The Addison Wesley Signature Series

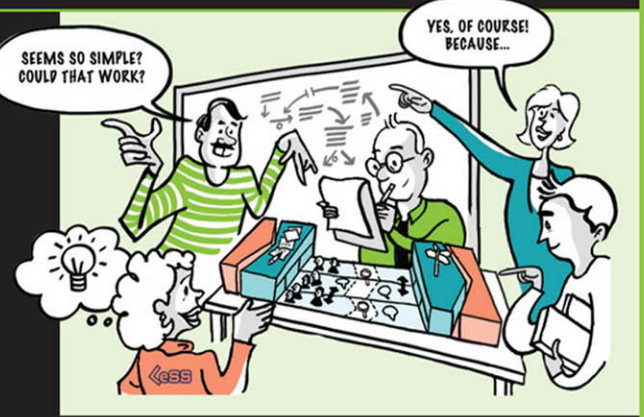
A Mike Cohn Signature Book

Large-Scale Scrum

More with LeSS

Craig Larman
Bas Vodde

Foreword by
Stephen Denning



With illustrations by Sketch Post

Large-Scale Scrum

especially that product group became one of the reasons the adoption eventually took a few steps backwards. Large-scale organizational change is a breeding ground of nasty politics.

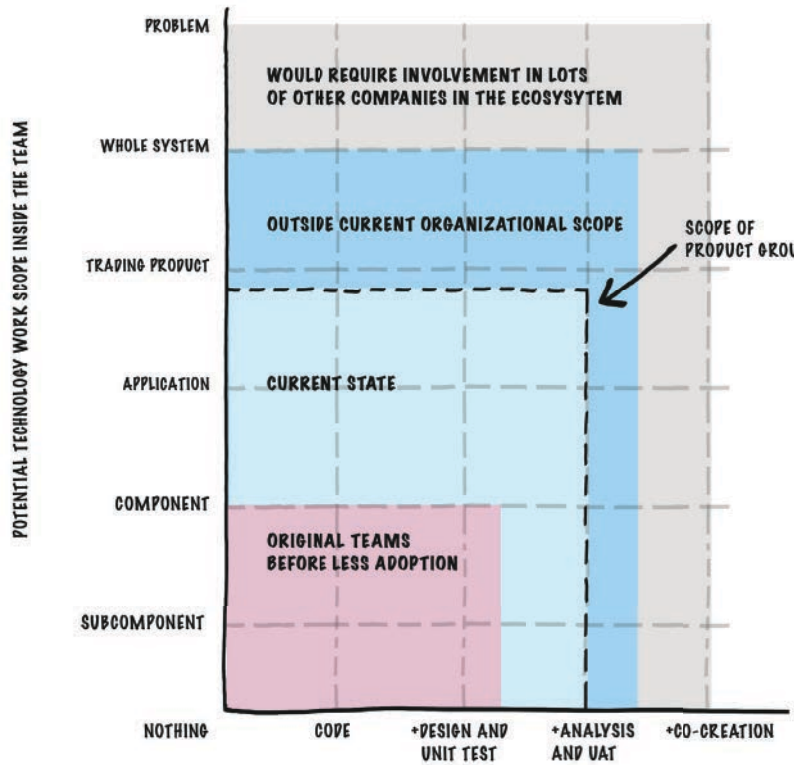


Figure 4.7 feature-team adoption map of financial trading system

Help with Decisions

A feature-team adoption map is an important tool when you are adopting LeSS. It helps with the following decisions:

- **What is "all"?**—A smaller LeSS framework adoption requires an *all-at-once* change to feature teams. Who is included in "all" depends on the scope of the feature teams.
- **Future improvement goals**—The map can be used for setting future goals as the telecom product group had done. These future goals frequently go hand in hand with the expansion of the Defini-

tion of Done. The map also shows the expected changes and their difficulty, as expanding beyond the current organizational boundary involves the hard work of "political" boundaries.

> **LeSS or LeSS Huge?**—The scope of the feature teams influences the size of your adoption and can swing a LeSS group into adopting LeSS Huge instead. For example, a network-performance tool is a customer-centric product and its development-group size leads to the smaller LeSS Framework. However, when realizing it is *always* sold as an integrated part of a network management system changes the product scope and then is likely a LeSS Huge adoption.

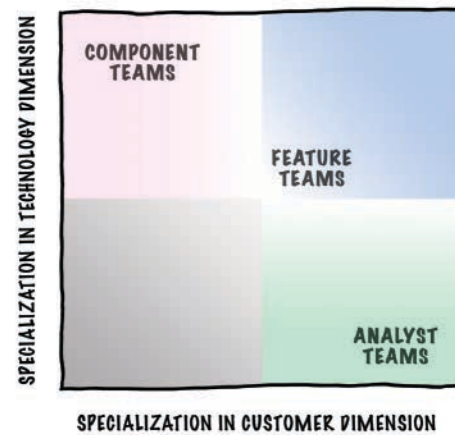
Guide: Prefer Specialization in Customer Domain

One essential concept behind feature teams is to organize and specialize in the domain of the customer rather than the domain of technology. The same concept also guides other LeSS structuring decisions.

A common misunderstanding of feature teams is that it leads to abandoning specialization altogether. Part of this misunderstanding comes from the false dichotomy of either specializing in one component or not specializing at all—which we've covered extensively in our writing on feature teams. Part of the misunderstanding comes from the belief that specialization is a one-dimensional thing—specializing in a component. But specialization is multi-dimensional. Exploring these dimensions leads to better decisions on how to balance these.

The conventional way of thinking about specialization is almost exclusively around functional skill or components, as shown in a feature-team adoption map. But other dimensions of specialization exist; they include programming language, hardware, operating system, API, market, type of customer, and type of feature. We can group these as (1) technological (component, OS, etc.), or (2) customer-oriented (market, type of feature, etc.). Looking at the adoption of feature teams from these dimensions leads to the chart in Figure 4.8.

Figure 4.8 two dimensions of specialization



LeSS brings users and developers closer together. The user perspective is almost always lost in traditional large product groups. Feature teams are one way of organizing by customer value, but not the only one. The principle of preferring specialization in customer domain also leads to other structuring decisions.

For example: Banks create mobile apps for banking services on mobile devices. The teams are typically organized by platform such as the iOS teams and the Android teams. These teams are feature teams *and* they are specialized in the technical dimension—namely, the platform. Alternatively, they can be organized in customer domains such as mobile payments, admin, and reporting. This leads to the teams implementing the same type of features on multiple platforms instead of implementing many types of features on the one platform.

Which specialization dimension is better? Traditional organizations tend to specialize in technology dimensions. Why? Perhaps technology is perceived as more difficult and hence specializing in that dimension leads to faster development? LeSS prefers specialization in the customer domain to increase collaboration with real users, remove handoffs, and make work more meaningful. Let's explore another example…

We worked with a company that builds graphics cards. They structured their organization around technology: (1) hardware team, (2) Linux-

driver team, and (3) Windows-driver team. These are component teams, but a move to feature teams requires a cross-functional hardware/software team. That's possible but difficult to achieve in most *hardware* companies—for cultural reasons. The software teams are additionally specialized in driver API. This organization is predicated on the assumption that learning the OS-driver-API is more important and difficult than understanding the hardware—the company's product. LeSS prefers organizing around the customer and thus an alternative team organization is (1) 2D-graphics chip team and (2) 3D graphics chip team.

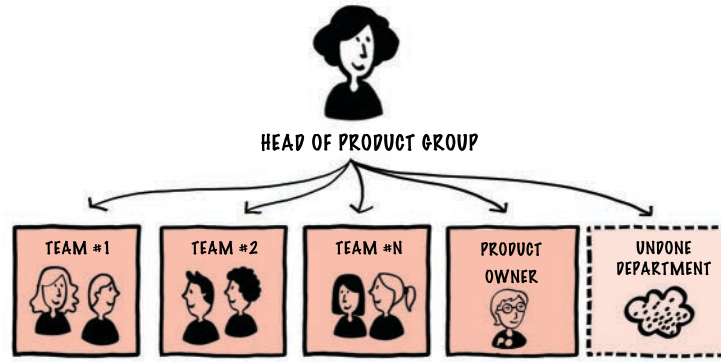
What is the perfect balance between technology specialization and customer specialization? A hard decision. When adopting LeSS, *prefer specialization in the customer domain.*

Guide: LeSS Organizational Structure

How does this all fit together in an organizational structure? Of course, each organization is different, yet LeSS organizations tend to follow a surprisingly simple structure. The first difference between LeSS organizations and most traditional ones is that the structure is stable as (1) work is organized around teams and (2) mismatch of skills triggers learning and coordination within existing teams.

A typical LeSS organizational chart is shown in Figure 4-9.

Figure 4.9 typical LeSS organizational chart



Notice what isn't here:

- **No functional organizations.**
  Having team members with programming skills report to the development manager and team members with testing skill report to the QA manager won't create great teams. Why? It causes *conflicting loyalty* where a QA person has one loyalty towards the team for all the work of the team and one loyalty towards the QA manager for his functional specialization. LeSS organizations avoid this conflict by abolishing functional organization and instead creating cross-functional line organizations.

- **No project/program organization or project/program management office (PMO).**
  These traditional control organizations cease to exist in a LeSS organization as their responsibilities are distributed between the feature teams and the Product Owner. Insisting on keeping such organizations will cause confusion and conflicts of responsibilities.

- **No support groups such as configuration management, continuous integration support, or "quality and process".**
  LeSS organizations prefer to expand the existing teams responsibility to include this work over creating more complex organization with specialized groups. Specialized support groups tend to own their area which leads to them becoming a bottleneck.

Let's examine a LeSS organization…

Head of Product Group—Most product development LeSS organizations still have managers including a "head of product group." They support the teams by Go See and help them remove obstacles and improve. (We cover manager responsibilities in the *Management* chapter.) LeSS organizations don't have matrix structures and there are no dotted-line managers.

See *Management* chapter for more on the role of management.

The name "Head of Product Group" might be confusing to you. This is probably because different organizations use quite different terms for this. What we mean is the line manager of all the teams, whatever that is called in your organization.

Feature teams—This is where the development work is done. Each team is a cross-functional, self-managing feature team with a Scrum Master. They are permanent units that stay together for the duration of a product (and sometimes longer). Preferably all members have the head of the product group as their direct manager. We've seen 150 people who all had the same direct manager as most management activities were taken over by the teams. But some larger LeSS organizations have some additional team line manager structure. Try to avoid the additional organizational complexity whenever possible.

Product Owner (Team)—This is also commonly called "Product Management." It can be one person but in a larger LeSS organization the Product Owner might be supported by other product managers.

An important point in this organizational structure is that the Teams and the Product Owner are *peers*—they do not have a hierarchical relationship. We have found it important to keep the power balanced between the roles. The Teams and Product Owner should have a *cooperative* peer relationship to together build the best possible product, and a peer structure supports this. This point is further explored in the *Product Owner* chapter.

This organizational structure is especially common for product companies. The frequent alternative, especially for internal development, is for the Product Owner to belong to a different organization—the business side. Thus, he is *not* within the hierarchy of the Head of the Product Group. This is recommended, though it does often require additional

Guide: Who Should be Product Owner?, p. 173