Evaluation of Scrum-based Agile Scaling Models for Causes of Scalability Challenges

Necmettin Ozkan[1][a] and Ayca Kolukisa Tarhan[2][b]
[1]*Information Technologies Research and Development Center, Kuveyt Turk Participation Bank, Kocaeli, Turkey*
[2]*Computer Engineering Department, Hacettepe University, Ankara, Turkey*

Keywords: Agile, Software, Scrum, Scale, Model, Framework.

Abstract: Agile Software Development (ASD) community have come up with the idea of scaling and created some models/frameworks for large-scale set-up. Despite these models, application of agile methods to large projects remains as a challenging research question for the community. The aim of this work is to evaluate these Scrum-based scaling models (including SoS, Nexus, LeSS, SAFe, Scrum at Scale, DAD, and RAGE) in terms of how and to what extent they are able to provide scaling solutions for the identified causes of challenges in the Agile Manifesto and Scrum Guide, in the context of software development and project management. The study maps the solution proposals of the models for underlying causes of challenges and justify them with three experts. To come up with an evaluation, considering the experts' views, the auditors provide a thorough perspective on models' solutions against to the common pain points underlying the manifesto and guide for scaling. With some exceptions, we see that all of the models try to maintain the pain points associated with scalability in the core of the ASD.

1 INTRODUCTION

Agile Software Development (ASD) methods were initially designed and work well for small, co-located development team settings (Dingsøyr & Moe, 2013; Costa at al., 2014; Abrahamsson et al., 2003; Hoda et al., 2010) and there exist claims asserting the ASD inherently bears inability to scale (Moore & Spens, 2008). In the case of the ASD, experts believe that large-scale projects are quite different from small-scale projects regarding issues and challenges that they face (Dingsøyr & Moe, 2013; Rolland et al., 2016). Despite the challenges, the charm of being agile attracts many medium- and even large-sized organizations (Dingsøyr & Moe, 2014) and more energy is being put into scaling the ASD across enterprises (Versionone, 2017). In response to this need, the ASD community have come up with the idea of scaling and created some models/frameworks (hereinafter referred to as the "model") for large-scale set-up. Well-known models include SoS (Sutherland, 2001; Schwaber, 2004), Nexus (Schwaber, 2015), LeSS (Larman & Vodde, 2016), SAFe (Leffingwell, 2011), Spotify (Kniberg & Ivarsson, (2012), Scrum at Scale (Sutherland, 2019), DAD (Ambler & Lines, 2012), DSDM (DSDM Consortium, 2014) and RAGE (Thompson, 2016), as listed on "agilescaling.org".

The existence of these models does not guarantee widespread usage or a full spectrum of solutions for scaling, and the application of agile methods to large projects remains as a challenging research question for the community (Dingsøyr & Moe, 2013; Moore & Spens, 2008). The academic studies providing evidence on scaling the ASD are still scarce (Paasivaara & Lassenius, 2014; Meyer, 2014) and it is challenging to assess whether the scaling models have indeed properly addressed the issues related to the scaling (Hobbs & Petit, 2017). In particular, the number of studies dealing with what kind of causes of challenges there might be pertaining to the core of the ASD (referring to what the Agile Manifesto and Scrum Guide propose) and how the models overcome the relevant issues is lacking.

The literature on scaling the ASD has been dominated by consultants (Paasivaara and Lassenius

[a] https://orcid.org/0000-0001-9876-8728
[b] https://orcid.org/0000-0003-1466-9605

Ozkan, N. and Tarhan, A.
Evaluation of Scrum-based Agile Scaling Models for Causes of Scalability Challenges.
DOI: 10.5220/0009390403650373
In *Proceedings of the 15th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2020)*, pages 365-373
ISBN: 978-989-758-421-3

2014; Hobbs & Petit, 2017) and they may easily evangelize the models with commercial concerns and it may cause polished success stories and a bias in evaluation. This calls for a proper independent evaluation of the subject. The current literature on scaling also blend issues those coming from the Agile Manifesto and Scrum Guide (called as the core of the ASD throughout this study) and implementation-specific ones. This makes difficult to distinguish the source of challenges; whether the issue is with how the Agile Manifesto and Scrum Guide are designed or it is merely implementation-dependent. This study aims to become free from implementation-dependent and context-based challenges by elucidating the factors that directly affect scalability from design of the Manifesto and Scrum Guide. This implies the Agile Manifesto and/or Scrum Guide does not provide absolute agility, or they are agile at best case for small scale or a particular context yet not suitable for the large-scale agility. Thus, our study thus opens gates to the questioning of the underpinning "sacrosanct" Agile and Scrum principles for the sake of scaling.

In order to identify the fundamental and underlying pain points in the core of the ASD inhabiting the scalability, Ozkan and Tarhan (2019a) is based on. In that study, the following topics of causes of challenges related to scalability were determined: physical dependencies, fragmentation, short and static events, narrow focus on product, narrow focus on construction, bottlenecks from one to many relations.

In term of the models, this study focuses only on Scrum based models, because of its share of use in the sector (Versionone, 2017). In parallel, a remarkable number of the aforementioned models is built on Scrum and/or supports Scrum. Thus, while considering agile software development in general terms through values and principles of the manifesto, Scrum is preferred to study in particular in this paper. From this point of view, Spotify that does not specifically follow Scrum or other common frameworks and DSDM that proposes a generic approach to agile project management are kept out of scope in the study. The remaining models (SoS, Nexus, LeSS, SAFe, Scrum at Scale, DAD, and RAGE) are all compatible with Scrum and therefore included in this study. The research question then was identified as following:

RQ: How and to what extent do Scrum-based Agile scaling models address the causes of the challenges related to the core in the context of software development and project management?

The remaining of this paper is organized as follows: Section 2 elaborates related work. Section 3 evaluates the models regarding how and to what extent they resolve the identified causes of challenges. In doing so, the study maps the solution proposals of the models and justify them with three experts for underlying causes of challenges. In Section 4, with consideration of the experts' views, the auditors provide a thorough discussion on models' solutions against to the common pain points underlying the manifesto and Scrum Guide for scaling. Finally, in Section 5, we present conclusions and future work.

2 RELATED WORKS

Vaidya (2014) reviews three scaling agile frameworks (SAFe, LeSS and DAD) from the perspective of their approaches to roles, processes, and other salient features. Uludağ et al. (2017) provides primary information about the methodologists, such as who invented and published the models, the organizations that were built upon them, and the scaling, agile approaches involved, for twenty scaling agile frameworks including ones in this study. It also provides a maturity assessment based on number of case studies, documentation support, training, courses, certifications, community, etc. Alqudah and Razali (2016) covers six of the models and compares them in terms of team size, training, certification, names of methods and practices adopted, level of technical practices required and organizational type, along with their common features mostly based on the existing literature. Kalenda, Hyna and Rossi (2018) provides generally known descriptive information content for the six of the models (SAFe, SoS, LeSS, DAD, Nexus, RAGE). Ozkan and Tarhan (2019b) reviews nine of the scaling models including ones in this study in terms of how they provide scaling.

The current literature focuses on the review of the models yet our study differently aims to provide an evaluation of the models, characterized by its distinct attribute: addressing the models against the identified causes of the challenges in the core, which is beyond what the current literature exhibits.

3 ASSESSMENT OF THE MODELS

In the study of Ozkan and Tarhan (2019a), the following topics of causes of challenges related to scalability were determined and briefly mentioned here:

Physical Dependencies: The face-to-face communication mandates, naturally, synchronizing people in terms of time and place. In this case, abilities relevant to the past and future decline. Addiction to the place requires the same place at the same time for all relevant actors of multiple simultaneous teams and parties. Declined capability of digitalization with documentation and tool feeds physical dependencies and weakens the memory and abilities of reaching for the long future manifesting as a hindrance in achieving scalable dimensions on the time axis, inhibiting flexibility and accompanying scalability.

Fragmentation: Self-management that is a central principle in agile methods can reduce the ability to coordinate across teams effectively (Ingvaldsen & Rolfsen, 2012). While a self-sufficient structure is an advantage, it may cause the teams to become isolated from central formations.

Short and Static Events: Each customer need is a whole and it is, sometimes, not literally dividable into solid, static and short sprint lengths. The same difficulty prevails for time boxes of events at the developer side without a proper and enough "thinking" stage, especially needed for large-size projects when it calls for a longer time than what the framework forces.

Narrow Focus on Product: Sticking to its history, Scrum's roots are nourished mainly by the production lines logic. However, it is difficult to find a proper place for a product-oriented approach in the field of Information Technology that is dominated by a process, service and a project-oriented approach. With lack of existence of a project notion, project definition and project manager, Scrum neglects the (scalable) project space rather, it distributes a whole (project / epic) on multiple teams of products.

Narrow Focus on Construction: Questions survive in Scrum for end-to-end solution development, including pre- and post-development stages, which leads to some obstacles to the scalability. Without a big picture, design development is reduced at the Sprint layer focusing on the low granularity.

Bottlenecks from One to Many Relations: A single product owner in Scrum may have to serve with an ability to reach the scalable dimension horizontally (systems, people, etc.) and vertically (from strategic to operational tasks), which may create a throat for scalability.

The authors went through a deep reading over the primary resources of the models for the related solutions approaches that the models provide. Additionally, the first author made interviews with three experts self-selected by the author to justify the claims of the models for solving identified causes of challenges. The interviewee A (IA) is an Agile transformation coach with SAFe for two years at an international company centred in Europe with more than 460K workers globally. The interviewee B (IB) is an academician holding a PhD degree from Malaysia on scaling Agile, with an experience on the subject since 2009. Interviewee C (IC) is an academician studying a PhD degree in Germany on scaling Agile for four years. During the interviews, the causes of challenges were explained to the interviewees and the solution approaches of the models to them were asked. IA included SAFe and LeSS specifically, IC does so for SAFe, LeSS and SoS and IB delivered information about the all-aforementioned models in general.

All of the interviewees agree on that "**Physical Dependencies**" is a challenge in the large-scale especially in globally distributed environments. Tools are essential for Agile in today's realities [IA, IB, IC] as well as processes [IB]. Thus, the Scrum Guide takes actions and tries to remove the requirement of colocation in the last version [IA]. However, even tools used, face-to-face communication and colocation are needed to build trust in teams [IA]. To support face-to-face communication, for example, the video conference can be used for remote teams [IA, IC]. SAFe and LeSS advocates colocation [IA, IC]. For instance, in SAFe implementations, POs and SMs travel frequently, and even, if physical boards are preferred for two distance locations, two physical copy of the boards are used to synchronize the two sides [IA]. LeSS places more emphasis on colocation [than SAFe] [IA]. The main reason for this is to de-scale and reduce complexity first for a proper scaling [IA]. In SoS, [IC] adds, the representatives play a critical role in communication between multiples teams. In this kind of exponential scaling there may occur loss of information [IC]. According to IC, the document approach in the ASD is largely misinterpreted. He states, "A lot of information hangs on the wall after meetings with wide participation. This is a disadvantage for people who cannot attend to the meeting. The digital version of the information reminds of its importance in such situations."

In terms of "**Fragmentation**", IB puts forward the applied practices of the models is based on Scrum of Scrums. He mentions that to facilitate inter-team coordination and communication, some models propose special roles dedicated to certain domain such as a designer role. IA advocates in SAFe, there is not much fragmentation in the model because of the synchronization and communication mechanism. He adds, "The program layer provides common backlog and Program Increment supports co-operation and transparency between teams. He says in LeSS, there is only one PO to reduce coordination and communication overload. IC mentions, “There are some practices for multiple teams such as Communities of Practice and dependency identification in SAFe. LeSS leaves dependency management more to the team, to stay simple. It recommends less meetings”. IC adds that inter-team and intra-team characteristics of the teams should be in balance, otherwise, team performance may decrease. He says, “The team's autonomy is not very possible in case of dependencies. When Agile leaves its comfort-zone, different factors and unknown situations emerge and the models focus on finding solutions in this regard.”

Within the scope of “**Short and Static Events**”, according to IA and IC, the ASD models (including SAFe, LeSS, SoS) advocates events with time-boxes to deal with complexity and to build a stable cadence for predictable systems. IC adds that showing small portion to customers is useful for feedback even though it is not possible to end up with a demonstrable increment in each iteration. IB, on the other hand, states that decomposability is a tough challenge in Agile, and developers usually hate time boxing. He says, “MVP and lean practices can be a solution to it. For instance, instead of sprint, Kanban uses continues management of the pipeline with no static iterations, which is more agile in essence. More and more the models integrate Kanban in their solutions recently. Additionally, instead of working with the proxy of the customer (PO), the actual customer and development team should be intertwined continuously, as recommended by XP practices, rather than waiting for the end of the sprint to extensively touch the actual customer at one shot.”

For the title of “**Narrow Focus on Product**”, IA says, “A thinking by means of the project concept is not preferred since it constitutes an obstacle to team spirit in agile approaches.” Being product-oriented has already existed since very long past, it just came back to the surface with such streams [IC]. Project is not preferred because it proposes a temporary structure and the long-term team formations of teams support to establish a lasting basis of performance and trust [IA, IC]. SAFe likewise has a product concept [IA]. He adds, “The “Value Stream” concept in SAFe seeks to provide a customer perspective. In this respect, product should be considered as a value stream. SAFe takes epic as a project. Instead of a solid DoD, good enough approach is preferred for epics. Based on the cost of delay, in accordance with the dynamic value of the developments in the pipeline, a selection is made”. However, IB concerns about with a narrow focus on product, challenges may become possible in the integration phases.

In terms of “**Narrow Focus on Construction**”, IB asserts that in the Agile Manifesto, there are no sufficient indications for DevOps, continuous integration and such yet some of the models fill this blankness. IA states, “Scrum implementations are getting closer to the reality, such as with Sprint 0 adoption. For the pre-development phases, in SAFe implementations, value stream is identified. Later on, systems that will be used and who will use these systems are determined. What are going to do in the three-month cycles is addressed and translated to program backlogs.” SAFe provides DevOps, Continuous Delivery Pipeline, and Continuous Integration [IA, IC]. SoS does not provide such post-development activities [IC]. IC adds that “SoS do not propose architect roles but SAFe does. LeSS advocates emergent architecture directed by high-mature teams, which is somewhat optimistic. In general, up-front and emergent approaches in design should be in balance. Contemporary architect roles may contribute to solution for this. Architects may have a role in determining road maps and general architectures from a multiple team perspective, especially in large-projects. It is important that these architectures and road maps be not in the long horizon, but in shorter horizon, simple and adaptive.”

For the “**Bottlenecks from One to Many Relations**”, IB states that this challenge can be overcome with SoS mechanism of product owners at the different layers like second layer PO and Chief PO at the top. IA adds that while the main responsibility at Scrum belongs to the product owner, he/she may also delegate responsibility for the certain operations to the team; therefore, the focus of the product owner should be on value always. However, IC puts forwards that product owners cannot devote enough time to operational works. He says, “For this reason, there are applications that try to support product owners with additional roles. There are many meetings coming with the models and the meetings are waived first”. In LeSS, there is only one PO for a product with the consideration of less dependency existence [IA]. LeSS proposes APOs as a sub-division product management [IC]. In SAFe, chief product owner roles are available [IA]. He adds that “the product owners work with the program manager and reports to him/her. This approach exists to approach to classical structures and to be easily adaptable of the model by such organizations. In essence, this layered structure is against the essence of agility.” Compared to SAFe and LeSS, there is no suggestion regarding this challenge in SoS [IC].

The challenge topics, challenge items and related solution proposals of models according to the authors and interviewees come together as in the Table 1. The added items after considering the interviews is marked by "*" sign. The shares of the interviewees largely coincide with the findings of the authors. Some expressions of the interviewees that support the ideas promoted by the ASD (such as product-oriented development, short and static events) are known but have trade-offs as discussed below.

Table 1: Causes of Challenges and Solution Proposals of Models (*added after considering the interviews).

Challenge	Challenge Items	Solution Proposals of Models
Physical Dependencies	Depending on meetings and face-to-face communication on physical platforms	-
	Lack of digitalization within documentation and tools	**DAD**: disciplined by documentation
Fragmentation	Inter-team coordination and communication, individual, isolated, feudalized and self-organizing teams' environments	**SoS, Nexus, LeSS, SAFe, Nexus, Scrum at Scale, DAD, RAGE**: Clusters of teams/roles conducting common events **LeSS, SAFe, Nexus**: Synchronized events and/or artifacts **SAFe, DAD**: Specific teams/roles with specific responsibilities **DAD**: Non-solo development, collective ownership
Short and Static Events	Fixing maximum duration and the frequency of the iterations and meetings	-
	Breaking whole business needs down to small and solid pieces	**DAD**: Pre- and post-development stages
	Creating a potentially shippable product in each short and static iteration	-
	Starting counting from (sprint) 1	**DAD**: Sprint 0 and -1
	Overall time planning and design of sub-parts of sprints/teams/software/solution	-
Narrow Focus on Product	Production lines logic with a pure product concept	**DAD**: Proposing a generic approach to agile project management **SAFe:** Value stream*
	Lack of existence of project notion	**DAD**: Proposing a generic approach to agile project management
	Lack of project definition	**DAD**: Proposing a generic approach to agile project management
	Lack of project manager definition	**DAD**: Project Manager role
Narrow Focus on Construction	Absence of pre-development phase	**SAFe:** Value stream* **DAD**: Disciplined by documentation, pre-development stages such as Sprint 0 and -1
	Absence of post-development phase	**DAD**: Delivery of consumable solutions over just the construction of working software, transition phase **SAFe**: Release Management Team (RMT), DevOps Team, Continuous Delivery Pipeline, Continuous Deployment, Continuous Integration **Scrum at Scale**: The set of modules includes Release Planning, Release Management, Product & Release Feedback **RAGE**: Release planning, Dev-Ops roles
	Absence of general, top-to-bottom, upfront design without a big picture	**SAFe:** System Team, Architect roles such as Solution Architect, System Architect, **DAD**: Architect roles, Inception phase with lightweight visioning activities including such as Sprint 0 and -1

Table 1: Causes of Challenges and Solution Proposals of Models (*added after considering the interviews) (cont.).

Challenge	Challenge Items	Solution Proposals of Models
Bottlenecks from One to Many Relations	A single Product Owner to serve for a large number of customers and a large number of development teams at the same time	**LeSS, RAGE**: Area Product Owner **DAD, SAFe**: Chief Product Owner* **Scrum at Scale:** Product Owner Team
	Product Owner role for managing a wide range of issues from operation to strategy of a product	-

Table 2: Assessment of the Models with the Causes of the Challenges.

Challenges/Models	SoS	Nexus	LeSS	SAFe	Scrum at Scale	DAD	RAGE
Physical Dependencies	Not achieved	Not achieved	Not achieved	Not achieved	Not achieved	Partially achieved	Not achieved
Fragmentation	Partially achieved	Largely achieved	Largely achieved	Largely achieved	Largely achieved	Largely achieved	Partially achieved
Short and Static events	Not achieved	Not achieved	Not achieved	Not achieved	Not achieved	Partially achieved	Not achieved
Narrow Focus on Product	Not achieved	Not achieved	Not achieved	Partially achieved	Not achieved	Fully achieved	Not achieved
Narrow Focus on Construction	Not achieved	Not achieved	Not achieved	Fully achieved	Partially achieved	Fully achieved	Partially achieved
Bottlenecks from One to Many Relations	Not achieved	Not achieved	Partially achieved	Partially achieved	Partially achieved	Partially achieved	Partially achieved

* Rated in the following scale; Not achieved: 0-15%, Partially achieved: 16-50%, Largely achieved: 51-85%, Fully achieved: 86-100%

Despite the solution proposals of the models, it is noteworthy that there are issues remaining as seen at Table 2, when compared the identified causes of challenges and solution proposals of models. Considering this gap, we evaluated whether the models are capable of solving the stated causes of challenges in terms of ratio (solution/challenge) and Table 2 shows the assessment results (relating to RQ).

4 DISCUSSION

With some exceptions, we see that all of the models try to maintain the pain points associated with scalability in the core of the ASD. When viewed horizontally, it is remarkable that the main and common concentration of the models is what is mainly on the hand: teams and their coordination (Fragmentation). The challenges named as Physical Dependencies, Short and Static Events and Narrow Focus on Product are hardly touched. When viewed vertically, it is seen that the number of models that can respond, partially or fully, to the causes of challenges mentioned in this study is few.

For the “**Physical Dependencies**”, the interviewees are agreed that the requirement of colocation is a challenge especially in distributed environments and tools are needed in the context of global working environments. However, it seems that the majority of the models still cling to the fundamental doctrine of the ASD in this regard.

In terms of “**Fragmentation**”, we see a significant effort of the models maintaining the self-organizing team principles. However, the methods proposed by the models as well as by the majority of the interviewees have some inherited trade-offs. Assuming that teams are able to be self-organized, difficulty in simultaneously standardizing across and preserving some local flexibility (Levina & Vaast, 2005) arise two opposing forces for the teams. Ingvaldsen and Rolfsen (2012) claims that inter-group coordination is a major challenge when groups are self-managing and thus, teams’ being self-organized emerges as a part of the problem when scaling is required (Rolland, et al., 2016). The emergence of dependencies between teams indicates that the ability of teams to act independently and in a self-organized manner is not valid. Simply saying, the goal of being self-organized for each individual team appears to be in opposition to organizing themselves with the consideration of other teams. This is dilemma to overcome for the core in order to support the large scale.

Regarding the communication perspective, reminding that team-of-teams issue may be down to human nature (Lyon & Evans, 2008), it would be misleading to reduce this issue to the level of formal mechanisms as proposed by the models. It is an optimistic approach to solve the communication issues between teams with the principle of composition relying on the simplistic view of the representatives (Rolland, et al., 2016) (dynamic roles), additional roles (static roles) and/or with a meetings paradise, as proposed by the models. The Scrum-of-Scrums meeting is basically the only practice Scrum offers for inter-team coordination (Paasivaara & Lassenius, 2011) however, they are usually inefficient and insufficient for coordinating teams (Paasivaara & Lassenius, 2014). With the proposed approach of sending representatives to the meetings, there are some challenges; the representative must have the abilities to reach his/her team's point of view and the boundaries to represent the cross-functional team in the regarding aspects.

In terms of "**Short and Static Events**", even though the models come with the short iterations and events with time-boxing to deal with complexity and to build predictable systems, it can be a source of the challenge for the large-scale. For instance, the sprint constraints, with which a single team may have difficulties, now appear as a common and even bigger challenge for multiple teams with synchronized sprint schedules, especially with interdependencies arising with issues outside the local teams. Similarly, considered with the time-box limit of sprints, more meetings of the large-scale solutions may create more time pressure resulting in the less people participated in the meetings, as stated by Paasivaara and Lassenius (2016). Interviewee B advocates MVP and lean practices, Kanban for continues management of the pipeline with no static iterations and the actual customer involvement, as the solutions to short iterations and time-boxing issues.

For the "**Narrow Focus on Product**" item, the models (except DAD) do not prefer a structure with the project concept, since project is temporary-based and thus damaging the team spirit. However, the management of the large-scale initiatives over the product teams based on static products may require distributing a whole (of a customer requirement) to the multiples development teams, if the specific requirement touches more than one product. This brings more dependency on static entities (such as products) and weakens flexibility that reinforces agility. In the case of Scrum, being depended on the physical facts and lack of an abstract layer emphasize the need for project notion to encapsulate and manage abstract boundaries. We propose using the project with its unifying and abstracting power. It opens gates to the possibility of gathering individuals around a project-specific team, thus removing boundaries between the product teams during a project rather than distributing a whole (a project) over teams. Even with this feature alone, it can be a significant contributor to solving the inter-team communication issue, which is the focus of most of the models.

For the "**Narrow Focus on Construction**" item, in the core there are no sufficient indications yet some of the models fill this blankness by adopting Sprint 0, release management, DevOps, continues delivery, etc. By definition, we do not consider the preparation activities the teams conduct before the sprint run as a solution to this issue. This item covers development-oriented challenges.

Regarding "**Bottlenecks from One to Many Relations**", the models mainly adds layers and SoS mechanism for POs, as a linear continuation of the core.

It seems major factors in the difficulty of the scalability in the core can rise from two dimensions: horizontal and vertical. Horizontally, an expansion spans multiple systems and team boundaries and vertically (even for a single team), it is the way to breaking down all incoming customer requirements throughout the iterations and merging them into the whole. To cover such an expanded field, most of the models are satisfied with adding additional roles, events, artifacts and layers and synchronizing and integrating sprints of associated teams to provide coordination between teams. These are what the Scrum provide: events, roles, artifacts (for a single team). Thus, the majority of the models provide a general approach to scaling as a linear continuation of the core, initially intended for small-scale.

The similar case can be seen in amplifying by duplication of "the same" (adding more product owners, product backlogs, meetings etc.) on a quantitative basis. This is basically because of that Agile principles seen as axiomatic and sacrosanct, and tailoring of agile methods is achieved through adding practices (Rolland, et al., 2016). Addition of identical or similar practices is easier and default part. However, "adding" is apparently simple part of the "solution" yet it may not work as expected as stated in the cases by Paasivaara and Lassenius (2011), Pawlowski and Robey (2004) and Lyon and Evans (2008). Most of the models present additional roles for coordination, yet, the majority of the roles focus on vertical coordination rather than horizontal coordination. With a similar approach, dividing the project into "orthogonal" parts works only if the complexity is of the additive kind (Meyer, 2014), which is not a usual case in the large-scale. Briefly, a solution to large-the scale does not simply imply "more of the same" found in small-scale agile projects (Robinson & Sharp, 2010).

Additionally, such add-ons can lead to more complex systems and networks of interdependencies (Perrow, 1999), which do not suffice and go against the very idea of agility itself (Rolland, et al., 2016). Such a mechanism deepens the scaling problem by reinforcing hereditary constraints related to scalability in the core. Shortly saying, complex interdependencies intrinsic to large-scale agile projects make laying on practices and principles from small-scale projects problematic (Rolland, et al., 2016). Different kind of problems associated with potential "complex interactions" and "tight coupling" (Perrow, 2011) should be handled in a different way (Rolland, et al., 2016).

5 CONCLUSION AND FURTHER WORK

The models have challenges to scale as they try to establish their solutions on Scrum that has already created a "sweet spot" (Reifer, Maurer & Erdogmus, 2003) for itself on the small scale in practice. Leaving this comfort zone to reach scalable areas, by upgrading the core or not, is necessary if the agility is assumed to be the right of organizations regardless of their size. Such challenges in this work may also lead to questioning the core in terms of scalability and may imply a call for an endeavour for re-designing or at least re-reading of the Agile Software Development Manifesto and Scrum values and principle with scalability considerations.

There are two main points that the agile scaling models claim: providing scaling and maintaining agility even when providing scaling. Assuming that the models produce solutions for the large-scale, it is a separate assertion that the models can remain agile during doing so, with their quite mechanical and rigid structures as found in some models, making the actual agility hardly possible. It should be studied as a future work how much they offer agility at the last point, especially with their own additions. We will also in particular propose new scaling models preserving or not the core of the ASD, primarily and mainly designed about two decades ago, in the year of 2001 and we will provide possible updates to the core to come up with solutions supporting more effective scalability.

REFERENCES

Abrahamsson, P., Warsta, J., Siponen, M. T., & Ronkainen, J. (2003, May). New directions on agile methods: a comparative analysis. In *Software Engineering, 2003. Proceedings. 25th International Conference on* (pp. 244-254). IEEE.

Alqudah, M., & Razali, R. (2016). A review of scaling agile methods in large software development. *International Journal on Advanced Science, Engineering and Information Technology*, *6*(6), 828-837.

Ambler, S. W., & Lines, M. (2012). *Disciplined agile delivery: A practitioner's guide to agile software delivery in the enterprise*. IBM Press.

Costa, N., Santos, N., Ferreira, N., & Machado, R. J. (2014, June). Delivering user stories for implementing logical software architectures by multiple scrum teams. In *International Conference on Computational Science and Its Applications* (pp. 747-762). Springer, Cham.

Dingsøyr, T., & Moe, N. B. (2013). Research challenges in large-scale agile software development. *ACM SIGSOFT Software Engineering Notes*, *38*(5), 38-39.

Dingsøyr, T., & Moe, N. B. (2014). Towards principles of large-scale agile development. In *International Conference on Agile Software Development* (pp. 1-8). Springer, Cham.

DSDM Consortium (2014). DSDM Agile Project Framework - Handbook. Ashford, Kent, UK: DSDM Consortium.

Hobbs, B., & Petit, Y. (2017). Agile methods on large projects in large organizations. *Project Management Journal*, *48*(3), 3-19.

Hoda, R., Kruchten, P., Noble, J., & Marshall, S. (2010). Agility in context. *ACM Sigplan Notices*, *45*(10), 74-88.

Ingvaldsen, J. A., & Rolfsen, M. (2012). Autonomous work groups and the challenge of inter-group coordination. *Human Relations*, *65*(7), 861-881**.**

Kalenda, M., Hyna, P., & Rossi, B. (2018). Scaling agile in large organizations: Practices, challenges, and success factors. *Journal of Software: Evolution and Process*, *30*(10), e1954.

Kniberg H. & Ivarsson, A. (2012). Scaling Agile @ Spotify [Online]. Available: https://blog.crisp.se/wp-content/.../11/SpotifyScaling.pdf Spotify scaling-agile-spotify-11.pdf.

Larman, C., & Vodde, B. (2016). *Large-scale scrum: More with LeSS*. Addison-Wesley Professional.

Leffingwell, D. (2011). Scaling Agile Framework (SAFe). [Online] Available: http:// www.scaledagileframework.com/.

Levina, N., & Vaast, E. (2005). The emergence of boundary spanning competence in practice: implications for implementation and use of information systems. *MIS quarterly*, 335-363.

Lyon, R., & Evans, M. (2008, August). Scaling up pushing scrum out of its comfort zone. In *Agile, 2008. AGILE'08. Conference*(pp. 395-400). IEEE.

Meyer, B. (2014). *Agile: The Good, the Hype and the Ugly*. Switzerland: Springer International Publishing.

Moore, E., & Spens, J. (2008, August). Scaling agile: Finding your agile tribe. In *Agile, 2008. AGILE'08. Conference* (pp. 121-124). IEEE.

Ozkan, N., & Tarhan, A. K. (2019a). Investigating Causes of Scalability Challenges in Agile Software Development from a Design Perspective. *In 2019 1st International Informatics and Software Engineering Conference (UBMYK),* pp. 1-6. IEEE.

Ozkan, N., & Tarhan, A. (2019b). A Review of Scaling Approaches to Agile Software Development Models. *Software Quality Professional*, 21(4), 11-20.

Paasivaara, M., & Lassenius, C. (2011, September). Scaling scrum in a large distributed project. In *Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on* (pp. 363-367). IEEE.

Paasivaara, M., & Lassenius, C. (2014). Communities of practice in a large distributed agile software development organization–Case Ericsson. *Information and Software Technology*, *56*(12), 1556-1577.

Paasivaara, M., & Lassenius, C. (2016, August). Scaling scrum in a large globally distributed organization: a case study. In *Global Software Engineering (ICGSE), 2016 IEEE 11th International Conference on* (pp. 74-83). IEEE.

Pawlowski, S. D., & Robey, D. (2004). Bridging user organizations: Knowledge brokering and the work of information technology professionals. *MIS quarterly*, 645-672.

Perrow, C. (1999). Organizing to reduce the vulnerabilities of complexity. *Journal of contingencies and crisis management*, *7*(3), 150-155.

Perrow, C. (2011). *Normal Accidents: Living with High Risk Technologies-Updated Edition*. Princeton university press.

Reifer, D. J., Maurer, F., & Erdogmus, H. (2003). Scaling agile methods. *IEEE software*, *20*(4), 12-14.

Robinson, H., & Sharp, H. (2010). Collaboration, communication and co-ordination in agile software development practice. In *Collaborative software engineering* (pp. 93-108). Springer, Berlin, Heidelberg.

Rolland, K.H., et al. (2016). Problematizing agile in the large: alternative assumptions for large-scale agile development.In *Thirty Seventh International Conference on Information Systems.* Dublin.

Schwaber, K. (2004). *Agile Project Management with Scrum*. Microsoft Press, Redmond, Washington, USA.

Schwaber, K. (2015). Nexus Guide. The Definitive Guide to Nexus: The exoskeleton of scaled Scrum development. *PDF). scrum.org.*

Sutherland, J. (2001). Inventing and Reinventing SCRUM in five Companies. *Sur le site officiel de l'alliance agile*.

Sutherland, J. (2019). SCRUM@Scale Guide. The Definitive Guide to Scrum@Scale: Scaling that Works.

Thompson, K. (2016). Rage – recipes for agile governance in the enterprise. [Online]. Available: https:// www.cprime.com/rage/.

Uludağ, Ö., Kleehaus, M., Xu, X., & Matthes, F. (2017). Investigating the role of architects in scaling agile frameworks. In *Enterprise Distributed Object Computing Conference (EDOC), 2017 IEEE 21st International* (pp. 123-132). IEEE.

Vaidya, A. (2014). Does dad know best, is it better to do less or just be safe? adapting scaling agile practices into the enterprise. *PNSQC. ORG*, 1-18.

Versionone. (2017). *9th annual state of agile survey*. Technical report, Version One.