Optimize your organizational structure for agility

WOLFGANG STEFFENS, kai kaku Oy

Organizations typically focus on improved collaboration within teams, yet they do necklet the potential of improved inter-team collaboration. For an organization "being agile" instead of "doing agile", it is crucial to apply organizational design thinking and systems modelling to optimize the whole organization for agility and provide the surrounding conditions for the teams' collaboration. Successful agile adoptions require that there is a real need for change, and that the senior management is willing to lead and support the change. Cross-functional, cross-component, autonomous, real teams are the key to building an agile organization.

1. INTRODUCTION

This report is based on my coaching experiences across 3 different companies from the insurance and automation industry. Company A (2017) was a department of about 250 people out of which were roughly 120 developers. Within a very large insurance company in Germany they developed vehicle insurance which was sold at car dealers [1]. Company B (2017-2019) was a large department across 5 international sites with roughly 450 developers in a division of a very large German company. They developed a platform, used by several other application units in the automation industry. Company C (2021) was a department of about 180 people within a large insurance company in Germany developing the online shop for insurances. Successful agile adoptions require that there is a real need for change, and that the senior management is willing to lead and support the change.

2. MY BACKGROUND

After working 10+ years as a program manager in Nokia Networks, I was exposed to Scrum for the first time in 2005. Willing to learn, I left my comfort zone and started to explore what it meant to "be agile" in a large organization. I learned from the best and became an Agile Coach/Scrum Master helping individuals and organizations within and outside of Nokia (Siemens) Networks. I gave many different training classes around agility like Product Owner, Agile Estimation & Planning, Problem Solving workshops, and Agile introduction workshops to support functions (like Human Resources). After 20 years, I quit and traveled 3+ years around the world on my motorcycle with my dogs learning a lot of soft skills. Back in Europe in 2016, I was able to continue what I love doing—coaching, supporting and training organizations with the aim to "be agile".

3. BACK TO COACHING AND THE FALLACY FROM THE ONE-TEAM SCRUM PILOT

The first coaching assignment after my journey was company A in 2016/2017, a department organized as a program, led by a program manager. About 6 months before I joined them, the organization "piloted" Scrum to see whether Scrum might work for them. The department leadership selected a team which worked on an isolated product or service. This team did not have dependencies on other teams, and as a result it did not need to interact much, if at all, with other teams, and had limited interaction with the rest of the organization. Here, the one-team Scrum approach worked well and brought positive results due to the increased collaboration within the team. This pilot demonstrated improved productivity and even some kind of adaptiveness when it came to responding to changing requirements, which is the main purpose of "Agile".

All-the program manager, senior managers, and first level managers believed that by copy-pasting Scrum, a.k.a. Multi Scrum-team, to all teams, they could repeat the same positive results for the entire organization consisting of many teams and hundreds of developers. These managers decided upon team scope, i.e., which team works on which component, system-component, or part of the system, or the product, or on the architecture.

Wolfgang Steffens, kai kaku Oy; email: wolfgang@kaikaku.fi

The organization moved from a functional setup with single-function teams, team leads, and business leads, to this multi Scrum-team set-up with "cross-functional" teams, one "Product Owner" per team, and team backlogs. The new teams were almost cross-functional, but not cross-component, nor autonomous. Because a customer feature needs changes to be done in several or even all of the components, this set-up with still specialized teams caused significant dependencies between teams as we will see.

This was the state of the organization when I joined. On any coaching assignment my first question is "what is your product?" Here, managers as well as architects answered by sketching and explaining technical bits and pieces of the product, basically nothing more than components of the overall architecture—a technical view.

The teams were almost fully cross-functional, except architects still formed their own team, and much of the testing capacity was outsourced to a company in India. These teams could do work within their system component but not at product level (in fact they were component teams). There were some minor dependencies to the host system but changes to this complicated sub-system seldom occurred and could easily be coordinated.

As the organization was drowning in dependencies between these component teams, the coordination effort between the teams was tremendous, feedback cycles (from testing) were very long, and most importantly, the customer features did not get done. Changing direction was hard and cumbersome. There I was, observing an organization "doing agile" instead of "being agile".

Why are multi Scrum-team set-up with cross-functional, not cross-component, not autonomous teams such a big problem?
Imagine a web-shop like a retailer, or an online insurance shop. Very often there is a division of work, a specialization, into Front-End, Back-End, UX, and database. A typical customer functionality or requirement needs changes to be done in several or even all of the areas. In fact, the customer typically does not care whether the problem is solved in the Front-End or the Back-End. These dependencies require a significant amount of coordination, resulting in a coordination nightmare with customer functionality hardly getting done, leading to very long cycle times. Please note that the same applies for most of the organizations which use different terms like "microservices", "squads & tribes", or similar.

After 3, two-week long Sprints, it looked like not a single customer feature would get done. This was a disaster for the program manager and, as a consequence, he pulled the emergency switch in the middle of the release, created ad-hoc a single backlog for the entire organization, and threw the entire organization into a task-force mode. Now I got the attention to start my coaching concerning the organization's structure. Through daily conversation with the program manager, we started to build a trusted relationship.

KeyLearnings: (1) People in product development like managers, architects, analysts, or developers, often lack a customer-centric product view; instead, they focus on the technical parts. (2) "In offering one's opinion, one must first ascertain whether or not the recipient is in the right frame of mind to receive counsel."—Tsunetomo Yamamoto (1659-1716)

3.1 The Agile Guiding Coalition

When I joined the company, there already existed an Agile Guiding Coalition [2][3]. I do not know how it was formed, yet it seemed to have followed Kotter's advice regarding its composition. Its purpose was to guide and lead the organization through the change process—a long-lasting operational change support group. The group consisted of the program manager, four 2nd level managers, the chief architect, a representative from the Team Product Owner (ex-team leads), a representative from the Scrum Masters, and the two Agile Coaches (including me). We had every day after lunch a 30min public stand-up in the coffee corner.

This Agile Guiding Coalition was crucial for the change in the organization. In this active meeting, we discussed problems, experiments, future directions, pros & cons of ideas, and made decisions together. We learned from each other, and this frequent stand-up demonstrated a high level of commitment from those managers to the change.

My role was twofold. On the one hand, I made the members aware of the "agile mindset" while we had conversations and reviewed solution proposals. On the other hand, I brought in observations, made others aware of potential problems, and provided ideas for experiments. In this way, I was sometimes acting as a coach, sometimes more as a consultant.

4. OPTIMIZING AN ORGANIZATION FOR AGILITY

The interesting part was that the AGC members did not want to follow any existing frameworks like LeSS. Instead, we talked about what did work well, what did not work well, what required improvement, and what we could do about it (see Table 1). The typical release cycle was three months (6 Sprints). In the AGC we planned that the best time for those major structural changes would be between two releases.

Action	Order of discussion	Order of happening
Create one Product Backlog through an initial Product Backlog Refinement workshop	3	1
Create cross-functional, cross-component Feature Teams through a self-designing team workshop	2	2
Create a Product Definition and Product Areas	1	3

Table 1 Flow of events

4.1 One Product Backlog

As a result of the emergency switch pulled by the program manager, we saw that one Product Backlog with customer-centric Product Backlog Items brought focus in the entire organization. Instead of creating this in the middle of the release, it needed to be created before the release, and by all team members in collaboration with the stakeholders. The foundation for a 1,5day initial Product Backlog Refinement workshop was agreed upon. The preparation for this workshop was minimal. I trusted in the collaboration of developers, team Product Owners (i.e. analysts), architects, and stakeholders to clarify the requirements once the problem was articulated and the release goals known. Give people the boundaries, time, and space, provide rudimentary tools (pen and paper as well as white boards), and magic will happen. The biggest challenge was to educate people to think in terms of customer-centric Product backlog Items instead of technically sliced items and tasks.

4.2 Create cross-functional, cross-component Feature Teams

Another major learning from the initial messy set-up with component teams was that this team structure was not “fit for purpose”, so we needed to reorganize the teams. This was a harder nut to crack, because now management and other managers from the AGC, needed to admit that their team design was not working, resulting in a fear of losing face. After longish conversations, we found a way out by calling the first team structure an experiment, which did not produce the desired results, and now we knew what to do better. This view was helpful and we could move forward with a self-designing team workshop.

The workshop was held on a Monday afternoon with the goal that the team members decide by themselves with whom, and in which part of the product they wanted to work. This was a big event, involving 100+ people, not recommended by the text books, and yet we succeeded in establishing a new organization with cross-functional, cross-component teams which had the required skills and competencies to convert a customer-centric Product Backlog Item into a “done” product increment. I had some additional support from other experienced coaches, all the Scrum Masters, the architects, and the newly established Area Product Owners (see next chapter). Management was thrown out during this workshop to avoid them pushing and influencing what they thought was best. The event was a blast. It boosted the motivation of the team members beyond imagination, at a level managers only dreamed of achieving earlier.

4.3 The Product Definition

Before this team self-designing workshop, we needed to find an answer to the definition that was missing: “what is the product?” The Product Definition determines what organizational elements (people, components, processes, and systems) are needed to develop and run the product [4], e.g., insurance, banking, web-shop. In this rather typical organization with a strong technical view, this proved to be a challenging task.

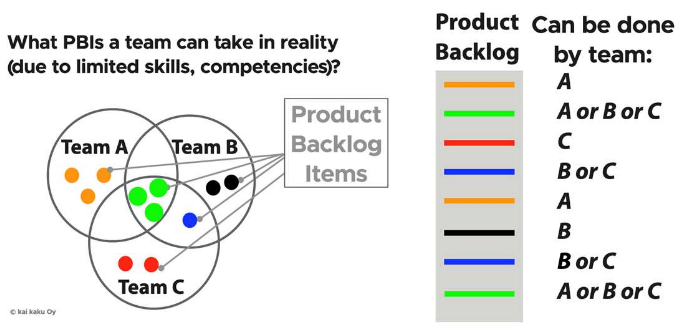
I paired up with the internal Agile Coach as much as possible. We sat in the same room and had a lot of discussions, learning happened, and pretty quickly he knew what we were up for and what to aim for. It took the internal Agile coach 2-3 weeks of intense conversations with many developers, architects, and others to figure out the product definition by analyzing different workflows and stakeholders. Due to the number of teams, and the cognitive capabilities of the one Product Owner to handle the work, it became obvious that we needed to sacrifice adaptiveness somewhat (see Box below) and split the product into the following three customer-centric Product Areas:

- Contract underwriting (5 teams): Selling vehicle insurance for a newly bought vehicle through the vehicle dealer. The technically challenging part was the integration of all the IT systems between the car dealer and the insurance company.
- Contract modifications (3 teams) had to be made so that the insurance holder could change the contract or personal data.
- Claims (2 teams) so the insurance holder could file a claim report in case of damage.

Note: The number of teams per product area was a result of the self-designing team workshop and not known beforehand.

The dilemma of Product Backlog items only known by one team / product area.

The fact that a team (or a product area) can only work on some of the Product Backlog Items will result in a sub-optimization and thus in reduced adaptiveness. The Figure below illustrates this. As you can see, due to specialization, limited skills, or silo competencies, e.g., only team B can work on the black items, team A only on the orange ones. Other teams (Product Areas) do not know the requirements, nor do they have the design and implementation skills. If the Product Owner now decides that the orange items have the highest priority, it means the capability for the organization to shift direction is limited due to this knowledge silos. An organization that is optimized for adaptiveness has a large intersection area and smaller disjoint areas.



Each of the Product Areas needed to have a business-responsible Area Product Owner to work together with the teams, maximizing the Return Of Investment for this Product Area, in short acting like a Product Owner yet with a narrower product scope. We needed to find volunteers to fill the role. The discussions and selection happened in the background without my presence. It was clear to everybody what the goals were: to select three Area Product Owners out of the many Team Product Owners. It is important to note that the remaining Team Product Owners went happily back to working as developers (doing analysis and coding).

Here is a summary of what happened: Two weeks preparation for the Product Definition, the self-designing team workshop, and the initial Product Backlog Refinement workshop as well as selecting the Area Product Owners, and then:

- Monday afternoon: New organization with 100+ people was formed
- Tuesday and Wednesday morning: Initial Product Backlog Refinement Workshop
- Thursday: Sprint Planning

Other meaningful improvements were also made by (this is not a complete list):

- Conducting Multi-Team Product Backlog Refinements
- Having Sprint Planning 1 with all teams within a Product Area
- Having a Sprint Review with all teams
- Establishing regular retrospectives across the entire product
- Building a robust Continuous Integration System
- Continuing to provide an "Improvement Service" for the organization
- Getting the Product Owner Team (Product Owner & Area Product Owners) working

The result of these structural changes was phenomenal. The collaboration between the teams increased, as well as the collaboration with stakeholders and product management. The product quality improved significantly due to shorter feedback cycles and improved inter-team collaboration. Furthermore, through common Product Backlog Refinements held by teams together with stakeholders, and relentless inspect & adapt, the product did what the stakeholders needed. The flood of complaints (bugs) from the users when the release went to production was eliminated, besides a general improvement in product quality.

The organization started "being agile" and I was proud of myself having witnessed the impossible become possible—supporting this fundamental change in this organization. Yet, there was plenty of room for continuous improvement.

Some political games were played in the background as one of the four 2nd level managers did not like me, my doings, or what we achieved. I am not aware of what all happened nor can I recall why my connection to the program manager faded. Maybe I was too busy with the continuous improvement in various areas, maybe the managers got afraid of their own success. Anyway, after only 6 months, I got the info that my services were no longer needed and another coach should replace me.

I admit that at that point in the past, I did not pay too much attention to this, and reflected little about it, since I had different plans for my life. Interestingly, six months later, the program manager and one of the managers asked me to re-join the organization as a coach, but I was already busy with the next assignment.

KeyLearnings: (1) You need to have your toolbox with you, instead of coming with ready-made frameworks. (2) If something works well, try to do more of it. (3) Enjoy the improvements. (4) Self-designing team workshops are fun and they typically produce great results. (5) Pay attention to maintain a good connection to the person who hired you, and to key leaders in the organization.

5. ANOTHER ROUND OF MULTI SCRUM-TEAM

Anyway, plans do change and after this overall extremely motivational yet short experience, I started my next coaching assignment a few months later with a high level of energy and enthusiasm at company B. My entrance into this coaching assignment started by providing a two-day Scrum Master training to the newly appointed Scrum Master, the chief architect, and program managers. Interestingly, the topic of "how do we make Scrum work in our large organization" was kind of taboo. The ones who ordered the class did not want to start the "framework discussion", so the topic was left unanswered.

Here, the idea of creating a new organization through a self-designing team workshop, but per location, was already decided by management. I was involved in these workshops at two different sites within a few days apart. Unfortunately, because having these workshops by site meant the site dimension was stronger than the product dimension.

However, once more the question "what is your product?" was not answered from a customer's or user's point of view, but from a technical one instead. This resulted in the composition of cross-functional component teams per site. The huge problem with component teams, or system-component teams, is that they cause significant dependencies between teams as we have already seen in company A.

In company B, those cross-functional teams included the role of explicit team architects. This is a violation of the Scrum rule where we should have only role-free (in the sense of job-title and specialization) team members. Scrum calls them all developers. This led in some cases to a hierarchy within the team and other dysfunctional behavior. Furthermore, due to their limited scope of a "platform" the teams could not perform end-to-end testing within their teams.

The team forming itself happened similar to company A within the first weeks of my coaching assignment. Based on previous experiences, I tried to do as little as possible on my own. Instead I supported and coached the Scrum Masters and the "Transformation coach". We avoided major mistakes in the workshop moderation, and by making tiny ones, they all learned a lot.

After the workshops, I suspected that this organization would have another self-designing team workshop after a few months, once they learned that this set-up is not working well. I was very wrong. Over many years, the people in the organization got accustomed to suffering. So they suffered, and instead of changing the organization they put huge efforts in managing those dependencies and coordinating between teams. My problem started to be that since I had witnessed what an organization can achieve with the same people by structuring the work differently, I felt eager to do the same here again.

My focus over the next year was implementing Scrum structure together with the Scrum Masters, improving the collaboration within the teams, as well as coaching the Scrum Masters, management, support functions, and the "team Product Owners". After being there for about a year, the further improvement at team level was minimal and it was time to do more structural changes so that the collaboration between the teams would improve, the organization as a whole could improve, and become more adaptive.

That was about the time when I lost the connection to management. Since intra-team collaboration improved, I got the feeling they were happy with that level of improvement, they had seen exactly that before, and so they settled for it. Several attempts to improve the situation by clarifying what the organization defines as a product got rejected or did not lead anywhere. The mental step and the required changes in the power structure towards this direction obviously was too big. It was like "We worked for 20+ years like this and this is how we work—agile or not makes no difference!" [5]

Over the next 6-9 months, I focused on coaching the teams in the support functions. Again, here we improved the intra-team collaboration, and realized that e.g. working in a pull-mode with a Kanban board seemed more appropriate compared to the Scrum framework.

The organization had hired a total of 3 coaches, and besides our best attempts we were not able to break down barriers. One coach was dedicated to work only with management, the other coach and me, we were supposed to stay with the teams. Working across boundaries was not welcome and critical access to other needed information was often denied. So all of us tried our best individually like a loosely connected group without common goals. A big difference compared to company A.

In the last few months, I was branded as a religious person, a theorist who has no idea how things work in real life [6]. I helped the many teams as much as possible, yet was not able to improve the collaboration between teams, nor between teams and stakeholders. I felt devastated because all I wanted to do was to help but was denied the opportunity to do so.

LeSS [7] suggests a guide called "organizational perfection vision". Without having this vision, continuous improvement towards perfection is almost impossible, ending in the situation where people do what they can and/or want to improve instead of improving what needs to be improved from an overall systems view.

KeyLearnings: (1) The concept of a Product Definition and its importance is hard for the typical technically-minded person to comprehend. (2) It is absolutely necessary to have a close regular conversation with all the involved coaches, the sponsor, and the department lead (if they are not the sponsor). (3) Realize the potential limits in supporting the change in an organization which you have as an external agile coach. (4) Clarify and visualize with your sponsor what the sponsor really wants, what is the target state of the organization, and why you were hired in the first place. Managers might say they want to be "agile" and yet most managers do not know what it means when it comes to organizational design. In fact, they do not know that organizational design is one of the key elements to becoming an adaptive organization (5) "Nothing changes, if nothing changes"—probe your client's willingness to change.

6. ONE MORE ROUND OF MULTI SCRUM-TEAM

My next assignment brought me again to an insurance company in Germany. Due to Corona, all coaching was done remotely. I was delighted to see that the teams could do many end-to-end customer-centric features by themselves and the coaching sponsor was enthusiastic about the further need to change towards "being agile". What a great start. This time, I had another like-minded colleague with me and both of us hoped for a serious change in this organization.

The situation in company C seemed a little better since they had established full-stack teams with analysts/Subject-matter-experts, Front-End, and Back-End developers. Those members had sufficient testing knowledge to cover unit and acceptance tests. We quickly noted that besides the need to improve collaboration within teams, there were still significant inter-team dependencies on Ux design, deployment services, and host. Again, the coordination of those dependencies between teams were mind-blowingly energy-consuming.

The product "insurance" was divided into 3 customer-centric product areas: contract underwriting, landing page, claims, and one platform group. We focused our coaching efforts on the product area contract underwriting. The company had intelligent and motivated people with the right skills and competencies to be real business-responsible Area Product Owners, yet the political games in the company killed all meaningful attempts to establish this role. Instead they introduced an additional role of "technical Area PO" for each area.

Now that these "technical Area POs" did a lot of analysis, documentation, and coordination, they became the bottleneck and a new layer was introduced—the Agile Manager (or in some cases called "content team-PO"). Now, each team had the dysfunctional set-up of a "team specific PO" (like a team lead) doing all the detailed analysis and coordination on behalf of the team, spoon-feeding the team with requirements (Note for Systems Thinkers: A vicious reinforcing loop appears if one does a systems model for this case).

Due to the strong hierarchies in this organization and the fact that, from an "Agile" point of view, flawed promises were made when hiring people to the position of the Agile Manager, none of those Agile Managers were willing to join the teams doing analysis work as a regular team member (like we saw in company A). Agile Managers were somewhere between the Scrum roles Scrum Master and "Product Owner", a completely dysfunctional set-up [8].

Anyway, those Agile Managers were not new in the organization and they had their connections within the rest of the company. As a result they hired Scrum Masters for their team (making the situation even worse due to the manifestation of their role), leading to a lose-lose situation for both of us coaches.

Instead of making more futile attempts to improve the Product Owner structure, after about a month we started to tackle one of the structural elements: shifting line management to being less focused on the technical side (e.g. one line manager for Front-End only) and more focused on the product area (and thus customer dimension). In a regular line manager's meeting, we, the coaches, explained our observations and the consequences in people's behavior in both of the scenarios. We proposed this experiment and also provided guidelines about the implementation. Those guidelines were ignored, the change was not done as an experiment, but directly as a full implementation. The role was changed at all product areas at once whereas the other two product areas did not receive coaching. LeSS suggests as the first adoption principle: "deep and narrow over broad and shallow" [9], so that coaching effort can be focus and real change will happen.

During this management meeting it became clear that this proposal fired backwards. The department's head "1. Officer" (our sponsor) was all in favor, and it seemed the rest were against it. Nevertheless, immediately after the change was implemented the "technical Product Owner" saw the positive effects of the change and yet the change was officially opposed by those managers and other "technical Area Product Owners". LeSS promotes the concept of owning a process (or a solution) instead of renting it. The ideas were "rented" from us coaches, not owned by the managers. Retrospectively, we think that the biggest problem was that the department leader delegated change activities instead of leading them, so we coaches got pulled into the political game as we experienced firsthand in the previous experiment.

Working with what we had in hand, focusing on what we could focus on, we had a suggestion for one more experiment a few weeks later. In contract underwriting, the area had 3 mostly independent teams (vehicle, life, household insurance), working on their three mostly independent backlogs (see the box "The dilemma of Product Backlog items only known by one team / product area."). This was leading to the situation that the most important work seemed to miss the deadline whereas other teams worked on low-value, uncritical items. Since the team members had the competencies to work in all of these three fields, we coaches, after explaining why, suggested another experiment changing the way of working so that the 3 teams could work on one common backlog (see company A). Again, this was understood and accepted by senior management, yet it would mean that the "Agile managers" would need to change their role. LeSS offers the guide: "job safety but no role safety" [10].

Basically this was the moment where we coaches were rejected. Additionally, the friendly-to-us "technical Area Product Owner" got badly sick, so he was replaced by another person with whom we did not have a good start. The tide turned against us, we were not welcome anymore in this product area. My assignment ended after only three months with improved team collaboration in some of the teams, yet no meaningful improvement on the organizational level.

KeyLearnings: (1) Apply the learnings from previous engagements to the extent possible. The other coach and I had regular sessions together, and sometimes with other external Scrum Masters and coaches. (2) For any successful adoption, all parties need to find the time for meaningful conversations. We wanted to have regular conversations with our sponsor and department lead, which were denied to us with the explanation that the lead was too busy. Our meetings happened sporadically and were short. (3) Real change appears hard and most people/senior managers/leaders are not willing to change their own thinking, and avoid taking the risks associated with the change.

7. ORGANIZATIONAL DESIGN—THE MISSING SKILL IN ORGANIZATIONS, MANAGERS AND LEADERS?

In all the cases, I have experienced ignorance and lack of knowledge when it came to the basic understanding of organizational design. Management made important decisions on the structures of organizations typically resulting in dysfunctional set-ups which were not optimized for Agility.

None of the managers that I met in those companies had systems modeling skills. Thus, they did not have tools to understand the dynamics in their organization caused by all the many interactions between people, teams, and processes. None of the managers took the time to learn about those tools. Instead, managers copy-pasted models from other companies, previous departments, or naively thought that copying one-team Scrum to multi Scrum-team would work. They lacked the basic understanding that their environment represented a complex system, that they needed to run their own experiments, and needed to find out through Inspect and Adapt what works in their organization and what does not. On the other hand, those managers and leaders are often too busy to take time to learn these fundamentals.

In the meantime, I have been focusing on providing training classes, organizing meet-ups, going to conferences, making people aware of and teaching organizational design to managers, Scrum Masters, “product Owners”, team leads, transformation coaches, and many others in the hope they start to influence from the inside of their organizations. Occasionally I have those busy managers in my classes, and then I have seen real change in their organizations.

The path to agility is hard, painful, and long. I will practice more patience in working with leaders in my upcoming assignments. Further, I will try to understand better what the different people really want, why they want a change, whether they want to lead the change, and most importantly whether they are willing to start changing themselves.

If there are any managers reading this report, I urge you to take the time to learn more about organizational design and systems modeling. It will help you to enter into a new world.

KeyLearnings: (1) Make leaders aware and involve external coaches before you decide to start your journey towards an agile organization. (2) The most promising way out of this chaos is to compose teams in such a way that asynchronous dependencies between teams are minimized. As a result, “dependency meetings” with Scrum Master and/or “team POs” can be eliminated. The remaining inter-team coordination is the teams’ responsibility. (3) Structural changes in team composition that minimize dependencies between teams, and the related change of roles and responsibilities, seems to be one of the most difficult changes in organization. (4) I think managers might be afraid of these structural changes because they do not understand the underlying nature of their current system, these changes are far beyond their comfort zone, and the risk that something goes wrong in this change appears too high to them. (5) Looking at the current job offerings and descriptions, Scrum Masters or Agile Coaches are limited to focusing on a team only from the beginning. Furthermore, there exists an illusion that one can hire an expert on the field of large-scale organizational design for the salary of a newbie once the organization is willing to change. The consequence might be the change attempt does not meet expectations because of the inexperienced Scrum Master and Coaches, thus reinforcing the thinking that the change was too big and too risky. (6) Leaders, please learn about organizational design and what it means to make Scrum work in your organization. (7) Leaders, please take the time to learn (do not expect to get a driving license after one lesson). (8) As a coach, learn to let go—you cannot save the world alone

8. CONCLUSION

In general, leaders in organizations did not seem to be aware of organizational design and lacked systems thinking skills. As a consequence, the organizations fell short on getting most out of introducing “agile” methods like Scrum. The collaboration within a single team improved, but collaboration across teams working on the same product was typically neglected. There lies a huge potential upside. Every case is different and yet certain patterns appear. Looking back, I realized how important it is to align expectations between the coach and the management in the beginning, clarify the assignment, and gently probe the degree of which the organization is willing to change.

9. ACKNOWLEDGEMENTS

I thank Rebecca Wirfs-Brock for seeing the potential of this experience report in the first place, and especially I thank her for the reviews, comments, and suggestions in creating this report. Thanks, Rebecca, I could have not written this without you.

REFERENCES

[1] Steffens, Wolfgang. German Big Insurance [Online] 2007 https://less.works/case-studies/german-big-insurance.
[2] Kotter, John P. Leading Change. [Chapter 4]. Harvard Business Press, 1996.
[3] LeSS. Improvement Service [Online]. https://less.works/less/management/improvement-service.
[4] Cesario, Ramos. Product Definition in Large Scale Scrum. [Online]. https://less.works/blog/2020/07/07/product-definition-in-less-1.html
[5] Larman, Craig. 1st Larman’s law of organization behavior. [Online]. https://www.craiglarman.com/wiki/index.php?title=Larman%27s_Laws_of_Organizational_Behavior
[6] Larman, Craig. 3rd Larman’s law of organization behavior. [Online]. https://www.craiglarman.com/wiki/index.php?title=Larman%27s_Laws_of_Organizational_Behavior
[7] Larman, Craig and Vodde, Bas. Large Scale Scrum. [Guide: Organizational Perfection Vision]. Addison-Wesley, 2016.
[8] Steffens, Wolfgang. The harmfulness of Scrum Master double roles and what you can do about it [Online]. https://www.kaikaku.fi/the-harmfulness-of-scrum-master-double-roles-and-what-you-can-do-about-it/
[9] Larman, Craig and Vodde, Bas. Large Scale Scrum. [Guide: Three Adoption Principles]. Addison-Wesley, 2016.
[10] Larman, Craig and Vodde, Bas. Large Scale Scrum. [Guide: Job Safety but not Role Safety]. Addison-Wesley, 2016.