



72 | Chapter 4 Iterating Toward Agility

a wiki, and nothing more. The amount of time an IC member spends on the community is determined by each individual, the person's boss, or organizational culture.

OBJECTION

"Scrum teams are supposed to be self-organizing. Doesn't an ETC conflict with this? Shouldn't teams get to decide what they want to improve at?"

Self-organization occurs in response to a challenge taken on by a group of individuals. For a development project, the company may tell a team, "Develop this software to run faster and take less memory than the current version and do it two months faster than we've done in the past." Individuals then organize themselves around how to achieve that goal. It is no different with the ETC. An ETC states what it would like to see improved but not necessarily how to achieve that improvement. The how is left up to the improvement communities or Scrum teams.

Additionally, keep in mind that an ETC's biggest goal is to create an environment such that improvement communities identify their own goals and form spontaneously to address them. We will look at self-organization in detail in Chapter 12, "Leading a Self-Organizing Team."

Catalysts for Improvement

Communities, when used as part of the effort to adopt and get good at Scrum, become catalysts for improvement. Consider the case of Google, where improvement communities are called "grouplets." Google's Testing Grouplet was formed "to drive adoption of developer testing" (Striebeck 2007). Bharat Mediratta founded the community and describes its activities.

We started with engineers from all over the company meeting every couple of weeks to brainstorm. Slowly, over time, we started turning into activists, planning to actually start improving things. We started building better tools and giving informal talks to different technical groups. (Mediratta 2007)

Notice that although this community met initially to brainstorm, they soon found themselves as activists with plans for actual improvements. Improvement communities act. This is why they aren't called task forces, work groups, committees, or any of the other terms that too often bring to mind ineffective groups. If the Google Testing Grouplet had merely created presentations on the benefits of developer testing, or if it had chosen to convince a powerful vice president to mandate developer testing, its efforts would have been fruitless.





What the testing community at Google did instead was find direct and immediate ways to help teams. Mediratta recalls how, in addition to building tools, the community found a unique way of providing concrete, short examples and advice about testing.

One day, toward the end of a long brainstorming meeting, we came up with the idea of putting up little one-page stories, called episodes, in bathroom stalls discussing new and interesting testing techniques. Somebody immediately called it "Testing on the Toilet," and the idea stuck. (2007)

The most effective communities are usually those that form not in response to management dictate but because company culture or the ETC has created an environment in which communities can naturally emerge. J. F. Unson, a coach at Yahoo! during its large-scale Scrum rollout, says this is exactly what happened at one of Yahoo!'s remote facilities.

At Yahoo!, in our Santa Monica campus, all the entertainment agilistas started a monthly ScrumMaster lunch. This happened organically as Scrum started to grow in the organization, without having the agile group [ETC] pushing it. (2008)

Not all communities will form in such an organic manner, of course. Especially during the early weeks or months of adopting Scrum, the ETC will need to encourage an improvement community to form by highlighting the importance of a goal and then hoping a community forms around that goal. Occasionally, an ETC may need to go so far as to ask someone to form a community around a specific goal.

Two Metrics for Effectiveness

Professor Jeffrey Goldstein has written, "Change does not need to be imposed; it simply needs to be released" (1994, 32). You can gauge how well the ETC is doing at releasing change in two ways:

1. The number of improvement communities that have formed without a direct request from the ETC
2. The percentage of such improvement communities to the total number of improvement communities

If the number of spontaneously formed improvement communities is high, and especially if these represent a majority of the total number of communities, this indicates strong interest in Scrum and the changes it is creating. If these metrics are increasing or remain high over time, the organization is well on its way to becoming agile. You should, of course, look at other metrics. These are just two that I like.





74 | Chapter 4 Iterating Toward Agility

An Improvement Community Sprint

As you might suspect, ICs perform their work in sprints as well. As with the ETC, each IC can select its own sprint length, but two weeks is the recommended length. An IC that was formed spontaneously will usually serve as its own product owner, with members of the community electing to devote their time to the improvements they are the most passionate about. An IC that was formed in response to an ETC-identified goal, on the other hand, will usually work with a member of the ETC as its product owner to plan a sprint.

That being said, an improvement community does not exist to serve the ETC. It exists to serve its customers: the Scrum development teams who are building products or systems. Although an ETC member will act as product owner for some improvement communities and will serve as the official product owner for the sprint reviews, you should expect members of interested development teams to be active participants as well. Additionally, the wise ETC understands that the best results will be achieved when improvement communities are given broad latitude in achieving their goals. In practice, this means an IC, even one formed in response to ETC-identified goals, will be responsible for prioritizing its own work, while balancing the needs of the organization to improve in particular ways and its members' passion for working on those issues.

During its sprint planning meeting, each improvement community selects one or more things it can commit to completing during the sprint. If an improvement community has formed in response to a specific goal of the ETC, sprint planning begins by taking an item from the ETC's backlog and breaking it down into smaller items that will be placed on the improvement community's improvement backlog. The best way to see this is with an example.

The ETC improvement backlog shown in Table 4.1 on page 64 includes the item, "Establish an internal program for developing ScrumMasters." An improvement community formed a month after the ETC put that on the improvement backlog and made it known to the rest of the company that creating such a program would be valuable. There were three people in the community initially, but that was plenty to make progress toward this goal. In their first sprint planning meeting, they discussed the ETC's goal ("Establish an internal program for developing ScrumMasters") and created their own improvement backlog of what they would do to achieve this goal, which is shown in Table 4.2.

Also during sprint planning, the community members took some of the items in Table 4.2 and identified the tasks necessary to complete each. For example, for the final item in Table 4.2 (working with local groups to share the expense of bringing in speakers), the community identified the following tasks:

- Search web to see what user groups are in our area.
- Create budget of expenses.



- Send e-mail to internal distribution lists to see if anyone here is connected to these groups.
- Set up phone calls to introduce ourselves and what we're doing.
- Conduct phone calls. See if any groups have previously split the cost of bringing a speaker into town with another company. See if any will work with us on this.
- Meet with Susan to go over budget and get approval.

What	Note
Figure out how to identify good candidates to become ScrumMasters (in addition to those who ask to participate in this program).	
Establish an internal mentoring program.	
Develop some internal classroom training. Which courses? Who can teach them? Develop our material, or can we license it?	
Determine which classes we can teach internally.	
Get budget for next year for external coaching. How many days? At what expected daily rate?	James has already asked for rates from three coaches.
See what we can do with local user groups to share the expense of bringing in speakers.	Savannah has contact in local Scrum lunch meetup group.

TABLE 4.2

An improvement community's backlog for establishing an internal program to develop ScrumMasters.

As in a development team's sprint planning meeting, the community then estimated each item and decided they could commit to completing these tasks during the sprint. Two weeks later at its sprint review, this team showed its product owner, a member of the ETC, a list of local user groups and a plan to work with one of them twice a year, sharing the expenses of bringing nationally known speakers into the area.

- ❑ Add to your improvement backlog by looking at the section headings of the chapters in this book. Many of them were written with this possibility specifically in mind.
- ❑ Review any notes available from recent sprint retrospectives. These are often an excellent source of improvement backlog items.

THINGS TO TRY NOW



76 | Chapter 4 Iterating Toward Agility

Focus on Goals with Practical Relevance

For an improvement community to have the most impact, its members must focus on goals of immediate and practical relevance to the development teams using or attempting to get started with Scrum. The best way to do this is for improvement community members to work side by side with development team members on something important to the development team. This is what Google's "test mercenaries" do. Test mercenaries are members of the testing community who are experienced engineers with a passion for and expertise in testing. They spend up to 20% of their time for three months on a project other than their own. During this time they add tests and refactor code as a direct help to the development team.

I suppose that test mercenaries could instead spend this time creating presentations and spreading the gospel of developer testing. Something tells me, though, they are better able to achieve their goals by working with a team rather than preaching to it. A development team that has had the help of a test mercenary ends up with improved code and more tests. It also witnesses the benefits of an additional focus on developer testing. This works wonders in motivating those on the Scrum development team to continue the effort after the mercenary moves on to another team.

Focusing on providing practical assistance to development teams also helps keep improvement community members from falling into the habit of preaching to the development teams. A common problem when adopting Scrum is that the early adopters often become zealots anxious to convert everyone else. What zealots often forget is that it took them time to get comfortable with the idea of Scrum and the changes it requires. When others fail to convert instantly, zealots often perceive the delay as resistance. Because zealotry and pushing others to rapidly adopt new ideas can cause more harm than good, it is important for improvement community members to understand that their role is to consult rather than preach (Allen-Meyer 2000c, 25).

Improvement Community Members

Organizational change expert Glenn Allen-Meyer says that change should be done "with, not to, the people expected to change" (2000b). Because of this, it is important that anyone with a passion for an improvement opportunity be encouraged to participate in its community. Membership should not, for example, be restricted to only the organization's most senior employees. Broad participation in improvement communities helps everyone in the organization feel that change is occurring with them rather than to them. There should be no limit on the number of people participating in an improvement community. Communities often include well over 100 members, with individual participation levels going up and down over time based on the other demands of each person's job.





Participating in a community is not meant to be a full-time job; it is something someone takes on in addition to regular work. Improvement community leaders at IBM are asked to contribute two hours per week, although many contribute more based on a desire to see more rapid progress. A participant's manager, product owner, and ScrumMaster, though, are responsible for ensuring that those passionate enough about a change to work toward it are given sufficient time to do so. Google accomplishes this by telling each employee to spend 20% of each week on something of interest. The time could be spent, for example, exploring a new product idea or participating in a community.

Successful Scrum adopter Salesforce.com has a similarly innovative approach it calls PTON, pronounced pee-tee-on and meaning "paid time on." Patterned after the common PTO ("pee-tee-oh") policy for paid time *off* in many companies, Salesforce.com's PTON program gives employees dedicated time at work to pursue initiatives of their own choosing. Each employee is given one week of PTON for each year with the company. Salesforce.com employees can use the PTON time to work on a community initiative, explore new product ideas, or do just about anything they want.

Google's 20% policy and Salesforce.com's PTON programs were not created specifically to allow people to work in an improvement community. And organizations do not need to make such dramatic changes just to get started adopting Scrum. An easy starting point is simply for managers to commit to freeing up some number of hours each week for those who want to work on an improvement community.

"We've been working on this new product for a year. We ship it in four weeks and, as the product owner, I need the team's full time and attention for the next four weeks."

OBJECTION

Absolutely. Team members probably already know this and have plans to scale back participation in any communities to the minimum possible over that period. A team member who generally feels valued and allowed to devote time to the longer-term initiatives of a community will willingly minimize community participation during a true crunch period because she knows she can devote more time to it later.



OBJECTION

"These improvement communities seem just like the Software Engineering Process Groups (SEPGs) our company created to push CMMI. Isn't this just a new name for an old idea?"

Not really, but I can understand why you might think so. Both ICs and SEPGs are focused on helping the organization improve how people develop software. However, while their goals are the same, an SEPG and an IC differ in a few subtle but important ways:

- An SEPG looks at the process and answers the question, "What could we improve?" Members of an improvement community look at their own projects and ask, "What could we improve?" and "What are we doing well that others should know about?"
- Some SEPGs force compliance with a process; an improvement community has no authority from which to force compliance.
- Some SEPGs are chartered to look only at portions of the overall development process. ICs are encouraged to look beyond the product development process to find improvement opportunities.
- Improvement communities are self-motivated and self-organizing. In general, no one is told to join an improved community. (Although this may occasionally be done to start a new community.)
- Members of an IC are more likely to take an experimental, try-it-and-see approach to process improvement.
- Improvement communities are ad hoc and organic, formed whenever passion for a topic brings people together. SEPGs are formally created and often discouraged from functioning in an ad hoc manner.

Disbanding a Community

Most communities will eventually disband. A community formed to promote automated testing, for example, may exist with members coming and going for years as long as that is an area in which the organization needs to improve. Eventually (at least we'd like to think), the organization becomes good enough at automated testing that those community members can contribute more by devoting time to other improvement communities and the opportunities they represent.

Regarding the ETC specifically: It should disband once the organization has realized its transition to Scrum and has entered a phase of continuous improvement. The ETC exists only during the transition period, which may be multiple years for a large transition.



- ❑ Identify an improvement you are passionate about. Ask two or three coworkers to help you. Create an improvement backlog and plan a first sprint. Even if you can manage only an hour a week on it, start. As you begin to make progress, incorporate your improvement in the work of your team or offer it to another team. Generate interest by telling (or, even better, showing) others what you've accomplished.

THINGS TO
TRY NOW

One Size Does Not Fit All

In this chapter, I've presented a community-driven approach to Scrum adoption. A guiding community—the Enterprise Transition Community—does some of the work of the transition, but most important it creates an environment that encourages other communities to form. These communities—called improvement communities—are formed when a group of employees choose to work together to improve the organization's use of Scrum. Both types of communities use Scrum to drive the organization toward becoming agile.

But one size clearly does not fit all. The approach I am describing in this chapter works well when transitioning a medium or large department to Scrum. Scale it down as appropriate. A software department of 20 professionals, for example, may benefit from having one group of passionately agile individuals who help drive change and improvement. They are both an ETC and an IC in that case.

Looking Forward

So far, in the chapters that make up the initial section of the book, we've discussed why transitioning to Scrum is hard, but worth it. We've talked about the activities that accompany change and some tools you can use to help people make the switch to Scrum. We've discussed patterns for adoption that can guide our general approach to transitioning to Scrum. Finally, we've looked at how to combine all of that information, and the Scrum process itself, and use it to manage a Scrum adoption, on any scale. Throughout the first four chapters, I've made a point of saying that, unlike other change initiatives, with Scrum there is no end state. There is no point when you're done. Instead, Scrum requires continuous improvement, which can be managed through improvement communities, using Scrum itself.

In our next chapter, we discuss how to pick your first project, your first team, and get started with the business of becoming agile with Scrum.



Additional Reading

Conner, Daryl R. 1993. *Managing at the speed of change: How resilient managers succeed and prosper where others fail*. Random House.

In this book, Conner describes eight key patterns of how people behave during organizational change. One of the goals of his process for change management is to foster resilience in people and organizations. His view of resilience is compatible with this book's presentation of change as continuous and agility as something to be iterated toward.

Katzenbach, Jon. R. 1997. *Real change leaders: How you can create growth and high performance at your company*. Three Rivers Press.

Katzenbach's book is based on extensive interviews with individuals who he found to be the true source of change in organizations. These are the "real change leaders" of the book's title. The book contains many engaging stories about individuals who would make good improvement community members.

Kotter, John P. 1996. *Leading change*. Harvard Business School Press.

Kotter's highly respected book is a classic on organizational change. In it, he lays out an eight-step process for creating change. In his second step, Kotter advocates the creation of a guiding coalition, which has some similarities to an ETC. Additionally, his article in *Harvard Business Review* (1995) offers a concise summary of this book.

Schwaber, Ken. 2007. *The enterprise and scrum*. Microsoft Press.

In this book, Schwaber, the coinventor of Scrum, describes what is necessary to transition an entire organization to Scrum. Included is advice on the improvement backlog and on the Enterprise Transition team, which is similar to the Enterprise Transition Community as I have presented it.

Wenger, Etienne, Richard McDermott, and William M. Snyder. 2002. *Cultivating communities of practice*. Harvard Business School Press.

Wenger is recognized as the authority on communities of practice. This highly readable book describes everything you need to know to begin cultivating communities within your organization, including a chapter dedicated to advice to community coordinators.

Woodward, E.V., R. Bowers, V. Thio, K. Johnson, M. Srihari, and C. J. Bracht. Forthcoming. Agile methods for software practice transformation. *IBM Journal of Research and Development* 54 (2).

Members of IBM's Quality Software Engineering organization are using an approach very similar to the one described in this chapter to spread agile throughout IBM. This excellent paper describes how they function as an Enterprise Transition Community, ways in which they encourage improvement communities to form, and how they use the Scrum framework to drive improvements in how they use Scrum.