does not allow time to define solutions. Problem-solving discussion is held later, where only those who are involved in the problem attend. Having the meeting focus on raising but not solving issues is definitely a factor in making these meetings effective.

During the meeting, the Scrum Master (usually the team leader) asks three questions of each team member. Each team member answers those three questions. That's it. Meeting over.

Here are the three questions:

1. Relative to the Backlog (list of incomplete tasks), what have you completed since the last meeting?

2. What obstacles got in the way of your completing this work?

3. Relative to the Backlog, what specific things do you plan to accomplish between now and the next meeting?

This meeting achieves several goals. The most important is focusing effort on the Backlog items. It doesn't help to accomplish tasks that have nothing to do with the list at hand. Removing items from the Backlog is the focus of the team. But it's also important to communicate the priorities of Backlog items as you go. Some items in the Backlog are more important than others. The team needs to be reminded of the priorities to make sure the most important things are done first. The meeting keeps everyone informed of progress and obstacles, which allows the team to share in the success and pain of others. The protocol enables the team to resolve obstacles as quickly as possible. Finally, the meeting makes it possible to track the progress toward delivering Backlog functionality. Fred Brooks in his still relevant book, *The Mythical Man-Month,* said, "How does a project get to be a year late?" The answer, "One day at a time." The meeting identifies any obstacle immediately, and everyone knows about it.

Having frequent, short meetings keeps everyone on track. It's no secret that many of us are addicted to work. We put in long hours, look so virtuous, and seem to be getting so much done. We look productive, but often waste time on less important tasks. We are well intentioned, but working in isolation can make it difficult to set priorities that best serve the team.

It was always an eye-opening experience to see how many developers had a tendency to wander off in the weeds—intent on solving some problem that they felt was important but, even if solved, would not reduce the Backlog. In many situations, the Scrum Master had to corral these cowboys and convince them that solving their very important problem was not going to help the team make progress. Sometimes well-intentioned people wanted to create a homegrown tool that would (in the eyes of its creator) solve a recurring problem for teams in the future. It was astounding to see how many closet tool writers we had and how many experienced old hands underestimated the effort it would take to produce a truly useful tool for cross-project use.

A couple of warnings: (1) Stick to the meeting plan. It's tempting to get caught up in problem solving. Don't do it! The Scrum Master must be a strong facilitator and restrict comments only to answers to the three questions. (2) Only active participants are involved. It's okay to invite a manager or customer, but these outsiders may not speak. Sorry. Rules is rules! The manager or customer can hear the progress but may not comment. This prevents any side discussion.

Advice and Observations

The Network Tools team in my example was less than thrilled about trying a new development method. However, after only one meeting, they were singing the praises of Scrum. Here are some observations and advice from the team that extended indefinitely its one-month pilot with frequent, short meetings:

- The Scrum Master must have the skill to run a short, tightly focused meeting.
- Stick to the three questions. It's easy to get off the topic and extend a ten- to fifteen-minute meeting into a half-hour.
- Some people are not very good at planning their workload. The Backlog keeps people on track and aware of expectations. Answering the questions helps people structure their work and address tasks in a more organized fashion.
- There's an increase in volunteerism within the team. We're taking an interest in each other's tasks and are more ready to help each other out.
- The best part of the meetings has been the problem resolution and clearing of obstacles. The meetings allow the team to take advantage of the group's experience and ideas. As Gerald Weinberg has said, "None of us is as smart as all of us!"

Case Studies

The following case studies report two of my experiences with frequent, short meetings. The first team was made up of software developers who followed the tenets of Scrum pretty closely. The second team was made up of testers. Their process could be only loosely described as related to Scrum. This shows that you can see the benefits of frequent, short meetings regardless of your team's product and regardless of whether you're following an agile process.

From Reluctance to Enthusiasm: The Month-Long Trial

This team had a problematic history. It was a support team made up of people from several projects given the task of creating an integrated set of tools. It was unclear where this team should be in the organizational structure, and they were passed around from one department to another. Each new "home" provided a new team leader and a new management hierarchy. The members of the team were experienced, tough, independent thinkers used to solving problems on their own. Unfortunately, as a team, they were going nowhere fast and losing ground with each reorganization. Finally, they came to rest in a new department with a team lead who wanted to succeed.

The new team lead called for a project checkup to understand what was going on. The checkup uncovered several problems, most of them related to poor communication. This team was not especially enthusiastic about trying a new development method, but decided to pilot Scrum and the frequent, short meetings for one month.

Daily meetings seemed too much to tackle, so they set a schedule of Monday/Wednesday/Friday one week, followed by Tuesday/Thursday the next week. The meetings involved all developers, including telecommuters—some from different time zones.

I remember the first meeting well. We had to round up some team members who grumbled all the way, "I don't have time for this! I've got work to do!"

Once we started around the table, problems quickly surfaced. One person said, "I've been having trouble getting the database access to work." Another team member responded, "What? I didn't know you were working with the database! I thought I was doing that! We need to talk after this meeting!" The next person: "I've had trouble finding someone who knows about the code libraries we created on the Intelligent Network project." A helpful offer: "Hey! I used to work on IN. I can help you with that. Come on over to my office after the meeting."

Detailed problem solving does not happen in the meeting, of course; only an offer of help and an agreement to get together after the meeting. For example, someone would say, "I ran into a problem." Typical responses would be: "I had that problem a couple of weeks

For You Skeptics

who might think my experiences are a fluke, let me share some results that have been reported by others. The December 2001 issue of *IEEE Software* featured several articles on agile processes that addressed the topic of frequent, short meetings. Here's an excerpt from one team that used frequent, short meetings:

> **We also incorporated the daily stand-up meeting into the team. Each team took fifteen to thirty minutes to review their progress, and each person took a couple of minutes to talk about what he or she was working on that day and the previous day. The focus was not only on improving visibility but also on encouraging communication between team members. The qualitative results were immediate. Some engineers found it difficult to explain why they spent so much time on issues or why they looked at things that weren't high priorities on the team's list of tasks.**

Here's an excerpt from another team—one that *didn't* use frequent, short meetings:

> **We never introduced these [frequent, short meetings], and I believe it was a major mistake. Quick, daily, face-to-face meetings keep developers informed about what others on the team are doing. They help stop people from stepping on each other's toes. They keep the team lead informed about who's ahead and who's behind. They air new ideas and prevent people from duplicating work.**

ago. I can help with that. Let's talk offline," or "I know the people working in that area. I'll help you get in touch with them," or "I'm having the same problem myself. Let's get together after the meeting and talk about it."

What would have happened if the meeting had never taken place and the problem was never voiced? The person with the problem would have probably struggled long and hard before asking for help. Or maybe that person would have created a workaround or another tool that would have contributed nothing to the list of required deliverables.

As problems surfaced and were met head-on, the team began to come together. I could feel it and I could see that the team felt it, too.

Along with this, the team started to have hope. Suddenly, they had a way to get their arms around the problems they had been fighting. They were excited and energized—all in one short meeting. When is the last time you felt that any meeting accomplished *anything*—let alone a team transformation? It's hard to believe, but after only one meeting, this transformation was well under way and picking up momentum.

At the next meeting, most people were there early and were eager to hear what the others were doing. It was a 180-degree turnaround from reluctance to enthusiasm. These people could immediately see the benefits of the meetings and what they meant for their project.

As the meetings progressed there was a shift in emphasis. At the beginning, everyone was there to complain. But after a few team members had reported, everyone became a helper—a problem-solver. Instead of lobbing a nasty roadblock at the team, people began to ask for help. Instead of taking aim at someone who was having trouble, people began to take a real interest in seeing what they could do to help out. When one team member shared an obstacle, the resources of the entire team were brought to bear on that problem. The entire team immediately owned every problem.

The fact that the team solved all or most of their own problems helped clarify the Scrum Master's role for me. I originally thought that this role would be hard to fill. I thought that the Scrum Master would need to facilitate a tight meeting, keep everyone on track, and solve problems on the fly. But for these meetings, a good facilitator allows the *team* to solve the problems and only jumps in when an issue must be raised to a higher level. Even if the team is not using Scrum, this experience applies.

The team began to grow together and display increasing involvement in and delight with others' successes. Frequent, short meetings helped the team "synch up" and allowed in-house members to know exactly what remote team members had accomplished. The team completed a successful on-time delivery. The pilot was declared a success and, as far as I know, this team is still enthusiastically having frequent, short meetings. (See the "Advice & Observations" sidebar on page 44 for more about what the team members have to say.)

Improved Communication: Short Meetings during Testing

My second example comes from a test team that was working on preliminary testing and bug fixing for a feature when they tried frequent, short meetings. The team was scheduled for four testing times in eight days, so they planned a meeting before each test time. The team was on the critical path for delivery. It was imperative that the feature be delivered on time.

The big problem this team was facing was hit-and-miss communications. Important information was sent out via email but seemed to always be missed by at least one person on the team. Team members were on so many email distribution lists that their mailboxes were full. They received dozens of emails a day and could not keep up with them, so they usually didn't try. As a result, unless they happened to see someone who could remind them of an update, they sometimes went about their work without knowing critical information. In addition, the testers sometimes worked strange hours, and team members didn't often see one another or communicate plans and results for a test session. This caused problems when loads were not ready or fixes had been added.

During the meetings, team members could ensure that the proper load was being used for the next scheduled test time and that the fixes that had been added were announced. The team redefined some testing procedures to make things go faster. The meetings allowed all the testers to hear what was planned and to volunteer to work the next test time. The meetings also served to remind everyone of current load problems and events scheduled for the next day. Decisions about the kind of testing to perform in the next test time were made by the group instead of just the tester who worked that time.

During one meeting, the team lead had to make a phone call to help decide whether to go ahead with the next scheduled test time. Without the meeting, without the phone call, without the information, the tester would have worked an entire shift on an obsolete load—a complete waste of time.

I saw the regularly scheduled meetings provide the team with an efficient way to share information and track progress. The meetings kept everyone together. The team met its goal—the feature was ready for integration testing at the end of the trial.

Conclusion

In his novel *The Deadline,* Tom DeMarco observes that projects need ceremony and suggests that projects *use* ceremony to focus attention on project goals and ideals. This is certainly one of the roles of frequent, short meetings. The agenda is the same for each meeting: ask the three questions of each participant. This small ceremony is the way most teams at our company began each day. If you were standing on the second floor of the building where product development took place, you would see clusters of teams gathering between 8:00 and 9:00 in the morning—team members standing with a cup of coffee, listening intently to the answers to the questions and then, *poof!,* the meeting is over and everyone is back in their offices, working with a clear sense of where the team is heading.

This is such an easy process improvement. Try it with your teams. Frequent, short meetings have a small cost and a tremendous payback. Every day or every other day, get the whole team together, form a small circle, and have the team leader ask the three questions.

Make sure that asking and answering these questions is the only thing that happens. This should take fifteen to thirty minutes—and it's a good way to start the day. STQE

Linda Rising is the author of numerous articles and three books. She is currently writing a book with Mary Lynn Manns, Introducing Patterns (or any Innovation) into Organizations, *to appear this fall. You can reach her at* www.lindarising.org *or* risingl@acm.org.