
Less is more

## Intro

Scaling Scrum starts with understanding standard one-team Scrum. From that point, your organization must be able to understand and adopt LeSS, which requires examining the purpose of one-team Scrum elements and figuring out how to reach the same purpose while staying within the constraints of the standard Scrum rules.

Agile development with Scrum requires a deep organizational change to become agile. Therefore, neither Scrum nor LeSS should be considered as merely a practice. Rather, they form an organizational design framework.

### Two Agile Scaling Framework

LeSS provides two different large-scale Scrum frameworks. Most of the scaling elements of LeSS are focused on directing the attention of all of the teams onto the whole product instead of “my part.” Global and “end-to-end” focus are perhaps the dominant problems to solve in scaling. The two frameworks – which are basically single-team Scrum scaled up – are:

---

### What does it mean to be the same as One-Team Scrum?

LeSS is a scaled up version of one-team Scrum, and it maintains many of the practices and ideas of one-team Scrum. In LeSS, you will find:

---

- a single Product Backlog (because it’s for a product, not a team),
- one Definition of Done for all teams,
- one Potentially Shippable Product Increment at the end of each Sprint,
- one Product Owner,
- many complete, cross-functional teams (with no single-specialist teams),
- one Sprint.

In LeSS all Teams are in a common Sprint to deliver a common shippable product, every Sprint. (1)

1. LeSS Large Scale Scrum - http://less.works/less/framework/index.html

# Principles

## Large Scale Scrum is Scrum –

LeSS, which can be defined as Large Scale scrum, is scrum. Using LeSS allows your teams to continuously adapt your processes to make improvements in an attempt to reach perfection and deliver a perfectly finished product to your customer. To implement scrum, people gather together to form teams. These teams perform various sprints to deliver features that match the definition of done and can be delivered into production to meet the customers’ requirements. These teams perform several two week iterations of these sprints in attempt to improve their processes after each iteration.  There is a review after each iteration which exposes weakness and provides transparency to other teams, the product owner, and the customer. Large Scale scrum is using scrum in a large scale environment such as a large enterprise consisting of thousands of employees and products. Unlike small scale scrum where one team may work alone on a product, large scale scrum is

comprised of several teams working on a single product. These teams interact with one another to exchange ideas and transfer knowledge between one another. More information on large scale scrum and this principle that large scale scrum is scrum can be found at less.works.

## More is LeSS

Identify the key contents of the ebook, keeping in mind the primary end customer and understanding the subject matter precisely. Content and details should be precise and accurately represented, with adequate references. The sentences should be short, concise and ideas presented should be well defined, with only the required information.

The approach taken should be actionable and easily applicable to serve the customer's business needs. This can be achieved by taking the primary subject and breaking it down into smaller components to help the sprints complete the chunks in time allocated.

The smaller chunks can then be periodically integrated to fact check that indeed the focus is maintained. The sprint teams should be allowed to plan, organize and execute the smaller chunks of work according to their convenience.

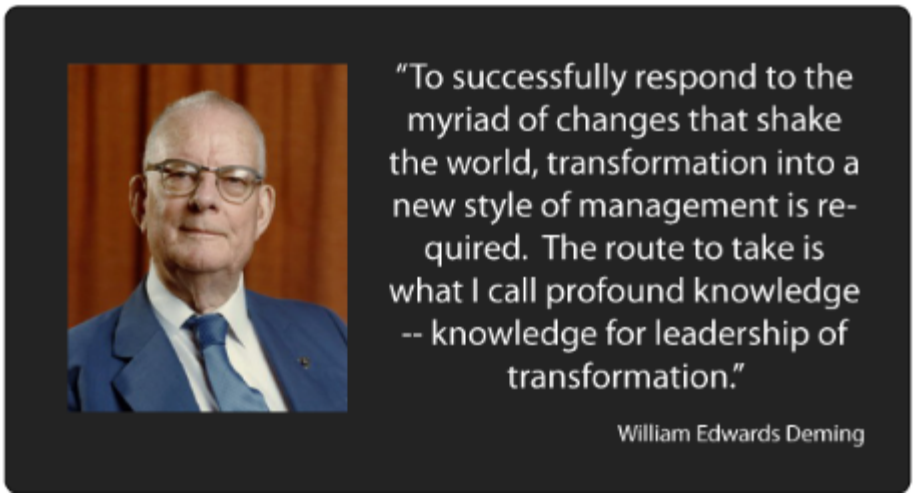
The sprint team should only be required to perform their assigned sprints without any additional disruptions. Teams should have autonomy to take on additional left over required sprints if they finish ahead of scheduled time.

Appropriate resources and tools should be made available to the teams to help perform optimally.

## Lean Thinking

The process of LeSS and Lean Thinking would follow a philosophy of Deming's Total Quality of Improvement focusing on an Agile approach of process improvement. The principles outlined that focus on this process is identifying the Value, Value Stream and enhancement areas that reduce waste within the assigned methodology. The Lean thinking process continues with identifying the wasteful areas within the principles of the workflow and utilizing the pull production model. Finally, perfection is what one will strive forward to when utilizing the Lean thinking process.

Deming states, "to successfully respond to the myriad of changes that shake the world, transformation into a new style of management is required. The route to take is what I call profound knowledge – knowledge for leadership of transformation."




Along with that, the Lean Thinking process can be related to the true benefit when focusing on Agile and Scrums. Utilizing this methodology and process allows the Retrospective Meetings to take place. This retrospective allows the team to openly discuss, what we did well, what we did not so well and how we can improve. This meeting takes place after every sprint to allow the empirical process of Total Quality Improvement to continually cycle. This ultimately showcases how Lean thinking can be applied to outline success for the LeSS Lean Thinking Agile Methodology.

Sources

http://www.panview.nl/en/lean-production/lean-thinking-jpwomack-dtjones-summary

http://quotesgram.com/deming-quotes-on-management/

## Systems Thinking

The following is a very short and incomplete description of one of the LeSS principles called Systems Thinking.

Systems thinking gives us a way to address lots of problems in everyday life.

It is a way of understanding reality that shows connections and relationships among a system's parts, rather than the parts themselves.

Systems thinking makes it possible to find smart design solutions to problems with long-term view.

A system is a group of components that make up a united whole. Focusing on a component without consideration of a whole system may is a sub-oprimization that can create a short-sighted solution and actually hurt the whole system. An example could be a human body. When a doctor needs to find a treatment for high fever he needs to consider many other conditions of a person so not to make it worse. Any change to a component of a system may affect other components and a system as a whole.

Emphasis on the interconnection and interdependency of components is in the base of Systems Thinking.

In software development boundaries of a system are different and extend with the maturity and a level of a developer. A component for one can be a system for another.

## Empirical Process Control

This is a key principle in the LeSS methodology. The process is not fixed, it is instead a framework within which the product development team can employ various processes and techniques.

With the short sprints, iterations of development, it enables the team to build small pieces of shippable product and at the end of the sprint review what was built and how it was built. From this learning practice, called the retrospective, the team learns and adapts the product and how the product is built.

The software industry has taken from the Taylor principle, based on the industrial revolution and interchangeable components, that the way to scale software development is to create component teams and layer with excessive management. This is a false practice because building software products is very complex with unknown variables and unknown unknowns. It is not realistic to build complex software solutions this way.

At the heart of Empirical Process Control is that we all learn as a group and adapt the product and the build process to improve. After the retrospective the team then agrees on improvements to be made and adjusts their practices which may include redefining the definition of done, obtaining better tools for product development and/or improving the overall process for better product quality.

1. Source: LeSS.works - http://less.works/less/principles/empirical_process_control.html

## Transparency.

Definition by Webster:

- Able to be seen through
- Easy to notice and understand
- Honest and open

Scrum is not a going to magically solve software delivery process. It will not create hyper productive teams. But it is a framework that dramatically increases transparency. Transparency becomes a mirror that shows the team how good or bad they are at making a product. Without transparency, it is hard to steer or adapt. With it, adaptive control and improvement is possible. The bad news is that transparency uncovers previously unseen weakness amongst people, processes, tool and environments. Transparency is also related to Definition of Done. Formally defining the meaning of ‘done’ reduces variability and the likelihood of undone work and measuring progress unambiguously (‘done’ or ‘not done’) increases transparency.

## Whole Product Focus

Whole Product Focus is one of the principles of LeSS. The basic idea is that the best way for team members to understand what they are producing is to have a good sense of what is the end product or result that is being crafted. If that understanding is not there, the issue becomes like the fable of the blind men who tried to know what an elephant looks like based on what part of the animal they had taken a hold of, as they had no way no knowing there were other parts to the animal.
If a person is only aware of one part of a project they may not be able to ensure that the piece that is being worked on truly fits on the whole. There is also a much better sense of purpose when a person comprehends the full picture of what they are producing. That sense of being a part of a greater endeavor can drive the person doing the work to be more enthusiastic. They will see the full picture of what Is needed rather than just s segment. That can make a great difference in purpose.

## Continuous improvement towards perfection

LeSS framework encourages to have releases at pre-defined intervals. The recommended interval to go to production is 2

weeks. There are many examples available in the industry showing how companies and individuals got better at making or

doing things. We will still not call them perfect because there is no such thing as perfection. There is nothing in the

world which cannot be improved. I recommend the readers to watch the documentary JIRO Dreams of Sushi which is available on

Netflix. This is the story of an 85 year old Japanese man who owns a Sushi bar in subway station. He follows the same steps

to his commute from over half a century to get to his store. He loves making sushi and runs only the 3 star Michelin star

rated restaurant. He refuses to call the sushi he makes as perfect and says he has a long way to go. He also says his 6o

year old son is not yet ready to take over the business.  This is one great example of how a team and individual can

identify their shortcomings by following the same process over and over. Every time they repeat the process they will find

themselves finding better at doing it. A short continuous cycle which ends with shippable product will definitely make the

team better in doing sprints with time.

## Customer Centric

In LeSS one of the 10 principles includes a Customer Centric focus.  The first step is understanding who is the "customer".

This may not always be obvious.  During the IPBR the customer is clearly identified and agreed upon by the Product Owner.  This also allows the Team to understand who the customer is.

The goal of clearly understanding the customer is multi-faceted.  First, the product should provide value to the customer.  In addition understanding who the customer is allows the team to consider the customer needs in their product development.  Understanding the customer will assist the product owner in prioritizing the customer requirements.

Also as the teams scale there is a common agreement of the customer needs and overall product development goals that create the customer value.

## Whole Product Focus

Customers don't buy a part of the product, but the whole product. This seems obvious but it is important to remember. It leads to a couple of important guidelines:

- Parts of software that are not integrated into the whole product have no value yet.
- Teams who finished 'their part' are not done until it is integrated.
- Whenever there is a choice to optimize a team output or the whole product, we always chose the whole product.

In many large product development groups, getting everyone to focus on the whole product instead of their individual parts/tasks/specialization is one of the hardest parts of scaling Scrum.

Always keep a whole product focus, keep reminding everyone there is no value in separate parts or half-working parts.

This principles suggests the application of Systems Thinking. (1)

1. LeSS Large Scale ScrumWhole Product Focus - http://less.works/less/principles/whole-product-focus.html

## Queuing theory

Queuing theory is a concept in LeSS that helps address the bottlenecks in SDLC (Software Development Life Cycle) that are identified during sprint planning for small problems or during initial product backlog refinement for product wide bottlenecks. The ways in which this can be solved is by laying out all the steps needed to achieve definition of done, identifying larger issues and them down into smaller components or assigning more resources against them.

For example if a development team is blocked by some unknowns on the requirements definition side, they would be prevented from moving on to coding pending resolution of this blocker, this may take a significant amount of sprint time even days to archive. In the queuing theory you can assign developers, managers, and other specialized skills team members to help with unblocking the issue by breaking it down to smaller components and having each member tackle finding out the missing requirements. This is a simplified example of how larger issues can be broken down to achieve a higher velocity for the team to be able to complete tasks on time and at the same time keep all members productive. Other examples of queuing theory methods and solutions can be found on less.works

# Ceramonies

## IPBR

Initial Product Backlog Refinement

Enhancing or Developing a product requires collecting an inventory of features that are intended to satisfy client (internal or external) requests; strategic upgrades to the product to remain competitive and/or required regulatory requirements. There is 1 product backlog that contains all requests.

Prioritization of the product backlog requires involvement from the Product Owner (see Product Owner) and Feature Team(s) (see Feature Team) and the scrum master (see scrum master). Attendance is required. This prioritization process can take up to 5 days. The first 1-2 days the Product Owner is required, 3-5 days optional. An overview of each of the backlog items is provided by the Product Owner and conversations with the feature teams occurs until a common understanding of each of the features being requested is obtained. Prioritization is the responsibility of the Product Owner. A key component of the Product Backlog Review is to agree to a common definition of Done.

## Sprint Planning

Sprint Planning is a key function of determining what can be accomplished during a sprint. The team comes together to 1) working with product owner to define features in more detail and 2) breakdown the tasks and sequence and effort estimates to complete the feature that is included in the sprint.

The team capacity is determined which then can be used to understand if the sprint planning outcome can be accommodated within a single sprint.

The sprint is defined based on the outcome of sprint planning and the sprint backlog is created and the work begins.

## Sprint Review

Sprint Review is one of the key principles of the Large Scale Scrum (LeSS). Multiple Scrum teams work on different features during the sprint. These features are part of the single product backlog and are built towards common business objective- "Whole Product" hence it's important to review the outcome of all the teams together. The participants of this meeting are Teams, Product Owner and users/customers and stakeholders. They review what the teams have built during the sprint and discuss the value it's adding to the whole product along with any new ideas or possible changes. It’s about Inspect-Adapt rather than Inspect-Accept. It's not the meeting to find the blame rather it’s an opportunity to collaboratively discuss the product. One of the recommendations is to conduct this meeting on a diverge-converge meeting pattern. Each team shows and discusses what they have built during the sprint. Stakeholders heavily participate in that discussion and in the end Stakeholders summarize their feedback and come up with suggestions towards the benefit of the whole product. It’s a team exercise and every team will be able to see potential shippable products of each team and how they together contribute towards the whole product.

## Product Backlog Refinement

Product backlog refinement (PBR) is a critical step in a LeSS framework. The complete shippable product will require product backlog refinement meetings to have a completed backlog of all required features and their relative priority. The product refinement meeting can be at the overall product level where the product owner defines the key features at the high level and reviews all the features needed for overall product with the team. In this meeting the requirements needed for coding are not detailed out and only the over product is defined.

Once sprint planning is completed the product backlog refinement is done at the smaller level for the individual sprint the teams are working on and the details of individual stories are fully flushed out between the team, the product owner and SME as needed in order to complete the sprint in full. This is the key step to getting a sprint completed and definition of done achieved through knowing clear requirements and being able to achieve them.

This key step is critical to an agile project so that constant review and refinement can help to uncover the variables and unknowns in agile development. For more please review information on less.works website.

# Roles


## Product Owner

The Product Owner is responsible for maximizing return on investment (ROI) by identifying product features, translating these into a prioritized list, deciding which should be at the top of the list for the next Sprint, and continually re-prioritizing and refining the list. The Product Owner has profit and loss responsibility for the product, assuming it is a commercial product. In the case of an internal application, the Product Owner is not responsible for ROI in the sense of a commercial product (that will generate revenue), but they are still responsible for maximizing ROI in the sense of choosing – each Sprint – the highest-value items.

In some cases, the Product Owner and the customer are the same person; this is common for internal applications. In others, the customer might be millions of people with a variety of needs, in which case the Product Owner role is similar to the Product Manager or Product Marketing Manager position in many product organizations. However, the Product Owner is somewhat different than a traditional Product Manager because they actively and regularly interact with the Team, prioritize by working with all of the stakeholders and reviewing the results each Sprint, rather than delegating development decisions to a project manager. (1)

### Area Product Owner

An Area Product Owner (APO) specializes in a customer-centric area and acts as Product Owner in relation to the teams for that area (see Figure 1). An Area Product Owner does the same work as a Product Owner but with a more limited, yet still customer-centric, perspective.

*Product Owner Team*—The APOs and the PO together form a team—the Product Owner Team. This team makes product-wide prioritization decisions, but the PO always has the final decision. Also, scope and schedule decisions stay with the PO—he decides when to release what.

Teams are distributed over the requirement areas based on the PB priority. Areas are of different size in terms of effort and so they contain a different number of teams. Too-small areas are not a good idea because they result in many backlogs and many APOs. The overview is lost and teams develop low-value features. Rather, prefer a couple of small areas combined in one broader area—increasing the flexibility within this area. On the other hand, too-large areas are difficult for one APO. To strike a balance, consider a minimum of four and a

maximum of ten teams per area. (2)

### Why Area Product Owners?

The main reason for having Area Product Owners is to prevent the Product Owner from becoming overloaded. Without an Area Product Owner, the Product Owner overload would be because:

- the PO dealing with too many teams. With all the tasks the PO needs to do, it seems impossible for him to work with more than a couple of teams. How to solve this? One way is for teams to take over the clarification of Product Backlog Item (PBI) work by including subject matter experts on the team. An alternative is for someone to assist the PO with clarification work. He joins the Product Owner Team but does not make decisions related to prioritization. Using these techniques, one PO can work with up to five or ten teams. More than that will cause information overload for the PO and makes iteration planning difficult.
- the PB becoming too large Lean thinking promotes small batches and short cycles. We suggest that each team have at least four PBIs for each iteration that they can complete independently within that iteration. With 50 teams this leads to a PB with 200 PBIs just for one iteration. Prioritizing 200 PBIs per iteration is too much work for one PO.
- Teams working on the whole system—Feature teams are good, and so is learning new parts of the system. But too much learning without delivering value is not. This can happen when teams work on completely unfamiliar features. They have no opportunity to specialize and this affects teams' velocity. How to strike a balance?
-

1. Source: LeSS More With LeSS - http://less.works/less/scrum/roles.html
2. Source: LeSS More With LeSS - http://less.works/less/less-huge/area-product-owner.html

## ScrumMaster

The ScrumMaster is a full time role with the goal of keeping the focus of the feature team on the product goal and providing the transparency to the product owner of the team's progress. The ScrumMaster helps coach the Product owner on understanding how to manage and prioritize the Product Backlog. The ScrumMaster helps the team become the optimal feature team and progress through the stages of team forming. They are there to help remove any impediments from the team that may be prohibiting feature development. The ScrumMaster is also continuously thinking on how they can improve the overall organizational system to speed time to market and velocity of the teams. As the ScrumMaster they facilitate the scrum meetings which consist of the sprint planning meetings, daily scrum calls, sprint reviews, sprint retrospectives and the product backlog grooming meetings. The ScrumMaster will also chair the initial product backlog review meeting and help guide the product owner in describing the features of the product for the feature team and decomposing those features into estimable chunks of work.