A Review of Scaling Agile Methods in Large Software Development

Article · December 2016

DOI: 10.18517/ijaseit.6.6.1374

2 authors:

Mashal Alqudah
National University of Malaysia
7 PUBLICATIONS **6** CITATIONS
SEE PROFILE

Rozilawati Razali
National University of Malaysia
56 PUBLICATIONS **194** CITATIONS
SEE PROFILE

International Journal on Advanced Science Engineering Information Technology

Vol.6 (2016) No. 6
ISSN: 2088-5334

A Review of Scaling Agile Methods in Large Software Development

Mashal Alqudah[#], Rozilawati Razali[*]

[#*]*Research Center for Software Technology and Management, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Bangi, Selangor, 43600, Malaysia*
E-mail: mashal_alqudah@yahoo.com, rozilawati@ukm.edu.my

***Abstract*— Agile methods such as Dynamic Systems Development Method (DSDM), Extreme Programming (XP), SCRUM, Agile Modeling (AM) and Crystal Clear enable small teams to execute assigned task at their best. However, larger organizations aim at incorporating more Agile methods owing to the fact that its application is prevalently tailored for small teams. The scope in which large firms are interested will extend the original Agile methods to include larger teams, coordination, communication among teams and customers as well as oversight. Determining particular software method is always challenging for software companies especially when considering start-up, small to medium or large enterprises. Most of large organizations develop large-scale projects by teams of teams or teams of teams of teams. Therefore, most recognized Agile methods or first-generation methods such as XP and SCRUM need to be modified before they are employed in large organizations; which is not an easy task. Accomplishing said task would necessitate large organizations to pick and select from the scaling Agile methods in accommodating a single vision for large and multiple teams. Deciding the right choice requires wholesome understanding of the method including its strengths and weaknesses as well as when and how it makes sense. Therefore, the main aim of this paper is to review the existing literature of the utilized scaling Agile methods by defining, discussing and comparing them. In-depth reviews of the literature were performed to juxtapose the methods in impartial manner. In addition, the content analysis was used to analyse the resultant data. The result indicated that the DAD, LeSS, LeSS huge, SAFe, Spotify, Nexus, and RAGE are the adopted scaling Agile methods at large organizations. They seem to be similar but there are discrepancies among them that take the form of team size, training, and certification, methods and practices adopted, technical practices required and organizational type.**

***Keywords*— Agile method; scaling Agile methods; DAD; SAFe; LeSS; Spotify; Nexus; RAGE**

I. INTRODUCTION

Agile method is the overarching term coined for a set of software development methods in line with four values and twelve principles stated in the Agile Manifesto [1]. The findings of 2013 IT Project Success Rates Survey Results attested to the superiority of Agile methods in the context of effectiveness and successfulness relative to its traditional counterparts [2]. SCRUM, Extreme Programming (XP), Crystal Clear, and Dynamic Systems Development Method (DSDM) were the major Agile methods opted when firms migrated to Agile approach [3, 4]. Furthermore, Feature Driven Development (FDD), Test Driven Development (TDD) and KANBAN along with most of Agile methods gained firm traction; therefore, have seen extensive applications in the industry [5] due to their assistance in incremental and iterative implementation of user requirements [6]. Agile methods offer specific practices that inform team on level practices; although they do not generally provide sufficient consideration to the risks that are related to delivering solutions on larger enterprise projects [7], [8]. For instance, SCRUM is a minimal set of practices that is widely utilized in the industry [3]; as defined by SCRUM Alliance, SCRUM indicates. There are three roles allocated in a SCRUM team: the Product Owner (PO), a SCRUM Master (SM), and a cross-functional team. SCRUM specifies each team to manage its work through four artefacts: a Product Backlog, a Sprint Backlog, a Product Increment, and Definition of Done. Meanwhile, SCRUM specifies five activities for teams in order to achieve its goals: Backlog Refinement, Sprint Planning, Daily SCRUM meeting, Sprint Reviews, and Sprint Retrospective. Hence, SCRUM makes no specific recommendations in scaling beyond the team level with less than 10 stakeholders.

Similarly, Kanban method is even less specific in its compulsory practices. In most of Kanban implementation, the main practices that enable process evolution are Visualise, Limit-work-in-progress, Manage Flow, Make Policies Explicit, Implement Feedback, Loops and Improve Collaboratively [9]. Just as Scrum, Kanban has high dependency on self-organised teams as well as excessive involvement of leadership in devising other practices that facilitate the adoption of Agile into the organisation.

Extreme Programming (XP) is another recognised Agile method [3]. XP enhances a software project in five crucial ways such as communication, simplicity, feedback, respect, and courage. Moreover, it comprises of 12 practices: Pair programming, planning game, test-driven development, whole team, continuous integration, refactoring, small releases, coding standards, collective code ownership, simple design, system metaphor, and sustainable pace. In spite of that, XP does not provide specific recommendations in scaling beyond the team level which should be 7 plus or minus 2 [10].

The result of aligning teams of teams to work together while still employing traditional methods and organisations is the profuse effort that generates hybrid methodologies combining techniques from different methods [7], [11]. Therefore, Agile different scaling methods (Frameworks) such as Scaled Agile Framework (SAFe), Disciplined Agile Delivery (DAD) and Large Scale Scrum (LeSS) were developed in order to resolve team size issue and other issues associated with customer involvement, project constraints, business case approval, interacting with partners and end-users, and project benefits realization [12]. Hence, it is intended to deal with the full, end-to-end delivery lifecycle [7], [11], [13]. Other methods such as Spotify, Nexus, and Recipes for Agile Governance in the Enterprise (RAGE) were developed at large organizations for the same purposes.

The scaling methods used above are the combination of methods or practices based on various Agile and Lean methods [13]. According to the proponents of scaling methods, an organisation that adopts a scaling method will certainly be at an advantage [12] especially in the light of scarcity of scaling Agile methods incorporation in large organizations [14], [15]. Therefore, the main aim of this paper is to review the scaling Agile methods, comprehend their roles and practices, and to identify the differences and similarities among them.

The pertinent issues are addressed by this paper in 5 sections. Section I introduces Agile and scaling Agile methods. Section II provides the related work of scaling Agile methods (Frameworks), Section III describes the research methodology that was employed in this study to explain various scaling Agile methods that have been used. Section IV presents the findings and discussion obtained from the review of the related works. Finally, Section V concludes and defines the scope of further work that could be undertaken on this topic.

II. Materials And Methods

The section discusses a brief background of scaling Agile methods when they started to emerge as well as each of method main practices.

A. Discipline Agile Delivery (DAD)

The main objective of Disciplined Agile Delivery (DAD) is to help fill in the gaps by extending the Scrum construction lifecycle to address the full delivery lifecycle while adopting practices from other Agile methods including Lean and Kanban [7]. Therefore, DAD is a hybrid process which extends SCRUM lifecycle with proven strategies from many methods such as Agile Modeling (AM), Extreme Programming (XP), Unified Process (UP), Kanban, Lean Software Development, Outside In Development (OID), Agile Data (AD) and several other methods" [16]. In addition, Ambler (2012) highlights that the focus of DAD is to address the project lifecycle from the point of initiating the project (Inception phase) to construction to releasing the solution into production (Transition phase). Adding together, DAD teams focus on delivering repeatable results which entail delivery of high-quality software; however, they do not aim at following the repeatable process" [7].

1) DAD Roles: DAD is adopting roles from SCRUM and Agile Modeling methods. The roles of DAD start with Product Owner and primary Team Members (SCRUM roles). Besides, a role called Team Lead which is similar to Scrum Master in Scrum and an Architecture Owner role were introduced from Agile Modeling method. DAD has secondary roles as well namely Specialist, Independent Tester, Domain Expert, Technical Expert and Integrator which are brought in to address scaling issues and can be arranged on a temporary basis [11].

2) DAD Practices: DAD framework occupies four distinct lifecycles, namely Agile/Basic, Advanced/Lean, Continuous Delivery Lifecycle, and Exploratory Lifecycle [16]. It is expected that an organization can employ the lifecycles to suit its needs. The basic lifecycle has three phases named Inception, Construction, and Transition. Table 1 shows the description of the phases.

TABLE I

PHASES OF DISCIPLINE AGILE DELIVERY FRAMEWORK

Inception Phase	Construction Phase	Transition Phase
Main goal		
Understanding the scope and what you want to achieve especially when a team is working on the first release of new products (lightweight visioning activities).	Producing consumable solution that provides business value and addressing any problems in the early development of the product by improving the quality and proving architecture early.	Arranging solution into production by careful planning and coordination across multiple teams to deploy a consumable solution into the hands of the customer (Ensuring Your Production Readiness).
Main practices		
Prioritization of the projects and selecting the appropriate ones. Form the team immediately after selecting the projects. Identification of the project vision and secure funding. Initial modelling and architectural visioning might be included based on the project type.	The practices of the construction phase are adopted from SCRUM practices (almost all practices) but mixed with other practices of XP (Continuous integration, refactoring, TDD, collective ownership), Lean/Kanban (Visualize the workflow), Agile Date (Database refactoring). Other advanced practice	Transition planning is the first practice of this phase, then at the end of each lifecycle testing should be done to fix any issue. The migration of data from old system to the new one and database refactoring should be adopted. After that, testing the product by a subset of the end-user should be done to get accurate feedback when

Initial release planning which addresses financial and schedules concern which assists in secure funding for your project. Exploring and Identifying initial requirements to have better understanding of the scope before the constructions begin.	such as Acceptance TDD (ATDD), Continuous deployment (CD), Look-ahead modelling, Parallel independent testing and Non-solo development should be adopted at this stage	using the product. Finalize documentation by including user manuals, operations manuals, installation guides, and so on. Communicate deployment, prepare support environment and train the customer are other practices of this phase. So, the customer is delighted with the final product.

The second lifecycle, Advanced/Lean, is nearer to Kanban in its Construction phase. Just as the first lifecycle, the Inception phase is used to stock a work item pool where the work item pools are organised along business values, fixed delivery date, expedited or some other intangible categories. On the other hand, unlike Agile/Basic, planning, retrospective, demos, stands up, and other activities are conducted as needed including deployment readiness during the Transition phase. Meanwhile, some upfront architectural modelling and visioning similar to the first lifecycle are carried out during the Inception phase [7].

Moreover, in the third lifecycle, Continuous Delivery lifecycle, the Inception phase is explicit and has very brief Transition period. In this lifecycle, a "product is shopped into production or the marketplace on a very regular basis…as often as daily. However, weekly and monthly basis are common as well [7]. Therefore, this cycle is often termed as a 'leaner' version of Advanced/Lean lifecycle.

The last lifecycle, Exploratory lifecycle, is intended to encourage the Agile or Lean teams to find themselves in startup or research situations where the stakeholders have clear ideas for a new product while do not have understanding the needs of their user base. This lifecycle comprises of six activities such as envisioning, prototyping, deploying, observe and measuring and based on the feedback cancelling or productizing the idea phase [7].

B. Large Scale Scrum (LeSS)

Larman & Vodde (2013) describe basic LeSS as a method that is applied to a median implementation covering approximately "70 people on one product and for LeSS huge thousands of people on one product at 5 sites with about 15 million lines of source code" [17]. Therefore, LeSS is also articulated as a method that applies Scrum to considerable multisite and offshore product development.

The organizational changes are specified in LeSS method while they are not directly addressed in the standard Scrum. Besides, LeSS also specifies cross-functional, cross-component, end-to-end feature teams through the exclusion of traditional team lead and project manager roles. This is also highlighted by Larman and Winn (2014) in a case study at J.P. Morgan by stating that the feature teams each had a "blend of domain, technical and functional skills." LeSS can cover up to ten Scrum teams; therefore, it can be suggested that LeSS should be used for up to 70 people [17], [18].

1) LeSS Roles: According to Larman & Vodde (2013) a single Product owner is common to all ten teams in basic LeSS [17]. In addition, there are no other special roles specified compared to standard Scrum.

2) LeSS Practices: There is a change in Sprint Planning meeting in LeSS; each of the Scrum teams is represented by two members per team plus the one overall Product Owner to decide which chunk of Product Backlog items to work on. This is in contrast with the standard Scrum where the rest of Scrum team participates. When a contention occurs in a backlog item, Product Owner mediates between teams. Likewise, Sprint Review changes to a single meeting for all Scrum teams. However, it is limited to two team members per each Scrum team [17]. Three more practices were established because of these changes, namely: Inter-team coordination meeting, a Joint Light Produce Backlog Refinement and Joint Retrospective meeting.

In order to improve the information sharing and coordination, the Inter-team coordination meeting can be conducted regularly during the week by occupying various formats including an Open Space, Town Hall meeting, Multi-Team Daily Scrum or Scrum of Scrums. While the Joint Light Product Backlog Refinement, restricted to two team representatives, has a maximum duration not to exceed 5% of the Sprint duration. The meeting aims at refining product backlog items for upcoming Sprint. In a Joint Retrospective meeting, the aim is to identify and plan improvement experiments cooperatively for the overall product or organisation [17].

In addition to the above practices, LeSS also has alternative practice named In-Sprint Item Inspection where the teams unofficially try to find early feedback from the PO or other stakeholders on finished product Backlog items. LeSS Huge is almost similar to LeSS but it meant for huge projects which have thousands of people working on one product. In LeSS method 2 there is a need for area product owner, area backlog views, pre-sprint product owner team meeting, area level meeting, overall sprint review and overall sprint retrospective [17]. So, the coordination among multiple teams will be done effectively and efficiently [12].

C. Scaled Agile Framework (SAFe)

The Scaled Agile Framework (SAFe) established by Dean Leffingwell and his collaborators highlights four levels (Layers) of organisation i.e. Team, Program, Value Stream and Portfolio. Each level carries its own activities and all levels are tied together [19]. According to Leffingwell integrates Agile and Lean practices at all four levels, and it offers team and program size patterns which later can be used for scaling across larger organisation. SAFe specifically identifies standard Scrum team size which consists of five to nine team members. A program is described as consisting of determined five-twelve Agile teams or 50-125 individuals that are dedicated to the program and capable of supplying business capability or value [19]. Value Stream level is to insure that multiple teams remain aligned. At the portfolio level, the teams will be focusing on new ideas.

1) SAFe Roles: At the team level, an Agile maintains the resemblance with a typical Scrum team with a few variations. It has a ScrumMaster that could be a part-time role for a team member of 25-50% or a single ScrumMaster that may be shared across 2-3 teams [19].

Just as standard Scrum, the team which is referred to as ScrumXP, maintains to have Product Owner and a team of five-nine team members. It can be a specialized component team, but it does not have to be broadly cross-functional or a feature team. ScrumXP teams collection coordinates with each other in order to develop and deliver cohesive end-user value. To achieve that, a ScrumXP team must have the ability to design, build and test its own work.

Some new roles and teams are established at the Program level. In this level, a Product Manager role serves as the 'content authority' for the release train and he is responsible for identifying the priority of Program Backlog. In addition, the Product Manager works with Product Owners to optimize Feature delivery and direct the work of Product Owners at the team levels. Furthermore, A System Architect's role is to execute some up-fronts architecture and guide the emergent architecture for all program teams. If a Product Manager is called a "chief Product Owner" for the program, a Release Train Engineer's role is as a "chief ScrumMaster" whose tasks include facilitating program level processes and program execution, escalates impediments, manages risk, and helps drive program-level continuous improvement. In addition, a User Experience or U-Designer offers cross-program design guidance in order to supply such consistent user experience across the components and systems of the larger solution [19].

Besides the individual roles previously described, SAFe has additional program level teams as well as described below. A Business owner team that consists of three-five stakeholders have "the ultimate fiduciary, governance, efficacy and ROI responsibility for the value delivered by a specific release train"; A Release Management Team (RMT) has the tasks for scheduling, managing and governing of synchronised released; a DevOps team that provides "tighter integration of development and operations as well as maintains deployment readiness for the program; a System Team whose responsibilities constitute providing assistance in building and using the development environment infrastructure which includes Continuous Integration, build environments, testing platforms, and Test Automation framework as well as integrating code from Agile Teams, conducting end-to-end system testing, and showing solutions to stakeholders at each iteration. When 125 people are needed to work on one large product, the value stream level may be created. This level is to insure that multiple large teams remain aligned. So, at this level, there is a need for Value Stream Engineers, Solution Manager and Solution Architect roles to assist the teams at the program level.

There is also a Program Portfolio Management team that characterises the "highest-level fiduciary (investment and return) and content authority (what gets built)." This team consists of business managers and executives who fully understand the enterprise business strategy, technology, and financial constraints…". The responsibilities involve providing portfolio vision, strategic and investment funding and overall portfolio government [19].

2) SAFe Practices: SAFe demonstrates a mix of Scrum and XP practices at the team level. That SAFe specifies practices surrounding quality or Agile software engineering practices that are mainly generated from XP is the most important departure from Scrum. The code practices include Agile Architecture, Continuous Integration, Test-First, Code Refactoring, Pair Work, and Collective Code Ownership. Besides, SAFe will not expect teams to produce Potentially Shippable Increment (PSI) every Sprint, but quarterly cadence. Moreover, it provides features at the program level which the teams deconstruct and size to fit into iterations [19].

At the program level, SAFe utilises an Agile Release Train (ART) to develop large-scale systems. An ART consists of four two-week iterations which are followed by a three-week HIP (Hardening, Innovation, and Planning) iterations. Teams dedicated to the ART will synchronously develop and release a PSI on the same quarterly cadence. In this level, a system team normally operates iteration behind single ART teams. Moreover, the system team integrates code from various ART teams and performs features of 'end-to-end and system performance testing' which are carried out either manually or through test automation.

In addition, the system team helps stage a 'system Sprint demo' where teams demonstrate the whole system to the stakeholders. The planning will be conducted at the end of every quarter during the HIP iterations. It will be done by all teams within the ART during an all-hands Release Planning meeting. The ART arranges a twice-a-week Scrum of Scrum for all the dedicated ART teams during execution of a PSI.

SAFe brings in concepts of investment themes and values streams aligning the ARTs at the program level. A value stream is described as long-lived series of system definition, development and deployment process steps which is utilised to develop and deploy systems that supply a constant flow of value to the business, customer or end user [19]. In SAFe parlance, a value stream is manifested through an Agile Release Train at the program level, and investment theme reveals how the portfolio distributes budget to the release train implementing the portfolio strategy [19].

The themes eventually act as a funnel, seeding a portfolio backlog with business or technical epics. There are two types of Epics in SAFe – Business and Architectural. Business Epics are "large, typically cross-cutting customer-facing initiatives that encapsulate the new development necessary to realize certain business benefits" [19]. In addition, Architectural Epics are "cross-cutting technology initiatives that are necessary to evolve portfolio solutions to support current and future business needs" [19].

D. Spotify

The main purpose of Spotify is to deal with multiple teams in a product development organization. Spotify has over 30 teams across three cities, but it has kept an agile mindset in its organization. Spotify's main focus then can be described as to be able to control agile with hundreds of developers [20].

Spotify has numerous Squads, in which similar to Scrum team. Each Squad is a self-organizing team that uses its own preferable method; some of the Squads use Scrum, the other ones use Kanban, and some others use the combination of

both. Squads in Spotify are encouraged to implement Lean Startup principles such as MVP (Minimum Viable Product) and validated learning. However, each Squad has a long-term mission and it sticks with the mission which is part of the product [20]. There is no such appointed squad leader in each Squad, but it has a Product Owner. The Product Owner is responsible for prioritizing the work done by the team. However, how the teams work is not the area that the Product Owner can control [20], [21].

The Product Owners of different squads collaborate to maintain a high-level roadmap document that shows where Spotify as a whole is heading. Moreover, maintaining a matching product backlog for each squad is also the Product Owner's responsibility [20]. Apart from that, a Squad has access to an agile coach. The agile coach will help the squad to progress and improve their ways of working to achieve the goal [20].

In Spotify, there is also a tribe. It is a collection of squads whose aim is to minimize dependencies that can obstruct or slow a squad. These squads work in the same location of office in order to promote collaboration between squads. Each tribe is led by a tribe leader whose responsibility includes providing the best possible habitat for the squads within the tribe [20]. Tribes are basically designed to be smaller than 100 people, and they conduct regular-basis gatherings to show what they have worked on, delivered and achieved so that others can learn from them [20].

Besides tribe, Spotify has Chapter as well. Chapter is a small group of people having similar skills 'different testers from different tribe' and working within the same general competency area, but within the same tribe. Chapters are the glue that sticks the company together by giving the company some economies of scale without sacrificing too much autonomy. The regular meeting of chapters of testers and chapters of designers, for instance, can help to identify and solve the problems faster [20]. Finally, there is also a Guild, a group of people whose desire is to share knowledge, tools, code, and practices. There are we technology guild, tester guild, agile coach guild and many others. A Guild is more organic and wide-reaching. While Chapters are always local to a Tribe, a Guild usually cuts across the whole organization [20]. Spotify with Squads, Tribes, Chapters, and Guilds was just introduced over the past seven years, so some people are still unfamiliar with it. However, the results of surveys showed that scaling model seems to be working well with them [20].

1) Spotify Roles: The Spotify Team consists of a Product Owner, Agile coach, Squad, Tribes, Tribe leader, Chapters, and organizational support to solve problems and technical issues [20], [21].

2) Spotify Practices: Spotify allows each squad to choose its own methods. Some apply Scrum while others use Kanban based on their needs. Spotify also takes advantages from DevOps, Lean Startup and SCRUM of SCRUM "just on demand" [20], [22].

In brief, Spotify has different squads "similar to Kanban, Scrum or Lean Startup teams" multiple squads which share dependencies and will be one tribe. From each tribe, there will be a chapter team to help each other to the problems. Finally, A Guild is members from different chapters and different tribes to solve different problems and share knowledge in the whole organization [20], [22], [23].

E. Nexus Method

The key purpose of Nexus has always been related to development and maintenance of scaled product and project of software development. Nexus is a structure that consists of roles, events, and techniques that unite the work of more or less three to nine Scrum Teams which work on a single Product Backlog in order to build and Integrated Increment that achieves a target [24].

The complications take place when there are more than one Scrum Teams working on the same Product Backlog and in the same codebase for a product. The problem of communication will arise if the developers are not in the same collocated team while their work is related to each other. Moreover, when they work in different teams, they will have issues with the integration of their work and the testing of the Integrated Increment [24].

1) Nexus Roles: The Integration Team in Nexus involves a Product Owner, a Scrum Master, and Nexus Integration Team member whose tasks are to coordinate, coach, and supervise the application of Nexus and the operation of Scrum in order to deliver the best outcomes [24].

2) Nexus Practices: The duration of Nexus events is determined by the length of the subsequent events in the Scrum Guide. They are timeboxes in addition to their subsequent Scrum events; although there are some additional practices that have been implemented in Nexus [25].

Nexus is exoskeleton of scaled Scrum. It has a product backlog for a large project which needs different Scrum Teams to work on it. After the product backlog is refined and decomposed, Nexus Sprint Planning will coordinate the activities of all Scrum Teams a Nexus for a single Sprint "Product Backlog items for each team." In this process, the Product Owner provides domain knowledge and guides selection as well as priority decisions; whereas each Scrum Team plans its own Sprint. The results generated are a set of Sprint Goals that align with the overarching Nexus Goal, each Scrum Team's Sprint Backlog and as a single Nexus Sprint Backlog [24]. Furthermore, a Nexus Sprint Backlog of the individual Scrum teams is combined to emphasize the dependencies and the flow of work during the Sprint. This is updated at least daily, which frequently becomes a part of the Nexus Daily Scrum. The beginning of development then starts. This involves all teams developing the software. In this process, all teams will integrate their work into a common environment and test the integration to ensure it is done [25].

This stage will include the daily meeting of the respective representative of each team. The purpose of this meeting is to identify if the integration issues exist. The existence of the issue might result in the change of the day's plan. The meeting of all teams and the Product owner is also held in the Nexus Sprint Review. In this stage, some adjustments might be made to the Product Backlog [24]. Eventually, there should be Nexus Sprint Retrospective where respective representatives from each Scrum Team share the challenges faced. Following this activity, each Scrum Team conducts individual Retrospectives same as following Scrum; and

then those representatives from each team will discuss the actions needed to be done based on the shared challenges. This is expected to provide bottom-up intelligence [24].

F. Recipes for Agile Governance in the Enterprise (RAGE)

The primary objective of RAGE is for allowing quick decisions in according to lightweight artefacts created using minimum work and can be employed in any process including Agile, Hybrid, and Plan-Driven. In other words, aside from not unbendingly prescriptive, Agile Governance is also adaptable. This allows our solution architects to assist in forming a custom "recipe" for optimizing the processes at all organizational levels [26], [27]. RAGE demonstrates the way in which the accomplishment of Agile Governance is possible via the definition and utilization of a set of standardized elements such as Metrics, artefacts, Roles, Ceremonies, as well as Governance Points at the differing levels of an organization. Agile principles can be seen in the entire Governance Recipes except that they are not limited to Agile projects in terms of usage [26], [27].

Thompson divides levels of governance into Portfolio, Program, and Project levels, and reviews practices that are appropriate for each. At the project level, a team will be similar to SCRUM team (SCRUM Master, Product owner, Team), and their main ceremonies are the refinement of product backlog, sprint planning, daily SCRUM, sprint review and conducting retrospective similar to SCRUM way of working. In addition, the artefacts are the definition of done, stories and epics, product backlog and sprint backlog. Adding together, at the governance point element, the team at the project level should apply ranking estimation and story development. Finally, to track the work, the team at this level could use cumulative flow chart, burndown chart or taskboard [26], [27]. At the program level, the roles are to have area product owner and program manager to conduct release planning meeting, SCRUM of SCRUM meeting or release review. In addition, the artefacts are the definition of done and release planning. Adding together, at the governance point element, the team at the program level focuses on release handoffs, staging readiness review, product readiness review and product development validation. Finally, to track the work, the team at this level could use burn up chart for the releases [26], [27]. At the portfolio level, the focus is on decision and planning. The main roles are portfolio owner, area product owner and program manages to conduct portfolio grooming meeting to develop value and estimate effort for initiation and portfolio planning meeting to reach a final decision regarding proposed initiatives. The artefacts at this level include portfolio backlog, decision matrix, Agile charter and business case. In Addition, at the governance point element, the team at the portfolio level focuses on initiative assessment, funding decision, monitoring, and quality assessment. Finally, to track the work, the team at this level could use burn up chart to track the progress of initiatives and keep track of the budget [26], [27].

1) RAGE Roles:

Project Level: SCRUM Master, Product owner, Team
Program Level: Area PO, Program Manager
Portfolio Level: Portfolio Owner, Area PO, Program Manager [27].

2) RAGE Practices: The practices at the project level are more similar to the practices of SCRUM.

However, at the program level, release planning meeting, SCRUM of SCRUM meeting and release review should be conducted, whereby at the portfolio level, portfolio grooming meeting and portfolio planning meeting should be conducted to make sure every team is working on the right requirements [26], [27].

G. Research Method

The main aim of this paper is to discuss the comparison of the scaling Agile methods. In doing so, a review was conducted to identify relevant and available studies in defining criteria comparing and highlighting the differences and similarities of aforementioned scaling Agile methods. The review is based on key words, study type, language, and publication year.

The search key words for the studies were (“Scaling Agile framework” OR “Scaling Agile method”): The scaling Agile methods used in the software industry are, DAD, SAFe, basic LeSS, LeSS huge, Spotify, Nexus, and RAGE. In inclusion criteria, for each category, the following key points were used to include the studies:

- Studies that discuss the scaling Agile methods used,.
- Studies that present a comparison between different scaling Agile methods.
- Studies that investigate the practices, roles and responsibilities, tools and techniques and the development process within one or more of the scaling Agile methods. On the other hand, the key points used to exclude studies in the exclusion criteria are studies that do not discuss the scaling Agile methods as their main objective or the studies are not in English language.

This study considers publications since the year 2001 when Agile methods started gaining serious attention from research community and industry. The selected studies were analysed qualitatively using content analysis, a scientific method through which a summary and analysis of textual messages is provided, was employed to compare the scaling Agile methods in-depth. Using content analysis, words in a text were directly compressed; and therefore the content categories generated were fewer in number based on the explicit rules of the coding [28]. During the coding, each text segment was labelled and the text segment could range from a few words to a complete paragraph. This coding enabled the rearrangement and integration of words, sentences or paragraphs that were interrelated so that a meaningful portrayal of the data could be generated [28]. As for this study, the outcomes were comparisons among the scaling Agile methods.

III. Results and Discussion

Different Agile methods provide a pathway to scale Agile, especially for large projects. DAD [7], basic LeSS [17],

LeSS huge [17], and SAFe [19] were the main scaling methods [12]. Based on the accomplished review and the analysed studies, the major outcomes resulted as follows:

Several measurable criteria were used to compare the scaling Agile methods (Frameworks) such as project size considering the team involved, training and certification, methods and practices adopted, technical practices and organisational type. Table 2 shows the differences between the methods based on different criteria.

TABLE II
METHODS DIFFERENCES AND SIMILARITIES

Criteria	DAD	SAFe	LeSS 1	LeSS 2	Spotify	Nexus	RAGE
Team size	200 people or more. It also supports small and medium teams.	Large Enterprise includes more than 1 release trains (50 to 124 people in each release trains)	Up to 70 people or 10 SCRUM teams, 7 stakeholders in each team	Any large projects, More than thousand people on one product	Any large projects, Normally 250 to 300 people at Spotify (30 teams)	Three to nine SCRUM teams	No specific size but it support different size for enterprises
Training and certificate on	Workshops to explain the idea of DAD, Book of DAD is available	Training is needed and there should be certified, coaches	Seven companies in six countries are available for coaching	Seven companies in six countries are available for coaching	Lack of training	Scaled Professional SCRUM Training is needed	Training is conducted by webinar and presentation slides
Methods and practices adopted	Kanban Practices (mainly visualizing Work and limiting work in progress), SCRUM (almost all SCRUM practices), Agile Modeling which is the source for DAD's modeling and documentation practices, the Unified Process, XP, TDD and Agile Data.	SCRUM, Lean, Kanban, SCRUMban, DevOps and some practices of XP	SCRUM was fully adopted including additional practices for large projects	SCRUM was fully adopted including additional practices for large projects	Allow Kanban, SCRUM, DevOps and Lean Startup	SCRUM with additional practices in solving the dependency-related issues in multiple teams	Allow SCRUM, Kanban, Plan-Driven development and Hybrid approaches
Technical Practices required	High (Need to adopt many technical practices which require high technical skills)	Medium but should understand the use of portfolio management tools	Medium and low for SCRUM adopters	Medium and low for SCRUM adopters	Medium but teams should be able to communicate well	Medium and low for SCRUM adopters	Medium and low for SCRUM adopters
Organization Type	Multiple Organization and Enterprise practicality	Enterprises and portfolio level	Large Traditional organization	Enterprises	Enterprises specifically similar to Spotify	Portfolio level for medium project	Traditional and Agile Enterprises

A. DAD Description

Based on the comparison in Table 1, DAD extends core agile development (SCRUM, Agile Modelling, Open Unified Process, XP, TDD, and Lean) in order to address the full system delivery, especially for large projects. However, DAD method mainly expands upon the SCRUM construction life cycle in several ways especially at inception phase (initial modelling, form initial team and secure funding), product backlog (defects, feedback for team and training) and explicit transition/release and production phases. Yet, there are many other practices from other Agile methods such as Lean start-up, Kanban, Lean manufacturing and XP [29].

To succeed in adopting DAD, delivery teams must work intensively with enterprise architects, operations engineers, governance people, data management people, and many others. In addition, the technical practices of the team should be high [11] since DAD stresses on the use of functional and data modelling.

One of the main important differences between DAD and other scaling Agile methods is that DAD is not prescribed method [11]. Other methods such as SAFe are prescribed and the solutions they offer are usually limited to restructuring the SCRUM methodology by combining it with traditional software development methodologies in order to resolve those issues [11], [30].

There are a few advantages that can be highlighted in DAD. The first one is it provides guidance in the areas of architecture and design (inception) and DevOps (transition) [11]. Secondly, it is not prescriptive and thusly accords the team with the ability to choose the process as they see fit. Thirdly DAD supports the highest number of lifecycles namely Lean Development, Lean start-up, SCRUM and Continuous Delivery [7], [11]. Next, DAD addresses regulatory compliance and meets CMMI [7], [29], and lastly, it allows different teams to cooperate within the organisation [11], [31]. In spite of that, the reality reveals that the marketplace adoption for DAD is sluggish, rendering the main disadvantage for it.

B. SAFe Description

SAFe includes practices from XP, Lean, Kanban, DevOps and SCRUM [32], [33]. It supports large enterprise, especially when confronting difficulties in adopting the Agile practices [32], [33]. It includes more than 1 release trains, 50 to 124 people in each release trains [34]. The Scaling Agile method aims to combine different Agile and Lean practices in one method to solve different problems in large enterprises [35]. An organization that plans to adopt SAFe should have a team of well-versed participants in the use of portfolio management tools [35]. Therefore, training is needed and trainers should be certified coaches [35].

There are a few advantages that can be highlighted in SAFe. The first one is that SAFe is highly prescriptive and therefore it provides the structure that could ease the transition to an Agile framework [32], [33]. The second, it provides higher ROI and productivity [19], [33]. Meanwhile, it was found that being too prescriptive constrains the continuous development of the teams [11]. Another drawback of SAFe is that it is not within Agile expectation [33].

C. LeSS and LeSS huge Description

Basic Large-Scale SCRUM (LeSS) is SCRUM applied to more than ten teams working together on one product [17]. On the contrary, LeSS huge should be used when there are more than eight teams and it is meant for thousands of people working on one product [17]. Since LeSS was developed based on the idea of SCRUM, therefore through the adoption of LeSS and LeSS huge, user simultaneously incorporates SCRUM with additional practices that assist large organizations to smoothly integrate SCRUM [36].

Seven coaching companies are recommended to conduct training on LeSS which are located in Singapore, Belgium, Finland, Israel, Netherlands, and Germany [37].

Since LeSS is based on SCRUM, the adoption of it as well as rigorous training is minimal [38]. LeSS huge is almost the same as basic LeSS but it is specified for large enterprises of which the participants are in thousands working on one product [38].

LeSS is deemed highly beneficial in two aspects namely ease of adoption [38] and emphasis on SCRUM method improvement in large scale [17], [38]. Meanwhile, LeSS adoption of other practices from other Agile methods is minimal relative to DAD and SAFe [17].

D. Spotify Description

Spotify includes diverse methods based on the needs of different squads. It allows different squads at the team level to use Kanban, Scrum, Lean Startup, and DevOps [20], [22]. In Spotify, there are more than 30 squads where each squad has five to nine members. The squads in Spotify is expected to solve the communication issues between different teams. The main purpose of Spotify is to deal with various teams in a large product development [20]. Large organizations that are interested in adopting Spotify should have a good background of Kanban, Scrum, DevOps, and Lean Startup methods. Moreover, each squad must have access to an Agile coach who has experience in the chosen Agile method [22].

The main advantage of Spotify is the ability of each squad to choose its own working method which eventually will allow large organizations to work better. Spotify also provides much better sustainable developer experience by implementing Chapters and Guilds. However, it also has disadvantages that include the loss of autonomy for feature squads and less technical practices. In addition, the training of Spotify method is also lack [21], [23].

E. Nexus Description

Nexus is defined as Scrum applied to more than two teams but less than nine teams which work together on one product [24]. Nexus was basically developed based on the idea of Scrum (same as LeSS); therefore through the adoption of Nexus, users will concurrently include Scrum with additional Nexus practices that assist two to nine teams to work on one product [24]. However, Nexus is still considered new so that the adoption of it is not as frequent as those of DAD, SAFe and LeSS [25].

The users need to attend intermediate-level Scaled professional Scrum (SPS) training before adopting and implementing Nexus in scaled software and product development. Most of these training are held in Europe countries such as Sweden, Germany, Netherlands, Spain, Norway, United Kingdom, and also in United States [25].

Nexus uses SCRUM as its building block; therefore, one of the requirements to attend the SPS training is at least for teams who have used Scrum [39]. The main advantage of Scrum is to solve the problem of independencies arising between the works of multiple teams working in a large project, and thus Nexus has the same ability to solve the similar issue [40]. For example, when there is an issue in the scope of the requirements that can overlap in a large project.

F. RAGE Description

RAGE is defined as a set of practices and meetings to make the right decision of the project development output [27]. It focuses on solving the problems of large enterprises which use more than one method (SCRUM, Kanban or Plan-driven or Hybrid) in the development of software applications. Since RAGE is still new [26] and based on the output of this research, there are no case studies of the success of RAGE method at the enterprise.

RAGE training is mostly done through webinar and presentation slides since it encourages the organizations to use their own recipes [41]. There is no specific team size for

RAGE but it supports different size for enterprises. The team could follow any method they prefer. Therefore, no specific technical practices are required to enable the adoption of RAGE [27].

IV. Conclusions

Software development teams in large scale enterprises are currently confronting the challenges in delivering high-quality software within arduous time constraints. The challenge in adopting agile methods to develop software increases further with the need to scale these practices to larger settings. DAD, SAFe, LeSS, LeSS huge, Spotify, Nexus, and RAGE are the prominent Agile scaling methods to address these challenges. On one hand, the aforementioned scaling Agile methods seem to be similar since all of them are geared towards solving the problems in large projects. On the other hand, the discrepancies between them take the form of team size, training, and certification, methods and practices adopted, technical practices required and organizational type. Further work will be undertaken by the researchers to empirically investigate how and when to utilize each scaling Agile methods for large Agile projects.

Acknowledgement

We acknowledge the Faculty of Information Science and Technology at Universiti Kebangsaan Malaysia (UKM) for supporting this research under Fundamental Research Grant Scheme project (FRGS/2/2013/ICT01/UKM/02/2).

References

[1] A. Alliance. (2001) "Agile manifesto," [Online]. Available: *http://agilemanifesto.org/*

[2] S. W. Ambler. (2014). *The Non-Existent Software Crisis: Debunking the Chaos Report*. [Online]. Available: http://www.drdobbs.com/architecture-and-design/the-non-existent-software-crisis-debunki/240165910

[3] A. M. M. Hamed and H. Abushama, "Popular agile approaches in software development: Review and analysis," in *Computing, Electrical and Electronics Engineering (ICCEEE), International Conference*, 2013, pp. 160-166.

[4] R. V. Anand and M. Dinakaran, "Popular Agile Methods in Software Development: Review and Analysis," *International Journal of Applied Engineering Research*, vol. 11, pp. 3433-3437, 2016.

[5] Z. Mansor, R. Razali, J. Yahaya, S. Yahya, and N. H. Arshad, "Issues and Challenges of Cost Management in Agile Software Development Projects," *Advanced Science Letters*, vol. 22, pp. 1981-1984, 2016.

[6] R. H. Al-Ta'ani and R. Razali, "Prioritizing Requirements in Agile Development: A Conceptual Framework," *Procedia Technology*, vol. 11, pp. 733-739, 2013.

[7] S. W. Ambler and M. Lines, *Disciplined agile delivery: A practitioner's guide to agile software delivery in the enterprise*, Indianapolis, Indiana: IBM Press, 2012.

[8] K. Dikert, M. Paasivaara, and C. Lassenius, "Challenges and Success Factors for Large-Scale Agile Transformations: A Systematic Literature Review," *Journal of Systems and Software*, vol. 119, pp. 87-108, Sep, 2016.

[9] D. J. Anderson, *Kanban: Successful Evolutionary Change for Your Technology Business*, D. G. Reinertsen., Foreword, Washington, USA: Blue Hole Press Sequim, 2010.

[10] K. Beck, *Extreme programming explained: Embrace change*. Boston, USA: Addison-wesley professional, 2000.

[11] S. W. Ambler and M. Lines, "The Disciplined Agile Process Decision Framework," in *International Conference on Software Quality*, 2016, pp. 3-14.

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

[13] I. Jacobson, I. Spence, and P. W. Ng, "Is there a single method for the Internet of Things?," in Ivar Jacobson International, 2016.

[14] V. T. Heikkilä, M. Paasivaara, K. Rautiainen, C. Lassenius, T. Toivola, and J. Järvinen, "Operational release planning in large-scale Scrum with multiple stakeholders–A longitudinal case study at F-Secure Corporation," *Information and Software Technology*, vol. 57, pp. 116-140, 2015.

[15] M. Paasivaara and C. Lassenius, "Scaling scrum in a large distributed project," in *2011 International Symposium on Empirical Software Engineering and Measurement*, 2011, pp. 363-367.

[16] S. W. Ambler and M. Lines, "Going Beyond Scrum: Disciplined Agile Delivery," *Disciplined Agile Consortium. White Paper Series*, 2013.

[17] C. Larman and B. Vodde, "Scaling Agile Development," *CrossTalk*, vol. 9, pp. 8-12, 2013.

[18] C. W. Larman, Matt (2014) Large Scale Scrum (LeSS) @ J.P. Morgan. [Online] Available: https://www.infoq.com/articles/large-scale-scrum-jomorgan

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

[20] H. Kniberg and A. Ivarsson, (2012) "Scaling agile@ spotify with tribes, squads, chapters & guilds," Available: *h*ttps://ucvox. files. wordpress. com/2012/11/113617905-scaling-agile-spotify-11. pdf

[21] V. Venkatesan. (2015). Scaling Agile Using Spotify's Framework. [Online]. Available: https://www.scrumalliance.org/community/articles/2015/december/scaling-agile-using-spotify-s-framework

[22] I. Lunden. (2012). Here's How Spotify Scales Up And Stays Agile: It Runs 'Squads' Like Lean Startups. [Online]. Available: https://techcrunch.com/2012/11/17/heres-how-spotify-scales-up-and-stays-agile-it-runs-squads-like-lean-startups/

[23] S. Marcus, "Growing Up Spotify," in Qcon San Francisco, 2014.

[24] K. Schwaber. (2015). Nexus is the exoskeleton of SCALED SCRUM. [Online]. Available: https://www.scrum.org/Resources/The-Nexus-Guide

[25] S. org. (2016). *Learn how to Scale Scrum with Nexus.* [Online] Available: https://www.scrum.org/Courses/Scaled-professional-Scrum

[26] K. Thompson. (2013). Recipes for agile governance in the enterprise. [Online]. Available: https://www.cprime.com/resource/white-papers/recipes-for-agile-governance-in-the-enterprise/

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

[28] K. Krippendorff, *Content analysis: An introduction to its methodology*, 3rd ed. Los Angeles, USA: Sage Publications, 2012.

[29] S. W. Ambler, IBM agility@ scale™: Become as agile as you can be, *IBM Global Services*, New York, USA: IBM Corporation USA, 2010.

[30] K. Crowston, K. Chudoba, M. B. Watson-Manheim, and P. Rahmati, "Inter-team coordination in large-scale agile development: A test of organizational discontinuity theory," *Differences*, vol. 19, pp. 21, 2015.

[31] M. Lines and S. Ambler. (2014) Implementing Disciplined Agile Delivery (DAD) at Panera Bread: A Recipe for Success. [Online] Available:https://www.agilealliance.org/wpcontent/uploads/2015/12/ExperienceReport.2014.Lines_.pdf

[32] J. M. Bass, "Artefacts and agile method tailoring in large-scale offshore software development programmes," *Information and Software Technology*, vol. 75, pp. 1-16, 2016.

[33] I. Stojanov, O. Turetken, and J. J. Trienekens, "A maturity model for scaling agile development," in *41st Euromicro Conference on Software Engineering and Advanced Applications*, 2015, pp. 446-453.

[34] R. Brenner and S. Wunder, "Scaled agile framework: Presentation and real world example," in *Software Testing, Verification and Validation Workshops (ICSTW), IEEE Eighth International Conference*, 2015, pp. 1-2.

[35] A. Pancholi and S. Grover, "Scaled agile framework: A blight," *International Journal of Innovative Research and Development*, vol. 3, pp. 425-427, May, 2014.

[36] C. Larman and M. Winn. (2014) Large scale scrum (LeSS) @ J.P. Morgan. [Online]. Available: https://www.infoq.com/articles/large-scale-scrum-jomorgan

[37] C. B. Larman, Vodde. (2014) Coaching companies. [Online] Available: http://less.works/coaching/coaching-companies.html

[38] V. Krishnamurthy. (2015) What is LeSS (Large Scale Scrum). [Online]. Available: http://www.tabar.com.au/blog/2015/1/13/what-is-less-large-scale-scrum-
[39] C. Ramos. (2015) Scaled Professional Scrum® Training. [Online]. Available: https://agilix.nl/scaled-professional-scrum-training
[40] P. Kong (2015) When do you use Nexus? [Online] Available: www.Scrum.org
[41] K. Thompson. (2014) How to Develop Your Own Recipe for Agile Governance. [Online]. Available: https://www.cprime.com/2014/06/how-to-develop-your-own-recipe-for-agile-governance/