Feb 2016

Scrum.org *Whitepapers*

Scale your product NOT your Scrum

Cesario Ramos | PST & Product Development Coach

Scaling Scrum & Agile has become a very popular topic over the last ten years. You can tell by the number of papers, talks, trainings and certification programs around it. Agile has become its own industry, as more and more enterprises want the benefits that Agile can promise.

Over the last twenty years we have been rather successful at getting Scrum to work with single Scrum Teams. Nowadays, we want to use Scrum with tens or even hundreds of teams, and an overly simplified approach to scaling might introduce big problems.

The problems with just scaling

Scaling is about increasing in size. I've been told that fire departments scale their operations depending on the severity of the fire. Depending on the scenario, they increase the size of the trucks that join the fight, the number of trucks, the number of people and the coordination and communication process as needed. This approach is what I call Copy–Paste scaling. You 'copy' the trucks and people needed and 'paste' them to form a larger group while adding extra processes for communication and coordination.

When you apply this approach to Scrum this means increasing capacity by copying and pasting Scrum Teams in your development group. You add Product Owners, Product Backlogs, Scrum Masters and Development Teams. To support and coordinate this growth, organizations typically augment with special roles such as 'feature owners'. They also add extra layers of coordination, e.g. 'release train management', extra processes e.g. 'integration test phases' and even additional artifacts e.g. 'value stream backlogs'. Unfortunately, this approach results in diminished team-customer collaboration leading teams to focus on delivering components in isolation, instead of an integrated, potentially shippable product. And now you are slowing down rather than speeding up.

The Copy-Paste approach to scaling gets you into trouble rather quickly because of three main reasons:

- Scrum and Agile are based on essential values, principles and practices and just adding more people does nothing about using those at scale.
- Developing software is creative work not production work. Therefore, adding people does not necessarily increase productivity.

- The ability of teams to independently produce valuable and bug-free product every Sprint is essential. Copy-Paste scaling does nothing about improving the required engineering practices.

An example at one of my customer sites.

One of my customers operates in the energy trading business and their development group is distributed across three sites. They initially started with a few teams, but quickly scaled up to thirteen teams over a couple of months due to market demands.

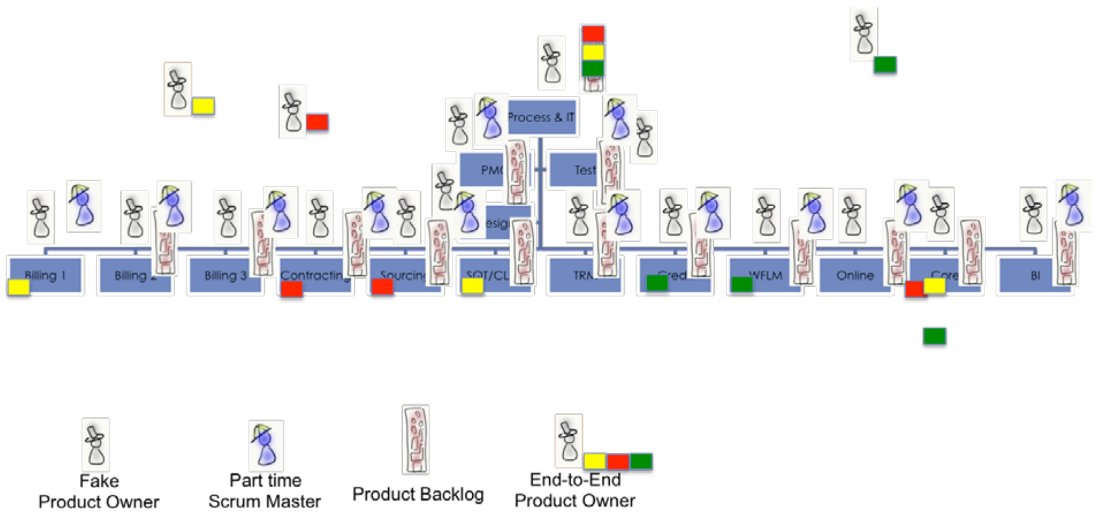

Figure 1: Organizational design result from Copy-Paste scaling

The development group supports a business process that consists of roughly thirteen steps. Naturally, following a copy-paste scaling approach, they formed thirteen Scrum Teams for those thirteen steps. Each team had its own fake Product Backlog, Scrum Master and fake Product Owner. Each team worked on a single step of the business process, even though a feature largely required multiple business process steps in order to deliver value to a customer. Therefore, the teams did not optimize for adding customer value but rather for producing lots of code. As a result, the teams and "Product Owners" needed extensive planning and coordination in order to align their work and deliver an integrated product. Additionally this model also introduced delay in testing and customer validation; hence more defects, information scatter of a feature across teams and opaque project progress. The result was low productivity, high defect rates and unhappy customers.

Instead of increasing capacity only, first focus on improving the engineering practices and the understanding of the Scrum principles and values.

Professional Scrum

Scaling a single Scrum team who are unable to build a releasable increment at the end of a Sprint will only make things worse. For example, if a single team already cannot complete testing within the

Sprint, then introducing multiple teams will give you an increase in untested product. The costs of testing later accumulate exponentially and therefore, it is necessary to improve your Scrum before you use it at scale.

Two axes for scaling

In Figure-2 you can see two axes for scaling. The horizontal axis is about structure & process. If you increase only along that axis, you will get more of what you currently have. If your use of Scrum, your focus on Agile values, or your engineering practices are troubling, they will give you much more trouble when you scale-up. You will end up with teams building more of the wrong stuff not only incorrectly, but also faster.

The vertical axis is for Learning; learning how to increase the Agile values, agile engineering practices, and the right architectures in your group so that the teams are able to reliably develop quality software every Sprint. Professional Scrum is also about learning how to increase your customer understanding, appropriate leadership and organizational design. If you increase only along the learning axis you will be able to build the right thing right, faster. But learning takes a lot of time and effort. If you want to learn while being able to ship product at the same time, find the balance between these axes.

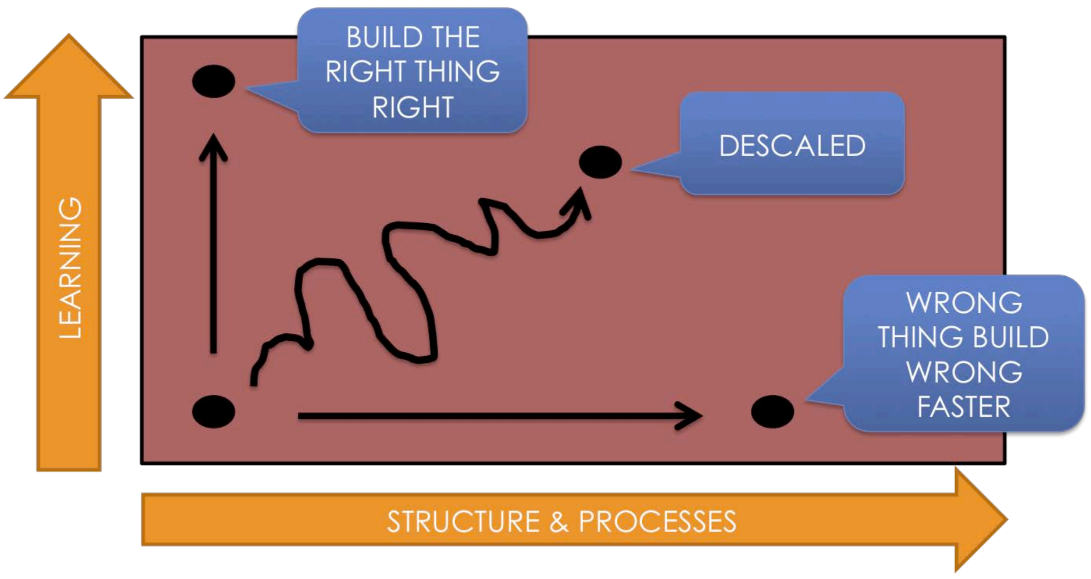

Figure 2: Scaling-Axes

Finding the balance

Instead of scaling your Scrum using the Copy-Paste approach you can use Scrum at scale to maximize learning while still shipping product. You start with just Scrum so that you will not add unnecessary roles and artifacts so your organization will be smaller then otherwise. If, however, your organization already initiated a Copy-Paste approach (similar to my customer example above) you will be able to remove superfluous roles, process and artifacts. Therefore, you will be able to descale your organization in order to move faster. Descaling is a key practice we discuss in Scaled Professional Scrum when multiple Scrum Teams working together to build one product are in trouble.

With the right balance, your teams learn and develop the right thing faster and you will need less capacity than otherwise. And that is what we did at my customer where we started learning because we scaled the product not just Scrum as I describe below.

Scale your Product

A highly performing Scrum Team independently delivers end-to-end features to customers every Sprint. The development of features flows from idea into the hands of the customer without delay. The team is self-managing and has excellent customer understanding. This situation is perfect.

We should expect the same at scale and thus design our Scrum at scale for perfection too. This scenario means that all teams can independently deliver valuable features to customers every Sprint. The teams do so without stepping on each other's toes.

But how do you remove these dependencies?

You remove these dependencies by scaling your product along customer domains. When you scale your product you will remove dependencies at both the feature level and the code level.

Removing dependencies at the feature level

With large products, there are generally a lot of users. These users typically get value by working in a single area of the product. When there are many such Value Areas in your product, often you find the necessary deep understanding of all those areas cannot be maintained within a single Scrum Team. Therefore, you should scale your Scrum organization along customer domains or Value Areas[0]. (The alternative is to organize teams around business process steps or architectural components but that will introduce a lot of waste as mentioned above.)

When you specialize your teams along Value Areas as seen from the customer perspective, the teams can focus on a subset of the customers. Therefore, each team needs to understand one customer domain only while still able to deliver complete features that the Product Owner can sell. Features in one Value Area are independent of features in another Value Area and therefore the teams working in those areas are independent of each other at the feature level.

Example at my customer

At my customer we redesigned the organization and identified the following Value Areas: I-Join, I-Pay, and We-Support – Figure 3. Each Value Area exists because they specifically address an area of customer value and requires specific detailed knowledge.

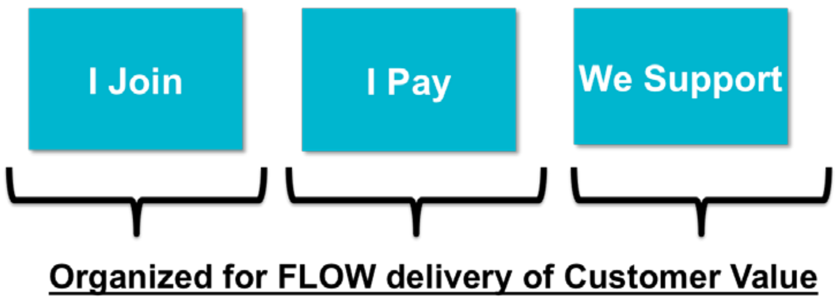

Figure 3: Value Areas

Each Value Area consists of four to six Scrum Teams working together in a single Sprint to add customer features to the overall product. We also have all Scrum events like Sprint Planning and Sprint Review per Value Area. All Value Areas work in the same cadence and produce an integrated product across all areas at the end of each Sprint.

Scaling the Product Owner

At my customer, each Value Area has an Area Product Owner and the Area Product Owners together with the Overall Product Owner form the Product Owner Team.

You can see the three Value Areas and their teams in Figure 4. On the left is the Product Owner Team. There is one Product Backlog that can be filtered into backlogs per Value Area. Each Value Area has an Area Product Owner and various Development Teams.

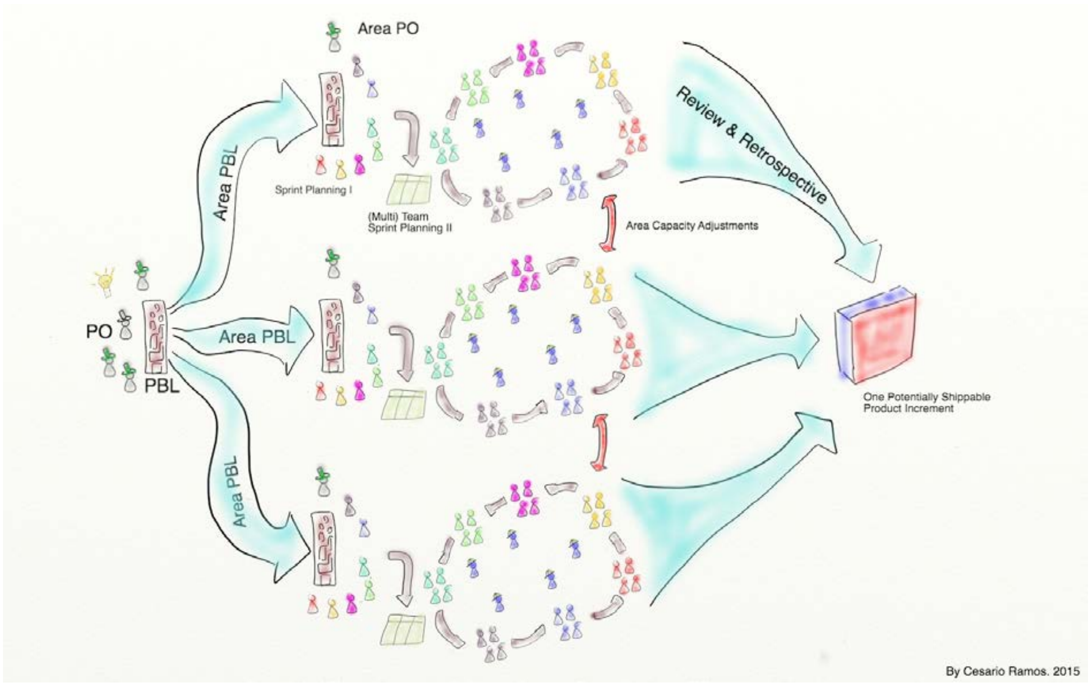
Area PO
Area PBL
Sprint Planning I
(Multi) Team Sprint Planning II
Area Capacity Adjustments
Review & Retrospective
PO
PBL
Area PBL
Area PBL
One Potentially Shippable Product Increment
By Cesario Ramos. 2015

Figure 4: Scrum at Scale with three Value Areas

The Value Areas have high independence at the requirement level because most of the requirements are separated by customer domain and that increases autonomy and productivity.

Organizing along Value Areas also leads to maximized learning. For example:

- The teams increase their customer understanding because they work closely with customers in one customer value area.
- The teams increase their product understanding because they are responsible for developing a complete working feature across components.
- The organization learns about real progress and product quality because all the work of all teams is integrated every Sprint and discussed with customers.

This organizational design significantly reduced the dependencies across teams. The result was higher productivity and happier employees. Customer focus increased people's intrinsic motivation.

A downside of this scenario is that we now had multiple teams in and across Value Area(s) working on the same code components instead of just one team. This introduced more code dependencies.

Removing dependencies at the code level

Working in shared components with multiple teams introduced more code conflicts and team dependencies at the code level. The good news is that we have Agile engineering practices encompassed in Nexus™ (a framework created by Scrum.org that drives to the heart of scaling: cross-team dependencies and integration issues) with Scaled Professional Scrum (SPS) to help. The use of test automation and continuous integration are essential to handle code dependencies within a Value Area. To handle cross Value Area code dependencies you also need an architecture that decouples the Value Area's at the code level.

Getting there

The common scaling challenges I encounter with my customers are around resolving dependencies at the requirements and code level. The top issues to tackle are dependencies between teams, requirements, domain experts, and especially code.

A good way to remove many of these cross team dependencies is to let the teams themselves address and resolve them. These dependencies do not need an additional group of people to fix them! When the Scrum Teams generate new ideas they take ownership of their process and are likely to continuously improve it.

Conclusion

When your product is successful and you need more teams to sustain its growth you should avoid Copy-Paste scaling. Adding teams that are specialized on single components or business process steps does not solve root causes and will only slow down development.

You'll increase your chances for success when your teams keep the focus on the customer. In doing so you need to find the balance between scaling your learning and scaling the structure & process in your group.

You find the balance by adding teams that specialize in the customer domain along a value area as seen from the customer's perspective. Your code level and requirement level dependencies will disappear and you will end up with totally independent Value Areas. As each area addresses a specific set of customer needs you can even promote Value Areas in order to separate products in the future.

Therefore you should Scale your product NOT just your Scrum.

References

- Value Area pattern – ScruPloP
- SPS - Scaled Professional Scrum - Scrum.org
- Emergent - Ramos C. 2014. EMERGENT - Lean-Agile adoption for an innovative workplace, CreateSpace publishing.
- Nexus - Nexus Guide – Scrum.org

About the author

Cesario Ramos

Cesario Ramos is an independent product development consultant and founder of AgiliX Agile Consulting, a small consulting organization that guides Agile improvements through coaching & training. He works as a management consultant on large and small scale Scrum adoption and as a Certified Coach, Professional Scrum trainer from Scrum.org and Certified LeSS Trainer. Cesario is also the author of EMERGENT – Lean & Agile adoption for an innovative workplace, a frequent conference speaker and happy to be an active member of the ScrumPlop® community.

You can contact Cesario at cesario@agilix.nl