Quality Deployment and Use of the Scaled Agile Framework® - Managing teamwork and software quality in the banking sector

Priya Mishra

University of Tampere
Faculty of Natural Sciences
MSc Degree Programme in Software Development
Thesis Supervisors: Dr. Eleni Berki, Dr. Juri-Petri Valtanen
Work Supervisors: Teemu Lepistö, Tomi Suomela
MSc. Thesis, 61 pages, June 2018

University of Tampere
Faculty of Natural Sciences
MSc in Software Development
Priya Mishra: Quality Deployment and Use of the Scaled Agile Framework® –
Managing teamwork and software quality in the banking sector
MSc Thesis, 61 pages, 5 index and 2 appendix pages
June 2018

Abstract

Agile software development method(ology) has recently been an increasingly popular research and development trend. Many, mainly small and medium size enterprises, have adopted agile development approaches and methods, hoping to i) solve organisational and developmental shortcomings that possibly derive from the traditional development methods use and ii) obtain better business outcomes. However, large enterprises have been hesitating in the use and deployment of agile methods.

Scaling the agile is a term for customising agile software development practices for large scale enterprises. *Scaling the agile* is another emerging concept, which, in recent years, seems to become another attention-grabbing trend by agile experts and practitioners. Hence, this MSc thesis has had the ambitious target to fill in the research and development gap and address the need for an improved understanding, In so doing, the thesis' contents comprise an overview and a survey on the application and applicability of scaling agile frameworks and, in particular, the "Scaled Agile Framework®" or "SAFe®" in the *banking domain*. SAFe® is a brand name for a well structured agile framework praised for i) supporting *teamwork* and ii) incorporating *agility* in large scale organisations, such as in the financial and health sectors.

This research case study on scaling agile for quality banking software is conducted for the first time. The study utilises a survey, wherein the participants/respondents are team members of the Nordea Bank working for agile software development projects, which mainly target to digitalise banking practices and services. The purpose of the survey has been to identify key success factors and main challenges considered by the team members who use the "Scaled Agile Framework®" or "SAFe®" environment, in everyday software practice.

In particular, the research results of this study can, first, be utilised by the Nordea Bank's software project management teams for *refactoring* current practices and, thus, achieve software process improvement by focusing on modern software quality-related knowledge. Moreover, business organisations and software practitioners that are in the process of moving towards agile or exploring new agile methods can be benefitted from this study's outcomes, because they can largely serve as a set of teamwork guidelines to keep with work-life quality in mind. Overall, the survey results that are documented and

analysed in this thesis could serve as a new, original information and knowledge repository for software developers, project managers and researchers, who aim at applying and investigating SAFe® and/or other "scaled agile" frameworks in industrial settings.

Keywords and terms: Scaled Agile Framework® or SAFe® ***(SAFe and Scaled Agile Framework are registered trademarks of Scaled Agile, Inc.")***, PI Planning Event, Scaling Agile, Agile Software Quality Improvement, Agile Software Teams, Banking.

Contents

1. Introduction

Our era is dominated by ICTs and software, which aim towards the enhancement of work-life quality. Ranging from simple mobile applications to create notes to complex applications involving banking transactions, most of the manually operated systems have been replaced by software applications. The customers are easily able to find multiple options in the market to fulfill their requirements. As a result, the market has become highly competitive and the companies are bound to deliver high quality and user friendly software. The method(ology) used to develop the software plays a major role in determining the overall quality as well as the acceptance, success or/and failure of a software product.

In the 1960s, the computer softwares developed were targeting a smaller customer base and had rather limited scope [Highsmith, 2002]. Accordingly, the software design was rather primitive and personalised while no authorised approach was followed for the software development teams and the software process management. Later on the software process methods became more scientific and disciplined through the use of metamodels, evaluation frameworks and associated toolsets [Berki et al., 2004]; but it also became more complex and meant to serve a huge customer base. As a result, traditional software development methods and processes were formulated and used to organise and conduct software development in an efficient manner [Highsmith, 2002]. Nevertheless, most traditional development methods and respective evaluation frameworks and metamodels required extensive documentation and a rigid set of activities to be followed [Berki, 2006]. The stakeholders were involved only during the initial phases when requirements were identified and finalised. There was no plan for requirements modification at the later software development stages and the stakeholders (end-users in particular) were hardly involved during the development phases.

To overcome the pitfalls of traditional software development, “agile development” was introduced in 2001, though informal sporadic practice existed earlier. The question though remained as ‘agile quality or depth of reasoning?’. Accordingly, a method’s applicability versus suitability with respect to stakeholders needs [Berki et al., 2007)]has been a discussion topic for most agile methodologies. The Agile Manifesto succeeded to formulate 12 principles which have been at the core of every agile software development process [Beck et al., 2001]. The core principle of agile software development is to ensure *flexibility* (agility) in every stage of the software development. The involvement of customers, flexible requirements and delivery of working software at regular intervals are the key (success) factors, which make the agile process a powerful approach and one of the best available options to develop a software product/ service.

There are a number of agile oriented development methods which are used in software process industries. Some of them such as *eXtreme programming* and *Scrum* are popular in practice and because of the use of metamodelling principles [Kunwar,

2013] and evaluation frameworks. This study focuses on yet another agile software development process the so called "Scaled Agile Framework®" or "SAFe®" [SAFe®/ Scaled Agile Framework®]. The Scaled Agile Framework® is built on the top of the agile methodology resulting in harnessing all the positive factors of agile development. Moreover, it is a combination of *lean agile* development in *systems practice* and *systems thinking* [Checkland, 1999] in its theoretical foundations. According to Dean Leffingwell, the creator of the framework, the Scaled Agile Framework® helps businesses to address the significant challenges of developing and delivering enterprise class software and systems in the *shortest sustainable lead time*. It is tailored in a way that this type of agility can support small enterprises [see also Soloveva, 2017] driven by 50-150 people and can also be scaled to support a large enterprise driven by thousands of team members [Knaster and Leffingwell, 2017].

1.1. Introducing the Scaled Agile Framework®

The Scaled Agile Framework®, commonly referred to as SAFe®, was introduced by Dean Leffingwell [Leffingwell, 2016]. It is a well structured framework and it is based on the principles of *lean agile methodology* of software development. The basic idea of SAFe® is to provide a detailed model, which could be implemented to deliver and develop enterprise class solutions within a dedicated time frame. Moreover, it incorporates the power of the agile approach and takes it one step higher to overcome the challenges faced by large scale organisations. Yet, it can also fit into a small scale organisation employing 50-125 practitioners [Leffingwell, 2016].

Developing an enterprise level software solution could be a complex process. Hence, it is essential to follow a full proof approach in order to establish proper coordination among the team members. In large scale projects it is a general practice to have a common scenario for the teams in order to have inter-dependencies amongst them and alleviate any lack of knowledge about relevant contact persons and customer involvement issues. Hence, a proper mechanism is essential to establish coordination among the teams and deliver business value within dedicated time. The SAFe® process metamodel approach provides synchronisation among the agile teams and provides essential solutions to overcome the challenges faced by the large scale enterprises.

As explained by Knaster and Leffingwell [2015], the key considerations, which are handled by the application of the Scaled Agile Framework®, are the following:

1. How to align the enterprise towards a common goal and how to improve the economic outcomes?
2. How can the solution be delivered in dedicated time frame and ensuring it is a quality solution making customers happy?
3. How to overcome the bureaucratic hierarchical team structure and organise agile teams?

4. How to ensure motivation, innovation and collaboration among teams and assist them towards handling risks and continuous learning?
5. How to measure that the process being deployed is providing the desired outcome?

The creators of SAFe® strongly praise numerous benefits such as up to 50% increase in productivity, more than 50% increase in time to market and quality and measurable increase in employee engagement and job satisfaction [Leffingwell, 2016]. In recent years, many enterprises have adopted the SAFe® approach as a strategy and have gained valuable outcomes. Furthermore, these enterprises are not only IT giants but act as organisational domains in various sectors such as electronics, healthcare, financial and government. Some highlighting examples are the Nordea bank, Fitbit, Amdocs and Cisco organisations/companies.

1.2. Research Rationale

Many notable enterprises have made transitions from the traditional practices towards modern methods such as the SAFe® software process metamodelling framework. Nordea, one bank that is considered as one of the leading financial groups in Northern Europe, has adopted the Scaled Agile Framework® in order to refine the digital experiences [Nordea Case study, 2017].

Numerous case studies were conducted in the past targeting to analyse and relate software quality management to the efficiency and fitness of the software development methodologies evaluation [Berki, 2006]. Many researchers identified success factors and challenges posed by the methodologies application in the software development industry (see e.g. Berki et al., 2004). Agile methods and their consequences in particular were studied by many field researchers such as Dikert et al. [2016], Chow & Cao [2008] and Misra et al. [2009]. Notwithstanding, Dikert et al. [2016] stated that agile methods have been explored from various perspectives but scaling agile frameworks such as SAFe® are yet to be explored thoroughly.

For example, the agile software team members, while working by using a specific development strategy, they also face certain challenges and would like to improve certain development work (sub)processes. Moreover, there may be positive factors associated with the method which are preferred by most of the practitioners. The identification of these factors could reveal such strategic information management knowledge that could be used to rectify current processes for increased developers satisfaction, better business outcomes and overall developmental efficiency, which could gradually lead to continuous process improvement.

This research study was conducted in the Nordea bank's premises and the participants were members of a project working towards digitalisation of the banking practices. The broad objectives were to identify the i) potential challenges of the agile

process in the software teams and ii) key enablers of the Scaled Agile Framework® among team members.

For this study, the participants chosen were *developers, Scrum masters, product owners, testers* and *designers*. Online questionnaires were created for each set of the above stated.roles. In return, the questionnaire assisted enormously to identify positive factors and challenges associated with the SAFe® work environment. In addition, ideas to overcome the challenges and improve the software development process were also gathered from each member's perspective.

1.3. Further Research Objectives and Research Questions

The SAFe® framework is quite new as compared to other agile methodologies such as eXtreme Programming and Scrum. According to the literature review carried out by Dikert et al. [2016], agile methods have been explored from various perspectives in recent years; however, SAFe® is not explored thoroughly and provides wide scope for researchers to study it further. The thesis author is currently employed as a software developer at Nordea Bank, dealing with one of the development projects running on Scaled Agile Framework® version 4.0. Therefore, practically experiencing the methodology and implementing the practices are the key reasons for personally selecting the Scaled Agile Framework® as the research topic and eagerly committing to this study.

Hence, the main objectives have been to identify the challenges and the enabling factors for team level members of SAFe®. Stakeholders' feedback was also planned to be captured, especially regarding their own ideas to improve the development framework itself. Another objective was to conduct, a broad literature review to get an essence of related work in "scaled agile", in industrial practice in particular. The latter revealed that there exist other available methodological frameworks in the market, which target to the scaled agile subject area. Such are the DaD (Disciplined agile Delivery) and LeSS (Large Scale Scrum) approaches/frameworks. These popular scaling frameworks are also discussed briefly as a part of the literature review.

This thesis work will, eventually, be useful for researchers who would like to achieve an improved understanding of the metamodelling principles and guidelines of the scaled agile method(ology). It could also be utilised by various organisations to identify the challenging factors and improving the software process while adopting SAFe®. Since this involved a personal motivation, the thesis author was also assisted in understanding the process clearly and was reassured that she herself could perform the tasks more efficiently as a team member in an organisation that decided to improve their software process running on the wheels of the vehicle offered by the Scaled Agile Framework®.

In this thesis the following research questions were explored:

1. What are the success factors of Scaled Agile Framework® at team level?

2. What are the challenging factors of Scaled Agile Framework® at team level?
3. What could be possible improvement suggestions for the betterment of the SAFe® process at team level?

This is the first time that these research items are investigated in the banking sector's software process. In so proceeding, different concepts of software quality and agility will be reviewed and discussed by the members of agile software teams. In related work exposed in past research studies investigating traditional methods, quality concepts were finally developed and presented as extensions of new knowledge. For instance, a similar research study [Eriksson and Törn, 1991] aimed at a division of quality concepts consistent with the different decision makers and decisions made during the software life-cycle. The main division was into: cost effectiveness, use quality and work quality. The last two were further divided into: requirement quality, interface quality, and efficient IS management, evolution quality and operation quality, respectively [Eriksson and Törn, 1991].

1.4. Research Methodology

In order to adequately answer the research questions stated above, it was decided that a "survey" should be the most suitable research method for this case study and thus was used for this thesis' work. The power of a survey, as a research method, is that resembles the capturing of snapshots of a given situation in a particular environment by utilising structured interviews or questionnaires. The data captured in this way can further be analysed and significant results can be derived [Galliers and Land, 1987] upon data processing. Since what mattered in this study was the research object that fits to a survey method, a questionnaire was constructed with questions that were related to gathering experiences of people who have been using the Scaled Agile Framework® for agile software development projects. The thesis author currently works in a branch of the Nordea Bank, which is considered as one of the major domains aiming for better digital experiences and for this reason the software development stakeholders have started using the SAFe® framework. This presented an opportunity to conduct a survey for practitioners using SAFe® processes for their work on a daily basis.

The complete questionnaire was prepared by structuring the appropriate questions with the help of an online tool, called *"*SurveyPlanet*"* [SurveyPlanet, 2017]. Interviewing the stakeholders individually proved to be a difficult option in this scenario as the participants were having busy work schedules and not everyone was co-located. Hence, for this study, an online questionnaire was the best possible way to distribute and collect the data in an easy, reliable and timely manner. Among multiple online questionnaire tools available, SurveyPlanet tool was chosen because it provides a simple and user friendly interface. This tool was thoroughly used to design, distribute and capture the survey results. In particular, five sets of role-specific questionnaires

were prepared targeting team level members in an agile development project using SAFe® at Nordea Bank.

Furthermore, along with the survey, *observation* was used as a methodological technique and thinking tool to discuss and provide final conclusions in a reflective and consistent manner. The complementary use of observation as a research method implied the collection of information by way of the investigator's (thesis' author) own observations in documentation, without interviewing the respondents. The obtained information relates to what is currently happening and is not complicated by either the past behaviour or future intentions or attitudes of respondents [Kothari, 2008]. As a team member working within a Scaled Agile Framework® the thesis author thinks that she was presented with a unique opportunity to observe and understand the implications of the 'applied agility' on daily development activities of the scheduled teamwork.

1.5. MSc Thesis Structure and Organisation – An Overview of the Thesis Work

In summary, the contents of this thesis are organised and presented in the current and next chapters as follows:

1. Chapter 1 presents an introduction to the thesis, followed by an overview of the Scaled Agile Framework®. Furthermore, research rationale, research objectives, research questions and research methodology used for this study are stated and a draft layout of the thesis informs the reader of the summarised contents of every chapter.
2. Chapter 2 establishes the background information used for this study. A brief overview of the traditional, agile and lean agile development methods are discussed in the initial section. This is followed by a detailed overview of the Scaled Agile Framework® (SAFe®).
3. Chapter 3 presents the details regarding SAFe® application and implementation in one of the recent projects at Nordea Bank.
4. Chapter 4 presents how the research methodology and data collection method were used in detail and describes the steps and deliverables of the research process.
5. Chapter 5 presents and analyses the results of the survey in a reflective and critical manner.
6. Chapter 6 includes final discussion, conclusions of the study, limitations and suggestions for further research and development work based on the research outcomes of this study.

2. From Traditional Development to (Scaling) Agile

2.1. Traditional Software Development- How *Rigid* and *Rigorous* Could it be?

The processes (and further subprocesses) of the software development (lifecycle) play a critical role in the success or failure of the quality of the final product or service. Until mid-nineties the traditional software development methods worked fine as the software was lower in complexity than today's software, with also less competition in the market and less possibilities of changing requirements [Li, 2012]. However, the era of digitalisation in the last 10-20 years marked significant changes in the software development field. According to Beck [1999], the traditional methods follow a definite structure with following practices, as stated next:

- All the phases including planning, design, development, testing and maintenance are segregated and include almost no overlap.
- Documentation is an essential component of almost all traditional development practices.
- The requirements are solidified and documented in the beginning of the life cycle and there is absolutely no scope of accepting any changes in later phases.
- The customers are initially involved in requirements gathering but in later phases of development customers rarely participate.

There were certain benefits of these methods such as proper division of work and roles in each phase, detailed documents and each phase of development was responsible to produce certain output used as input for the next phase(s). Despite of these advantages, many of the rigid practices followed by traditional development approaches resulted in software failures. Many studies such as Li[2012] and Glass[2001] identified the main reasons of software (system) failures, and these are, hereby, summarised as follows:

1. *Changing requirements*. In conventional development methods the customers are involved in the development during the very initial phases. The assumption has often been that the requirements will be static throughout the life cycle of the project. In this era of rapidly changing technology and competitive market, it is hard to stick to the same set of requirements throughout the project's lifecycle. Moreover, the customers demand to be able to provide developers with feedback and add/modify/delete certain requirements depending on their preference even at later stages. This power to the customers is completely lacking in most conventional development methods and hence it is a major concerning reason for software failures.
2. *The decision makers*. The deals with the customers/end-users and any crucial decision making is mostly done by middle and top level management. Additionally, there is a rigid hierarchical project management structure which

can be a root cause for bureaucracy. The development project team involves designers, developers, testers and other support members but decision making usually involves only the management. As a result, the decisions may not be appropriate for or accessible to the whole team and may result in chaos within the work team and overall failure of the software project.

3. *The traditional practices.* The project activities are often divided into non-overlapping phases. For instance, the quality assurance (e.g. testing, walkthroughs, inspections …) mostly comes into picture only after the development work is done. There are notable exceptions to this but, in general, when the design/specification testing or/and validation and verification with formal methods are not supported the software errors will be propagated to the later stages without being identified and fixed early enough in the lifecycle.
4. *Roles and responsibilities.* The roles and responsibilities of each team member are fixed and limited. For instance, the developer/coder is only responsible for coding and does not have any opinion during planning and design phases nor product (code) visibility. As a result of these rigid lines of division of responsibilities, the project could lead to a failure.
5. *Lack of value*. The conventional approaches involve excessive documentation, which may not be useful for customers but rather a documentation of the process formality. Thus, prioritising on an outcome of less value over real deliverable can be essentially considered as a waste of efforts and resources.
6. *Rare customer involvement*. The customers are normally involved in initial phases only and soon afterwards the requirements are documented. Thus, the customer has almost no clue of the actual result that is to be delivered later. There could be possibilities that the customer's expectations and the actual outcome(s) differ in reality. This can result in a chaotic situation between the customer and software organisation, which in turn could also lead to a project failure situation. Figure 1 represents an example of how customers expectations and final product(s) delivered could differ due to the lack of frequent customer involvement and other stakeholders' communication and collaboration during all the development phases.

Figure 1. Customers' expectations (left) versus actual resulted deliverable(s) (right)

2.2. Agile Software Development – How *Different* and *Flexible* is it?

Beck et al. [2001] stated that the turn to agile software development is the reaction to the pitfalls of traditional software development practices, and they further acknowledge the need for an alternative to documentation-driven, heavyweight software development processes. The concept of agile was first introduced in 2001 and since then many software development companies deviated towards agile. The agile manifesto formulated by some experts presents the principles and core values which formulate the backbone of agile software development [Beck et al., 2001]. Nevertheless, agile practices can be viewed even earlier in industry but were just named differently. For instance, a mixed methodology with flexible application was followed by considerable intellectual efforts for better method and tool integration [see e.g. Berki et al., 2007].

The basic values given in agile manifesto formulated by Beck et al. [2001] are:

1. Individual and interactions over processes and tools
2. Working software over documentation
3. Customer collaboration over contract negotiation and
4. Responding to change over following a plan.

It is interesting to relate the challenges associated with traditional development and evidence how agile values overcome these challenges. The competitive and rapidly changing market conditions in recent years support the necessity of agile software development. The agile methodology paradigm realises that projects are inherently unpredictable and thus welcomes the changing requirements throughout the development lifecycle [Taymor, 2013]. Taymor [2013] lists the highlighting reasons why agile is better over traditional development methods, as follows.

1. *Speed to market*. Agile supporters believe in iterative development and delivering values in the form of working software after each iteration. In this way the product reaches quickly to the intended users.
2. *Flexibility*. The agile manifesto turns into welcoming changes during the development cycle rather than locking the scope during the initial phases. This technique is essential to incorporate the rapid changing market needs and customer's feedback.
3. *Risk Management*. The incremental releases offer the platform to identify and remove the risks and faulty issues earlier.
4. *Cost Control*. Agile development is flexible enough so it does not need to stick to rigid budgets but rather be based on customer's feedback on the requirements handling; budget associations can be altered at any phase.
5. *Quality Assurance*. Agile methodology ensures high end quality because of the practices involved during the development. Agile-driven software is based on test driven development and thus every member is responsible to maintain the quality. The developers integrate quality assurance along with the development

and this ensures that bugs are identified and fixed in earlier stages. Other practices such as pair programming, continue integration and code reviews add layers of quality on the end product.

6. *Right product.* The customer collaboration provides a way for customers to test the product during the development phase and ensures that the product is shaped towards the right direction.

2.3. Lean Agile Development – Is it a Real *Alternative*?

The term "Lean" was initially coined by Toyota Corporation around 1950s. According to Poppendieck and Poppendieck [2003] the lean thinking has a long history of improving significantly the fields of manufacturing, health care, and construction. Shalloway et al. [2010] stated that the lean principles provide an environment where managers and developers can work together towards a common vision, providing best returns on development efforts. Additionally Scrum, one of the popular agile methodologies, is also based on lean principles. Poppendieck and Poppendieck [2003] wrote an interesting book which presented a toolkit on deploying the lean principles into agile environment and improving the effectiveness of the development process. The Scaled Agile Framework® also inherits lean principles at its core. As stated in the book by Poppendieck and Poppendieck [2003], the key principles of lean software development approach are as follows:

1. *Eliminate Waste*. If there is any feature, deliverable or activity in a project which is not adding value to customer and still utilising resources and time should be identified and eliminated from the cycle. Defects, unnecessary features, documentation and task switching are examples of waste in the context of lean agile development.
2. *Built in Quality*. The quality proofing should not be left for the last phases of development, but rather every phase should involve software quality checks. The developers should perform unit testing during coding phase so that less number of bugs would propagate further. This will ensure better software quality and would prevent major surprises during the testing phases.
3. *Create knowledge.* Every iteration should be followed by retrospective discussing on pros and cons of the iteration and creating knowledge base. This leads to i) minimising similar mistakes in the future and ii) following the enabler activities throughout the life cycle.
4. *Deliver fast*. Instead of planning exhaustively and delivering the product after months or years of work, keep planning for short spans and keep delivering in short batches.
5. *Respect People*. All the team members involved in the project should be empowered. The work environment should not be biased towards comforting

only the top management members. There should be transparency and coordination among all the team members. Learning an innovation should be an essential element of the project lifecycle.

6. *Optimise the whole*. The complex system can further be broken into several subparts. Each subpart fulfils certain responsibilities. It is essential that each subpart performs individually but also that all the subparts collaborate effectively. The *system as a whole* needs to be optimised.

2.4. The 'Scaling Agile' Concept – Could it be a/the *Silver Bullet?*

Agile software development has rather been a popular research and development area in recent years. Mostly organisations prefer a *method(ology) with agility* to traditional development methodologies in order to thrive in the competitive market. *Scaling the agile* is an emerging concept, which has gained interest among agile experts in recent years [Laanti, 2014]. When the agile paradigm was introduced, it was coined to be as a perfect fit for small scale organisations [Bohem & Turner, 2005]. According to Williams and Cockburn [2003] *"agile value set and practices are best suited for colocated teams of about 50 people or fewer who have easy access to user and business experts and are developing projects that are not life-critical"* [Williams and Cockburn, 2003]. In recent years, large organisations have also started to use agile methodologies. The idea of *tailoring and remodelling agile practices* to make it *suitable for the large* organisations is called *"scaling"*.

The term 'large-scale agile development', has been used to describe the agile development in everything: from large teams to large multi-team projects to making use of principles of agile development in a whole organisation [Dingsøyr & Moe, 2014]. There can be several (software quality) factors which can be responsible for making the project large scale such as number of lines of code, number of teams and the geographical project distribution. It is challenging to establish a common understanding of the "scale" factor in an organisation [Dingsøyr & Moe, 2013]. In the research by Dingsøyr and Moe [2013] a taxonomy was recently established to clarify the meaning of "large" in terms of agile development. The broad taxonomy presented in Table 1 refers to three types of divisions of agile projects, which are: small scale, large scale and very large scale. This provides an understanding of the factors that essentially group the project on different scales.

Level	Number of teams	Coordination approaches
Small- scale	1	Coordinating the team can be done using agile practices such as daily meetings, common planning , review and retrospective meetings.
Large- scale	2-9	Coordination of teams can be achieved in a new forum such as a scrum of scrums.
Very large scale	10+	Several forums are needed for coordination, such as multiple scrum of scrums.

Table 1. A taxonomy of scale of agile software development projects [Dingsøyr & Moe, 2013]

Incorporating agile or scaling agile in a large scale project is a complex process. There can be various factors, which can cause hindrances in scaling the agile practices in an organisation. An interesting research study by Ambler [2007] revealed some possible factors, which can cause complexities while scaling an agile project. While transitioning towards scaled agile the organisations could refer to these factors and pre-plan to manage smooth process flow. The factors are listed next:

1. *Team Size.* The team size would affect the working and coordinations strategies. The strategies for small colocated teams will be different for teams with several people working together.
2. *Geographical distribution.* The co-located teams working and coordination style will be different than the teams which are distributed in different locations.
3. *Entrenched culture.* The larger organisations can have people from different backgrounds, standard policies and processes which may not be positive for facilitating an ideal agile development environment.
4 *System complexity.* A large scale organisation has a complex system and thus requires a viable architectural strategy.
5 *Legacy systems.* The pre existing code, databases and documentation may not be ideal for agile development. They cannot be fully replaced and (re-)written from scratch in many scenario cases.
6. *Regulatory complaisance.* There could be some regulations and legal processes in large scale projects resulting in increased documentation.
7. *Organisational distribution.* The team members can be working as consultants, freelancing or from different companies. This increases organisational complexity and team disintegration chances.

The summary of a workshop conducted at XP2014 conference presented an interesting discussion on large scale agile development. The primary focus areas of the workshop were: i) the role of architecture, ii) inter-team coordination, iii) portfolio

management and iv) scaling agile practices [Dingsøyr & Moe, 2014]. The summary presented the following conclusions based on these four factors:

1. *Architecture.* Architecture plays a key role in defining how the work is coordinated in large scale projects. The organisation of architecture gets affected by the level of changes and uncertainty.
2. *Inter team coordination.* It is essential for teams to coordinate effectively within the team level and also outside the team level. Thus, there should be a common set of norms and values for facilitating smooth inter team coordination and an effective knowledge network for collaboration outside the teams.
3. *Portfolio management.* Agile portfolio management is crucial as it handles several dependent projects within one portfolio. The requirements of any project can keep evolving and thus affect the overall portfolio level. In order to facilitate smooth portfolio management, there should be continuous communication and feedback mechanisms between portfolio level and project level.
4. *Scaling.* It is essential to establish an understanding of the "scaling factors", which make the project of large scale. *Scaling must happen both at team level and system engineering level.*

2.5. Existing Scaled Agile Models

This thesis focuses on issues related to Scaled Agile Framework, which is a process (meta)model [Berki, 2006] based on agile principles and tailored to scale for large scale organisations. There are other frameworks/(meta)models available that harness similar concepts of providing agility for large scale organisations. The following sections contain details and comments of the analysis of some popular scaled agile models of recent use and deployment.

2.5.1 Disciplined Agile Delivery(DAD)

The disciplined agile delivery is a process framework for scaling agile practices for distributed agile projects. This is a hybrid approach which extends Scrum utilising proven software development practices such as agile modelling, eXtreme programming and other methods. The speciality of this framework is that it overcomes the shortcomings of Scrum and provides a full structured approach from project initiation to releasing the product in customer's hands [Ambler, 2012]. In more exact words, "the Disciplined Agile Delivery (DAD) decision process framework is a people-first, learning-oriented hybrid agile approach to IT solutions on delivery. It has a risk-value delivery lifecycle, is goal-driven, is enterprise aware, and is scalable". The Disciplined Agile Delivery was initially created by IBM, but nowadays it can be accessed and used freely through online available content. The best feature of DAD is that it does *not*

provide a rigid version and set of practices; it rather provides multiple lifecycle models. The idea is that every organisation is unique and a single lifecycle could not be suitable for all types of organisations. Thus, depending on the need of the organisation, the corresponding delivery model could be tailored [Ambler, 2013].

As stated by Ambler [2012], some common characteristic features of the Disciplined Agile Delivery framework are as follows:

1. *People first*. The teams in DAD environment should be self disciplined, self-organising and encouraged to expand their skill set to work in cross functional teams.
2. *Learning-Oriented*. The DAD framework promotes a learning environment within the teams so that every team member is well aware of the requirements they are working on, and be specialised in tools and techniques they are using to create the solutions.
3. *Agile and Hybrid*. The base of DAD is a set of agile principles and it further enhances it to address and solve the challenges associated with agile practices. It forms a mixture of several agile strategies such as scrum and extreme programming.
4. *Solution focused*. The DAD approach tends to provide not only working pieces of software but also focuses on full sets of solutions for customers. These solutions may, for instance, include upgraded hardware or necessary documentation related to the solution.
5. *Planned lifecycle*. The DAD framework addresses the project from scratch to the point where it is released to the customers. All the phases including the requirements phase, modelling, risk management and deployment are carefully planned within the project's lifecycle. The whole lifecycle of DAD is divided into phases and each phase responsible for certain activities. Between the conception phase and delivery, the development activities are performed. The first phase is conception *phase,* where the project framing initiates. The high level activities such as building the vision and deciding architecture are performed to lay a proper roadmap for the project to evolve. Next, in *construction phase*, the teams start working towards constructing the solution and delivering values at regular intervals. Finally, the *transition phase,* involves the deployment of the solution in the market and the provision of any necessary support.

2.5.2 Large Scale Scrum(LeSS)

The LeSS (Large scale framework) is yet another popular framework for scaling agile practices for large scale organisations. There are two frameworks available at the moment namely *smaller LeSS* and *LeSS Huge*. According to Craig and Bas [2017],

there are many organisations from various domains such as telecom, banking, trading systems and gaming, which have adopted and succeeded with LeSS framework. The smaller LeSS is suitable for projects which have up to ten Scrum teams working towards a common product. There is only *one product owner* (PO) for all the teams. In the sprint planning sessions, two members from each scrum team and the common product owner participate and perform the planning. The scrum teams further perform planning and daily stand ups for coordination and tracking of the development. The sprint review meeting is held similar to first sprint planning in presence of PO and two members from each team. The LeSS Huge framework is similar to the smaller LeSS framework, but it can be visualised as a scaled version for large scale projects. In this case the team members work on a dedicated product(s) area which is/are owned by area product owners. There is an overall product owner who works on top of all the area POs and establishes coordination between all the area teams.

2.6. Communication and Coordination in Agile Development Teams

In the agile manifesto, one of the key agile values is "interactions over processes" [Beck et al., 2001]. Agile development ensures constant communication and knowledge sharing among the team members. The team members interact more often with each other as well as with the management in order to track project development and share knowledge. Communication not only occurs face to face but meetings are also an essential component of the agile development cycle. The popular agile methods such as Scrum include daily meetings, sprint plans, sprint retrospective and review meetings as a part of an iteration. In general, communication is an *essential* activity for *daily* project development.

Studies conducted in the past in the agile projects context highlighted the importance of communication for effective project coordination. According to Williams and Cockburn [2003], face to face communication is most suitable for transforming ideas. Many other studies such as Elliot [2000] and Nelson et al. [2000] suggested that communication within the team members is of utmost importance for effective project management. Moreover, the coordination of projects highly depends on communication [Williams and Cockburn, 2003]. Constant interaction blurs the boundary between the team members and other stakeholders and boosts the feeling of trust. Berki et al. [2007] suggested that trust assumptions are linked to software quality properties and expectations and is built together with the product development through different stages of software development lifecycle. Researchers have also carried out studies on the role of agile teams meetings and how they influence the success of a project. According to Whitworth & Biddle, [2007] meetings are effective to boost motivation among the agile team members. Similarly, Stray et al. [2012] suggested that the team meetings are essential for decision making, knowledge sharing and strengthening of teams. Soloveva's most recent work [2017] also stresses similar issues.

Although there are studies supporting the importance of requirements communication [see e.g. Manninen and Berki, 2003] in the context of stakeholders' involvement, some studies have also presented negative opinions regarding this matter. An extensive literature review conducted by Hummel et al. [2013] concluded that communication is identified as one of the most fundamental aspects of agile software projects by few researchers, only. Other studies reported no or even negative effects of communication on the performance and software process/product success. Communication is neither a 'silver bullet' for successful agile development because; over-communication may become a blocker and hurdles could arise in distributed or extended environments with multiple stakeholders and distributed teams [Hummel et al., 2013]. A negative opinion regarding team meetings was provided by Luong and Rogelberg [2005], who found out that frequent meetings are disruptive and frequently occurring meetings lead to increased fatigue as well as greater subjective workload. Aydinli et al. [2016] have recently investigated these issues in the context of agile/ innovation project management antipatterns.

Similar to communication, another enabler for a software development process is project coordination [Layman et al., 2006]. In a well coordinated environment, members can interact effectively, share knowledge and gain clarity of vision. It is essential for a good methodology to provide an environment where teams can coordinate effectively towards fulfilling the business objectives. Verbal communication is one of the ways for teams to coordinate the project activities. Moreover, project management tools, visual boards and electronic displays, when used effectively, provide better visibility of work and this, in turn, makes project coordination easier. Otherwise, Wears et al. [2007] suggested that handwritten white boards and other visual aids significant role in inter- and intra-group are key enablers for effective project coordination.

2.7. A Scaled Agile Framework®: Agile Meta(modelling); but Scaled Industrial Context!

The Scaled Agile Framework® content and context were first introduced as an approach to the industry and market by Dean Leffingwell and his team in 2011. It is a metamodelling approach framework based on the agile and lean principles which were detailed and explained in previous sections. The emergence of the Scaled Agile Framework® (SAFe®) is, perhaps, best understood and explained by using Jack Welch's quote: *"if the rate of change of the outside exceeds the rate of change on the inside, the end is near"*. (!). Arguably, this has been true in the recent era, where digitalisation is the most powerful key to success for any enterprise. In a sense, the guarantee to the market of a product or service provided by an organisation is how they are tailored to meet the end-users needs. Thus, it also depends on the dexterity of any organisation that develops software and software-based systems. Hence, if the overall process of

software development is leaned and refined, it will directly affect the quality of product and customer satisfaction.

As explained in previous sections, the agile method(ology) paradigm overcomes most of the limitations and drawbacks of traditional software development methods. The agile approach could somehow be considered as a revolutionary change in the software development process but, on the other hand, agile development principles seem to be more suitable for small scale projects. Considering large scale enterprises involving hundreds of practitioners, a more polished and structured system is required. As stated by Leffingwell et al. [2016], the Scaled Agile Framework® is a powerful process metamodel/framework, which provides a systematic roadmap for disciplined software development in large scale enterprises. There exists online available guidance, in which one can evidence that this customised approach combines both agile practices and lean software development techniques that serve as process metamodelling rules in order to guide the large enterprises' software process teams for delivering quality software, regularly and efficiently. This framework not only fits into a large scale enterprise but can also be scaled down to fit the size of under 100 practitioners enterprises. Furthermore, the online material available also lists some case studies of organisations which have adopted SAFe® and have shared their success experiences. These case studies reveal that not only private sectors but even government organisations have adopted SAFe® and gained benefits and stakeholders' satisfaction. Other than software enterprises, sectors such as banking and healthcare have also given positive feedback regarding the use and application of the Scaled Agile Framework®.

2.7.1 An Overview of the Scaled Agile Framework®, SAFe® version 4.0: In Theory and in Practice

The Scaled Agile Framework® can be tailored and scaled based on the type of organisation it is deployed in. Depending on the size of the organisation (application domain) and other factors, the Scaled Agile Framework® can be customised as "3 level SAFe®" or "4 level SAFe®". Figure 2 presents the *3 level* Scaled Agile Framework® including *team, program* and *portfolio* levels. The *"4 level SAFe®"* consists of *one more* level which is "*value stream* level" placed between portfolio and program level. It is optional to include the value stream level; this is based on the needs of the enterprise. The different layers of Scaled Agile Framework® are illustrated and described below:

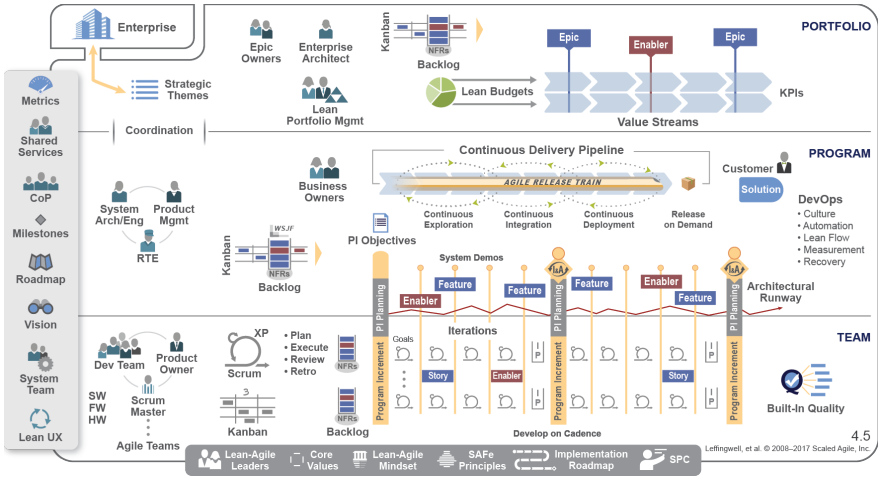


Figure 2: SAFe 4.0 for Lean Software and Systems Engineering(3 level) "© Scaled Agile, Inc." [Knaster and Leffingwell, 2015]

1. *Team level*. The core layer of SAFe® is the Team level and a subpart of the release train. The team level consists of the product owner, Scrum master and development team. The team level members are responsible for defining the user stories, building, testing and delivering values at regular recurring iterations. A release train can have multiple agile teams working towards a common goal. All the agile teams share common iteration cadence to align their deliveries by the end of each PI event. The agile teams participate in the PI event and create a backlog and dependency list among the teams. Later, the teams work independently on their respective backlogs using Scrum or Kanban agile methods. Each team follows a defined roadmap, which starts with planning the sprint, executing, reviewing and finishing with retrospective.
2. *Program Level*. The second layer in figure 2 represents the Program Level. This layer incorporates the agile release trains which are self organising teams facilitating operations of agile teams. Each agile release train can lead 5-12 teams towards achieving a common goal at a shared cadence.
3. *Portfolio Level*. The topmost layer of the Scaled Agile Framework® is the portfolio level. The members belonging to this layer are responsible towards fulfilling the enterprise level missions. The responsibilities include budgeting, funding value streams and coordinating the overall development initiatives of the organisation.

2.8 The SAFe® Values and Principles

In order to succeed in the current highly competitive market, any enterprise must determine certain focus points to ensure they are following the right track towards

achieving their business goals. As stated by Leffingwell et al. [2017], the Scaled Agile Framework® follows a set of values and principles which set the basic foundation for all the actions and activities performed by team members.

2.8.1 The SAFe® Values

1. *Alignment*. If the enterprise has many teams working in parallel towards a common business goal, it is highly essential that the teams are well co-ordinated and aligned. It is important that the teams value overall business goal or program objective higher than their personal responsibilities and outcomes.

How SAFe® fits into core value: The Scaled Agile Framework® has many agile teams working towards a common business goal. All the teams are coordinated by agile release trains. The agile release train ensures coordination among the teams, establishing and fulfilling inter team dependencies and delivering the required values within the same cadence. The agile release train's main objective is to support the value stream's objectives. Furthermore, the value streams further ensure that portfolio level objectives are met. Thus, each layer works towards alignment and coordination of teams and business goals rather than fulfilling personal achievements.

2. *Built in quality*. Quality is the most essential factor for a product or service to succeed. The quality assurance methods are used while development teams determine the end product's overall quality. Quality and customer satisfaction go hand in hand and customers may accept or reject a product based, for instance, on its usability or other quality factors.

How SAFe® fits into core value? The Scaled Agile Framework® has a set of built in quality practices to ensure high quality applications. Each team member is responsible for ensuring quality rather than designating quality assurance checking to other team members. Each phase of development ensures quality by standard quality assurance activities such as unit testing, automation testing, acceptance testing and continuous integration.

3. *Transparency*. In a large scale enterprise, it is difficult to gather a clear picture of the overall progress of all the subsystems. There can be redundancies, such as teams are not co-located, and, further, there night be different business goals to achieve. Hence, it is essential to have a transparent and open development methodology within the organisation to build trust among employees and realise smooth development.

How SAFe® fits into core value? The scaled agile leaders create an environment of openness and trust within the organisation. The agile teams are usually co-located and share the work space. Hence, it ensures a friendly environment and all the team members can interact smoothly. Further, the PI planning, sprint

planning and retros are carried out regularly to gather a clear picture of the overall progress of all the teams.

4. *Program Execution*. Any organisation driven into agile environment must be able to deliver value at regular intervals. The Agile paradigm values the actual working software rather than deliverables such as documentations, which do not add value to customers. Thus, it is essential that an agile methodology follows iterative development to deliver value to customers periodically.

How SAFe® fits into core value? The Scaled Agile Framework® provides a standard set of routines which the agile teams follow to deliver values at regular intervals. The agile release train engineer and other leaders at program level provide extensive guidance and support to ensure smooth program execution.

2.8.2 The SAFe® Principles

Similar to the values, the Scaled Agile Framework® relies on a set of principles ensuring refinement of the software development process and overall software quality. It is evident that enterprise level software development is a complex process involving million lines of code, complex software, multicultural environment and many more challenges. Any specific methodology or knowledge base cannot be a standard solution to all types of enterprises: it rather requires tailoring depending on requirements of each organisation. SAFe® is designed in a way that it can be tailored and customised depending on the enterprise adopting it. The principles, however, are in the roadmap or guidelines which are grounded in all the practices associated with SAFe® and ensure that customising the framework does not affect the credibility of the framework itself. The principles are listed as follows:

1. *Take economic view*. Achieving the business goals by ensuring delivery of values within short time is the essence of enduring in the competitive market. A technically competent product if not delivered to market in time takes too much cost to produce or is very expensive to manufacture is not an ideal fit in the market. The economic impacts of the product within the enterprise and in the market should be carefully studied to understand the overall value it adds to the organisation. The Scaled Agile Framework® values the economics and hence lists it as the first principle.
2. *Apply systems thinking*. It is a fundamental need for practitioner and leaders to understand the system they are building and the system they are working within. The understanding of “what they are doing” and “why they are doing” is essential to deal with any development complexities.
3. *Assume variability*. Preserve options: In contrast to traditional patterns where the requirements and designs are fixed in early stages, the lean agile

development respects variability. This opens the gates for any deign or requirement changes at later stages of development.

4. *Build incrementally with fast, integrated learning cycles*. The lean solutions are built in short incremental interactions. Each iteration is build over the top of previous iteration and keeps on adding value to the solution. Hence, the results are visible from early stages. The solution or the prototype can be subjected for customer validation during the process of development itself. As a result of early customer feedback, risk mitigation could be avoided and defects could be fixed during early phases.
5. *Base milestones on objective evaluation of working systems*. The enterprise level solutions require substantial amount of investment. Hence, it is essential to evaluate the system at regular intervals to assure that the solution will deliver required benefits. The incremental development approach and customer feedback is the key to evaluate the system throughout the development cycle.
6. *Visualise and limit WIP, reduce batch sizes, and manage queue lengths*. To maintain continuous work flow and rapid deliveries, it is important to manage the work in progress smartly. If too much of the work is stuffed within a sprint, it will maximise the chance of not able to finish and deliver any value.
7. *Apply cadence, synchronise and cross domain planning*. The enterprise level solutions inherit complexities because many teams run parallel delivering values independently or with dependencies amongst them. Thus, it is important to have a system which synchronises the development and navigates the cross functional teams to build and deliver at defined intervals. The Scaled Agile Framework® introduces the PI planning sessions, where all the teams gather and plan together for upcoming iterations.
8. *Unlock the intrinsic motivation of knowledge workers*. The work environment and culture directly affects the performance of workers. Thus, it is important to foster openness and trust to motivate the knowledge workers. This involves the leaders to create an environment where individuals can easily reach each other and enter into problem solving and business decisions.
9. *Decentralise decision making*. The rapid incremental development requires that the decisions should not be scaled for longer times. This empowers teams to take decisions when required and proceed with development. Apart from strategic and financial level decisions which are not in the hands of the teams, other development level planning and decision making are carried out in a distributed manner at team level.

2.9 The Team Level of the Scaled Agile Framework®

The Scaled Agile Framework® as introduced in the beginning is a vast framework or process metamodel. This study is limited to the "team level" of Scaled Agile Framework®. This section presents an overview from the existing literature regarding the "Team level layer" of the SAFe® [Leffingwell, 2017]. The team level layer lies at the bottom of the layered structure of Scaled Agile Framework®. The team level comprises of multiple roles, elements, events and artifacts that are depicted in figure 3.

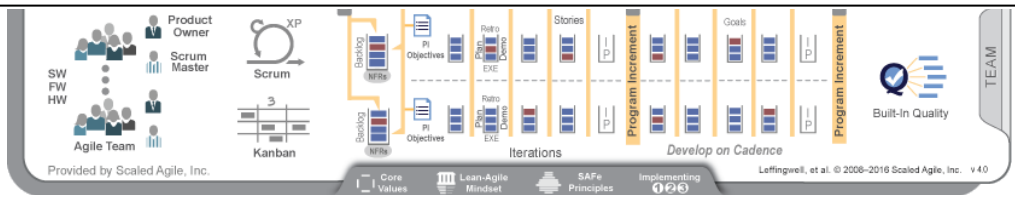


Figure 3. Team level Scaled Agile Framework® 4.0 "© Scaled Agile, Inc." [Knaster and Leffingwell, 2015]

In general, the team level comprises of 5-12 members. Each team reports to the agile release train. The agile teams are independent entities guided by agile release trains to deliver values at regular intervals. The teams are responsible for planning the backlog, development and quality assurance activities. All the teams are synchronised by the agile release train to deliver the values within common cadence.

Figure 3 depicts how the different elements are integrated together to drive the agile teams. A walkthrough from left end to right is explained below:

On the leftmost corner the roles associated with teams are depicted. The roles are assigned in a manner that all the necessary operations for delivering values can be executed by the team members. The description of each role is as below:

1. Product Owner. The product owner serves as the proxy or representative of real customers. This is a role of high significance because the PO serves as a link between business management and agile teams. The responsibilities of the PO includes ownership of team backlog, prioritising the tasks, defining the acceptance criteria for user stories and the only person eligible for accepting them. The PO deals with the product manager and other stakeholders to define the backlog for the agile teams and continues to participate throughout the iterations to successfully deliver the values created by teams.
2. *Scrum Master*. The scrum master is the coordinator and coach to an agile team. They are responsible for planning the iterations, establish tuning within the team members and fostering an environment for teams to work smoothly. The Scrum master maintains that the agile teams follow the track established during the planning, promotes quality practices, facilitates meetings and supports the teams to do continuous delivery depending on business needs. The Scrum master's job can be full time or part time depending on team size and

enterprise scale. It is a usual scenario that scrum master contributes to development or other technical activities together with scrum mastering.

3. *Dev team*. The dev team is a set of professionals who perform the tasks of designing, development and quality assurance activities for the user stories assigned to them. The teams include testers and coders who collaborate in a way that both the testing and coding activities are divided between them. The developers develop and participate in quality assurance parallels by unit testing and code reviews. The testers also participate in writing automation test codes of the features developed by their teams. The teams participate in PI planning and regular sprint planning and retros are guided by the Scrum masters and POs.

In figure 3 the roles are followed by the processes followed by agile teams. A brief overview of these processes is available next:

1. *ScrumXP*. It is a lightweight process followed by self organising agile teams to deliver the values in standard time intervals. The roles described above the essential participants for Scrum Xp process. The sprint is a time boxed event and is generally planned for 2 weeks, as stated by Rasmusson [2010]. The teams review the user stories, the user stories are broken into smaller tasks followed by estimation of story points for all the tasks. The user stories are formulated and entered on visual story boards assigned to each team. The individuals start to develop, perform continuous integration and testing activities for the tasks assigned to them. At the end of each sprint the teams in presence of PO and scrum master review the work performed. If the tasks are done, they are marked completed or they can be moved to next sprint depending on business needs. The teams further participate in sprint retro and analyse the enablers and negatives of the sprint. On a daily basis the teams coordinate in daily stand up meetings, and keep the team posted regarding everyday's progress and concerns.

2. *Kanban*. It is yet another lean method to organise and manage the iterations. The term "kanban" refers to *visual signal* and the idea here is to facilitate the work flow by visualising the progress and establishing the work in progress limits. According to Raut et al. [2015], the main objective of Kanban system is to maximise the productivity of a unit and this is done by reducing the idle time of the process. A Kanban system is a very cost efficient process if applied in proper manner. The teams working using kanban techniques follow the approach of "pulling" the work into the iteration if their team capacity permits, rather than getting pushed by the scope.

Followed by the lean processes in figure 3 (see page 22), the events associated with team level are listed. These series of activities repeating between the Program Increment(PI) planning event are described as follows:

1. *Plan(Iteration planning)*. The first event of every iteration is iteration planning where the agile team discusses the goals of upcoming iteration and how much they can commit from the team's backlog. The meeting is conducted by scrum master and all the team members participate in selecting the user stories and splitting the tasks. The user stories are estimated by the story point system and depending on team's capacity and priority of business needs the user stories are entered in backlog for next iteration.
2. *Execute(Iteration execution)*. Followed by planning, the next step is the execution of plan. The teams start working on the user stories and tasks assigned proceeding with high quality coding and testing activities. The agile team meets up daily for short stand up meetings present the progress of their tasks and identify concerns or impediments.
3. *Review (Iteration review)*. At the end of each iteration the team again meets to track the work done during the last iteration. The team's backlog is analysed and adjusted depending on the amount of tasks done. The work done during the iteration is demonstrated.
4. *Retro(Iteration retrospective)*. Followed by the review, the scrum master conducts iteration retrospective. The idea here is to analyse the positive and negatives of last iteration. This is helpful to avoid repeating the same mistakes and continue the good practices which helped to increase productivity during the iteration. Any other concerns or hurdles associated with the project are also identified and action points for resolution are listed.

Throughout the iteration cycle, the teams refer, maintain and update the backlog. The team's backlog consists of user stories for which individual teams are responsible for. The backlog incorporates all the necessary tasks with detailed descriptions and persons to whom the tasks are assigned. This presents a clear visual picture to set the goals and track the progress of each team (member).

As stated by Leffingwell et al. [2017], all the above events of iteration are, in turn, part of the Program Increment (PI) event. The PI event, in general, is boxed/scheduled within 8 to 12 weeks. All the teams participate, synchronise and demonstrate their plans for the upcoming PI in the Program Increment events. These events help the teams to build a layout of the features and identifying dependencies with other teams. The "Built in Quality" practices are followed at each step of iteration by all the team members to ensure higher product quality. The inspection and quality assurance is not left for a particular phase of development but is rather integrated within all the steps of the agile

development. All the team members are responsible to take quality as a prime factor to accomplish the tasks assigned to them.

3. The Nordea Bank's SAFe® method(ology) – Application and Agility

Banking is a socio-technical business sector which targets a wide range of users and customers. Banking services such as money transactions, loans and internet banking have been integrated into the daily lifestyles of people. In recent years, for providing convenience and better customer experience, the service-oriented banks have made transitions towards digitalisation. In a report in 2014 by McKinsey, [Nordea Case Study, 2017] it is claimed that the European banks pursuing digitalisation will be benefitted by more than 40 percent in their earnings. The banks have been already successful in providing digital services such as online money transactions, loans any many more. Providing web applications together with mobile experience is essential to capture the market and survive in a competitive economy. There are still many operations, which require physical presence or manual handling and present a wide scope to keep on digitalisation and improving customers experiences together with enhanced security and usability [Li et al, 2013].

"Nordea" is a leading financial giant in Europe ranked as the seventeenth largest bank [Nordea Bank website]. There are 1400 Nordea bank branches with the bank's headquarters being in Stockholm. The bank is located in 19 countries and serves around 11 million private customers and 700, 000 corporate customers [Nordea Case study, 2017].

With the aim of enhancing digital experiences for the customers, the Scaled Agile Framework® was adopted in guidance with Ivar Jacobson International in June 2014. The (meta)methodology of working greatly affects the end product and, in turn, affects the customers. Thus, step by step transition towards agile was performed. Introductory workshops were arranged for stakeholders and simulation based techniques on how the Scaled Agile Framework® can be incorporated within the workflow were arranged. There were further follow ups by training sessions to staff members so that they can understand the business goals and prepare their backlogs. The concept of the *'agile release train'* was, in particular, introduced and the teams were divided with each one reporting to their release train engineer [Nordea Case study, 2017].

The development cadence for all the teams was set for 10 weeks starting with the Program Increment (PI) event and followed by iterations for teams to develop the values. The PI event is marked by participation of all the teams belonging to a train and facilitated by their release train engineer and agile coaches for planning the upcoming iterations. The top/executive management of Nordea praised this agile way of working and marked the importance of highlighted issues such as the following: i) transparency in the work environment, ii) better collaboration, iii) self-deciding approach, and most importantly iv) empowerment of the employees.

3.1. Planning the Program Increment (PI) Event: The Necessary and Essential!

The Scaled Agile Framework® was introduced in 2014 in Nordea as a part of digitalisation initiative. Followed by introductory workshop, the first Program Increment (PI) event was held in October 2014 [Nordea Case Study, 2017]. *The PI Planning* is the face to to face event which aligns the teams belonging to agile trains for a common vision and mission. All the team members belonging to agile release train participate and plan together. If there are multiple agile release trains geographically distributed, the event occurs at multiple locations within the same time slot and synchronised by using audio and visual communication between the sites .

In the context of the Scaled Agile Framework®, the PI event is marked as *the most essential step to embark the development process*. The event is 2 days long and has a standard set of agendas for both days. The following depicts the brief overview of the activities involved during the 2 days of this planning event:

3.1.1 PI Planning - Day 1

Day 1 starts with the business leaders' presentations, which include the current state of of business and how the solutions will fit into the business context. The upcoming vision of the organisation based on current market needs are presented. This is followed by presenting the architecture vision, architectural issues, test automation, continuous integration and other development related information. The presentations follow a break and then the teams breakout to begin with the planning in coordination with the agile release train. Finally, the teams present the plans to the management and other business owners. The business owners review and provide the input(s) on the plan. The challenges presented in the draft plan are reviewed and resolved by the management.

Afterwards the teams breakout and gather around in the designated areas and begin planning the upcoming iterations visually, by the means of a board. Each team is provided with a board and a toolkit, which contains all the necessary items for preparing the visual plan, such as markers, sticky notes, threads and a guidelines document to understand the procedure of preparing the visual plan. The team starts with marking the features they are going to accomplish during the upcoming PI. The *feature* is a service that provides value to the stakeholders. Features are then broken down into smaller independent pieces of functionalities termed as user stories. The teams also identify the risks, assumptions and dependencies which can probably occur and mark them on the team board. The user stories are placed in the relevant iteration box based on business priorities and team's capacity. The whole team including the product owner, scrum master and other technical specialists participate and estimate the user stories in the form of story points. Each team repeats the same steps for all the target features and incrementally create the sprint planning board.

3.1.2 PI Planning- Day 2

On day 2, the managers begin the day by presenting the changes in the planning scope and the resources. Next, the teams gather again and adjust the previous plans based on the reviews.

The teams finalise the team board and project the dependencies on the common program board. The risks are discussed and marked as resolved, owned or mitigated. The teams, if needed, adjust the plans and finalise. All the teams objectives for upcoming PI are set by now. This is followed by a short closing dialogue by business leaders. The final program board captures the features and dependencies among the feature teams.

3.2. Iteration Planning and Development – Considerations in the First Agile Steps

Followed by the PI event, each feature team has a clear set of goals for delivering values in each iteration. Based on the planning each one starts executing the iterations. Each iteration is 2 weeks long and includes the following set of activities:

1. Sprint planning. The scrum master invites the team members to plan two weeks of work in sprint planning session. CA Agile Central, a project management tool by CA technologies [CA Agile Central] is used for recording and managing the project development lifecycle. The user stories identified during PI are documented in a project management tool. The backlog in CA Agile Central tool is referred to break the story into smaller tasks and are assigned to specific team members. The stories are then estimated by all the team members based on complexity level story points.
2. *Development and quality assurance*. Followed by planning, the team members start with development and quality assurance activities. The tasks are marked as progress or completed based on the status of each task.
3. *Acceptance testing*. Finally, when the tasks are completed and verified, the product owner of the team performs acceptance testing to accept the tasks as completed.
4. *Review*. At the end of each iteration, the teams gather in sprint review meeting leaded by the Scrum master. The user stories and tasks are again reviewed and marked as accepted or moved to next iteration if some further work is required.
5. *Retrospective*. The iteration is wrapped up by a sprint retrospective meeting. The agenda is to identify the challenges faced in the last sprint, positives of last sprint and any action decided must point to improving future iterations.

A similar cycle of steps is followed for all iterations by all the teams. In this way, any registered stakeholder can track the progress of all the teams.

4. Research Methodology

"Survey" is a popular research methodology utilised in the field of information sciences, usually undertaken by using questionnaires or interviews. Surveys can provide a description of a real world from a wide number of viewpoints [Galliers & Land, 1987]. The research study handled in this MSc thesis was conducted in Nordea Bank, where the agile team members have been using the Scaled Agile Framework® for a few years. For the purposes of this study, a survey method is suitable because the aim has been to capture the opinions and insights of the agile software development team members, who have been using the Scaled Agile Framework®. The participants in this survey were the employees and consultants working for one of the projects in Nordea Bank, aiming at improving digital experience for Nordea customers. The participants belong to the team level layer of the Scaled Agile Framework® working as software developers, Scrum masters, product owners, designers and testers.

4.1. Data collection methodology

A questionnaire was designed and used for the collection of data for this study. The questionnaire method requires well designed and easy to understand questions. The questionnaires can be conducted by meeting the participants or by distributing the questionnaires using email and requesting to reply within a specific deadline. According to Kothari [2008], the questionnaires must be carefully designed and a pilot study should be conducted before distribution, so that the questionnaire may prove to be effective in collecting the relevant information [Kothari, 2008].

For this particular study, a questionnaire was more suitable, comparing to interviews, because the potential participants were not co-located and all had different work schedules. In order to conduct personal interviews with approximately 40 participants would have consumed a lot of time. However, online questionnaires could be answered according to the participants' convenience. In addition, the questionnaires eliminate the possibility of biasing from the interviewers' perspectives.

4.2. Questionnaire Creation and Distribution

The first step was the selection of an online tool which could be used to create and distribute the questionnaires. Out of many online tools, 'surveyplanet' was decided as best to be used for conducting this study. This tool was tested by creating a pilot questionnaire with different styles of questions such as free text, rating based, single choice and multiple choice questions. Furthermore, individualised testing was conducted to ensure that the tool works accurately. This tool provides an easy to understand and user friendly interface.

After the selection of the tool, the process of formulating the questions started. For this study five (5) different sets of questionnaires were designed to serve the roles of the five (5) different categories of potential participants, namely: i) developers, ii) Scrum

masters, iii) product owners, iv) designers and v) testers. Each questionnaire included 25 - 30 questions. A brief introduction was prepared including the general purpose of the questionnaire and the acknowledgements for the respondents. The questionnaire included a few common opening questions, which were then followed by questions based on the respondents' roles and responsibilities. The preparation of the questionnaires took around one month. In the initial phase the questions were primitive and later on the questions were revised and improved a couple of times, based on the suggestions of the research study work supervisors.

After the selection of the tool, the process of formulating the questions started. For this study five (5) different sets of questionnaires were designed to serve the roles of the five (5) different categories of potential participants, namely: i) developers, ii) Scrum masters, iii) product owners, iv) designers and v) testers. Each questionnaire included 25 - 30 questions. A brief introduction was prepared including the general purpose of the questionnaire and the acknowledgements for the respondents. The questionnaire included a few common opening questions, which were then followed by questions based on the respondents' roles and responsibilities. The preparation of the questionnaires took around one month. In the initial phase the questions were primitive and later on the questions were revised and improved a couple of times, based on the suggestions of the research study work supervisors.

After finalising the questionnaires, each set was tested internally by the thesis author and one of the thesis supervisors at the work place. The basic testing was conducted to ensure that the participants could answer using commonly used browsers such as Google Chrome and Firefox. Additionally, it was tested if i) the responses were getting updated instantly after someone answered to the survey and ii) the results could be accessed by the administration.

Finally, a short introduction about the objectives of the survey was prepared and included in the questionnaire, so that the participants could understand the relevance of their participation.

The thesis supervisors at the work place distributed the survey using the Nordea Bank e-mail facility. The number of the respondents chosen were approximately 10-15 in each category. The deadline for answering the survey was kept one month from the date of the distribution. Most of the participants from the developer and tester category responded on time. However, there were less responses from the other categories. Based on the final responses, a reminder email was circulated by the thesis supervisor with an extended deadline of 20 more days. Finally, the answers were obtained and the links to the survey were disabled so that no other participant could respond after the deadline. Table 2 refers to the participants' categories and the final gender-based number of respondents from each category.

Participants	*Total*	*Male*	*Female*
Developer	15	13	2
Scrum master	5	4	1
Product owner	6	4	2
Tester	6	5	1
Designer	5	4	1

Table 2. Participants and male/female respondents of the survey study

4.3. Questionnaire Sections

The questionnaire for each category was focused on specific areas to be explored. The following paragraphs describe the areas which were considered for the preparation of the questionnaires.

The first section was targeted to explore personal data and background details of the respondents. For instance name and gender as well as agile practice experience were considered necessary data to obtain. Further, this section included questions regarding the participants' experience with any scaled agile frameworks and current project details. The participants were also asked about their own knowledge and experience with any other agile/traditional methodology. These questions were useful for establishing an understanding of the respondents' details and experience levels. This section included questions such as "*Years of experience with agile development"* and *"Which other software development methodologies have you worked with?"*

The second section of the questionnaire was focused more on exploring the communication and coordination among the agile teams that have been working in the SAFe® environment. The SAFe® environment introduced a standard approach for development for the agile software process teams. Notwithstanding, the particular practices may have considered to have/bring some disadvantages/pitfalls, so some questions were set to explore these. Hence, the participants were asked to rate their experiences with the SAFe® agile meetings and tools. Furthermore, this section presented free text questions/answers in order to enable the respondents to list the major challenges faced while collaborating with and coordinating other team members. For instance, this section included questions such as, *"Rate the ease of coordination between team members working with SAFe® (Scrum master, designers, developers, other experts)"* .

The third section was focused on exploring the Program Increment (PI) event introduced by the SAFe® methodology. This event is crucial as it includes some major planning activities. The participants were asked to rate their PI experiences and also to list any challenges faced during the event. This section included questions such as, "*How will you rate your overall experience of the PI planning event?"*.

The final section explored the success factors and challenges associated with SAFe®., in general. Moreover, the questions of the final section targeted to find out if the respondents have had any ideas towards the improvement of the agile software development process. In so doing, this section included feedback questions such as, "*What aspects of SAFe® do you like most?" and "What solutions do you suggest to improve the scaled agile framework in order to overcome the challenges?".*

4.3. Data Analysis

The data/answers obtained from the questionnaire were analysed using the *content analysis* [Kothari, 2008] approach. Each section of the questionnaire provided data/ answers in the form of free texts, ratings, single or multiple choice answers. The Survey Planet tool automatically summarises the rating-based results into graphs.

Firstly, each section was summarised in the form of a textual summary, graphical or tabular representation. A similar approach was carried out for all five (5) questionnaires. Subsequently, the results from each participant category were compared and summarised.

5. Results, Analysis and Description Commentary

5.1. Agile and Scaled Agile Framework® experience(s)

The first section of the questionnaire included questions exploring the agile experience of the participants along with the years of experience with the Scaled Agile Framework®. This data provided an overview of the respondents' awareness of the agile methodology and scaling in practice. The summary of the responses posted by the participants were as follows:

1. *Developers*. The level of the experience of developers ranges from 2 to15 years. 9 participants had 2 to 5 years of experience with agile development and the remaining 6 participants have more than 5 years of experience in the same field. The Scaled Agile Framework® had recently been introduced to them and this is the reason that experience with it lies within the range of 1 to 2 years.
2. *Scrum masters*. The level of experience of participants ranges from 1 to 15 years. The Scaled Agile Framework® experience lies within the range of 8 months to 5 years.
3. *Product Owners*. The level of experience of participants ranges from 1 to 5 years. The Scaled Agile Framework® experience lies within the range of 3 months to 3 years.
4. *Testers*. The level of experience of participants ranges from 7 to 10 years. The Scaled Agile Framework® experience lies within the range of 1 to 5 years.
5. *Designers*. The level of experience of participants ranges from 4 to 10 years. The Scaled Agile Framework® experience lies within the range of 1 to 2 years.

In the later part of the first section of the questionnaire, the participants were asked about working with some existing agile methodologies. Table 3 provides a summarised interpretation of responses. It is evident that *Scrum* and *Kanban* were the top choices from most of the participants. A few of the developers, testers and designers had also worked with *lean* development in the past.

	Scrum	*Kanban*	*Extreme Programming*	Crystal	DSDM	Lean development
Developer(15)	14	8	4	0	0	5
Scrum master(5)	4	2	0	0	0	0
Product Owner(6)	6	5	0	0	0	0
Tester(6)	6	5	1	0	0	5
Designer(5)	4	4	0	0	0	2

Table 3. Experience of respondents with other agile methods

5.2. Communication and Coordination in the Agile Process

In the agile software development process, constant communication occurs within team members as a part of the development process. Many studies conducted in the past highlighting the importance of communication and effective coordination towards the success of a project. According to Williams and Cockburn [2003] the most effective way of transforming ideas is face to face communication. Moreover, the coordination of projects depends on communication and this is central to software development [Williams and Cockburn,2003]. Effective communication essentially builds an environment of trust and confidence among the team members. According to Berki et al. [2007], trust is linked to software quality properties; it is built together with the product development through different stages of the software development lifecycle [Berki et al., 2007]. Elliot [2000] suggested that in complex situations communication effectiveness is particularly critical to project success; therein multiple and (dis)integrated stakeholder teams are involved and where 'time to market' and project efficiency are key drivers. The scaled agile process falls into a similar category where multiple stakeholder teams collaborate together and aim at delivering values to customers within regular intervals.

Yet another aspect is the coordination of a project, which involves effective management of expertise and knowledge. According to Nelson et al. [2000], effective coordination is dependent on communication, ability to share knowledge at mutual influence. Apart from verbal communication, the tools and techniques used to gather and share knowledge equally impact project coordination. According to Wears et al. [2007], traditionally, handwritten whiteboards, electronic displays and other such tools play a significant role in inter and intra group communication and coordination. In so acting, important information can become visible or available to everyone with the help of these traditional tools.

5.2.1 Communication and Coordination in SAFe® Environment

In the second section of the questionnaire, the participants were asked about communication and project coordination in SAFe® agile environment. Apart from verbal communication, the commonly used tools for daily activities involves CA agile central for project management, flowdock, emails and Skype for synchronising the team activities. The participants were asked to comment on the ease and convenience of communication within the team and with members outside the team. The following table illustrates the viewpoints from the perspectives of each role.

	Poor	*Below Average*	*Average*	Above average	Excellent
Developer(15)	1	0	8	4	2
Scrum master(5)	0	0	3	1	1
Product Owner(6)	0	0	3	1	1
Tester(6)	0	0	1	2	3
Designer(5)	0	1	3	1	0

Table 4. Ease of communication using SAFe® tools

Table 4 presents an interesting summary of the ease of communication in the SAFe® environment as experienced by participants. Eight (8) out of fifteen (15) developers found it average. However, some of them also rated it as 'above average' and 'excellent' experience. Most of the Scrum masters, product owners and designers rated their experiences as average. The majority of the testers were satisfied with the experience and rated it as 'excellent' and 'above average', which are the highest ratings in the given scale.

5.2.2 Challenges in Team Communication and Coordination

The participants were further asked to list the challenges faced regarding this matter. Developer number 8 (Dev8) mentioned: *"The design teams for some project teams are not co-located. In this scenario it becomes difficult to coordinate on daily basis. The Flowdock or Skype sometimes lacks the benefits of the face to face communication*". Dev15, also gave a similar response regarding the difficulty to communicate with the designers. Des5 had similar opinion regarding the difficulties in team coordination because teams had not been collocated in the same location. One of the testers, QA6, commented about the distributed teams and revealed that sometimes the personality and attitudes of team members can prove to be hindrances to collaboration. This is an interesting observation, as the skills of effective communication vary from person to person. QA5 pointed that there are 'lots of teams' and usually 'multiple dependencies with various teams'. Information gaps and delays are common occurrences and affect the project plans. Similar issues were also pointed by SM2, that are many inter-dependencies that can hinder end to end ownership of certain features.

5.2.3 Team Meetings

Other situations where the teams meet in order to communicate, collaborate and share knowledge are the *team meetings*. A software expert usually spends several hours of an iteration in meetings. The typical sets of meetings include sprint planning, reviews, retrospectives, backlog refinements and daily stand ups. These meetings are essential in order to plan the iterations, track the work progress and perform business decisions. Several studies had been conducted in the past to analyse the effects of the team

meetings in the agile environments of the team members. One of the studies in the past concluded that daily or weekly team meetings are an important motivator for both individual and group actions [Whitworth & Biddle, 2007]. Stray et al. [2012] suggested that meetings are the key towards shaping both team and organisational outcomes. The teams' meetings provide help for the teams to coordinate, make decisions, share knowledge and strengthen the team orientation. Nevertheless, in the study conducted by Luong & Rogelberg [2005], it was found that frequent meetings are disruptive and frequently occurring meetings lead to increased fatigue as well as greater subjective workload.

The Scaled Agile Framework® provides a standard set of meetings to be conducted. These meetings are an integral part of SAF®e environment to establish coordination among team members and to achieve a common set of goals. The given questionnaire also explored the different types of meetings attended by respondents and how useful these meetings had been from their point of view.

The SAFe® related meetings that were usually attended by the participants were sprint planning, mid-sprint review, sprint review, sprint retrospective, backlog refinement, PI planning, PI demo, daily standup and Scrum of Scrums. Table 5 illustrates the overall satisfaction level experienced by attendees of these meetings.

	Not useful	*Somewhat useful*	*Useful*	Very useful
Developer(15)	3	4	6	2
Scrum master(5)	0	1	3	1
Product Owner(6)	0	1	5	0
Tester(6)	0	1	3	2
Designer(5)	0	3	2	0

Table 5. Usefulness of SAFe® meetings

Most of the participants rated the meetings as useful. Some of the developers, Scrum masters and testers rated the meetings as very useful. A few developers marked the meetings as "not useful". However, the big picture signifies that the participants recognised that the meetings were a useful feature while working in the SAFe® environment.

5.3. Program Increment (PI) Planning Event Experience

The third section of the questionnaire covers the PI event experiences from the participants' points of view. The PI Planning is the face to to face event which aligns the teams belonging to agile trains for a common vision and mission [Leffingwell et.al, 2017]. According to Leffingwell et al. [2017], the PI event serves as the heartbeat of the agile release train which aligns all the teams for a shared mission and vision. All the team members belonging to agile release train participate and plan for upcoming

iterations together. It is a very crucial activity from the perspective of the SAFe® framework. The questionnaire included questions to find out about the responsibilities of team members during PI event and overall experience associated with this event.

5.3.1 PI Event

The team members gather together and perform the activities related to planning the upcoming iterations. These activities include formulating and marking user stories on team boards, estimating stories, identifying risks, issues and assumptions and finally resolving the dependencies.

	Poor	*Below average*	*Average*	Above average	Excellent
Developer(15)	2	4	7	2	0
Scrum master(5)	0	1	3	1	0
Product Owner(6)	0	1	1	4	0
Tester(6)	0	0	2	3	1
Designer(5)	0	0	4	1	0

Table 6. PI experience(s)

The participants were asked to rate their experience of PI event from poor to excellent. Table 6 depicts that most of the developers, Scrum master and designers rate the experience as average. Four 4 out of 15 developers rated this as a below average experience. However, the majority of the product owners and testers rated it as an above average experience. One out of 6 testers even rated it as excellent. The overall picture indicated how mixed were the opinions about the PI event from the different views of each specific agile team member.

5.3.2 PI Event improvement ideas – What were the agile developers' thoughts?

In addition to the rating of the PI experience, the participants were asked to suggest some ideas to improve the PI event. Dev15 suggested that it is not essential for all the team members to attend the 2 day long session, especially if there are members with same platform expertise. SM2 also shared similar opinion with the statement "*PI event has too many people and too many teams*"

The majority of the participants drew their attention towards the importance of pre-planning before the actual event. Quoting Dev10 "*Proper planning in teams beforehand. Only PO and SM to participate into PI*. According to Dev4, "*Teams should have user stories ready and estimated before PI planning. Teams could try to assign user stories to sprints. Teams should know user stories with dependencies before PI planning and just discuss with the other teams regarding schedule.*" The issue of the

clarity of the agenda before the planning was also stressed by some of the product owners and Scrum masters.

A recurring suggestion by most of the respondents was to shorten the length of PI event from 2 days to 1 day. QA6 suggested: "*Pr- plan most of the items before the event. During the event, just spend time to sort out dependencies. Cut shorten the event to 1 day*." Also SM4 stressed that *"only one day event is enough if prior work is done. we just need to align with other team during PI event."* Another interesting viewpoint in similar direction from Dev 8 was: "*One day instead of two would be plenty for most teams. Last PI planning was 60% just waiting for it to end. With a bit of pre-work (e.g. writing down the proposed user stories on post its), one day would be more than enough.*".

A few other respondents also mentioned about shortening the PI ceremonies and presentations and focusing more on resolving dependencies and planning agenda. One of the product owners PO5 suggested: *"More time for team work instead of spending time for presentation of the scope ambition."* Similar opinion was shared by QA3 stating: *"Minimise the common general presentations during PI event planning. Dependencies should be discovered early on first day"*

5.4. Success Factors and Challenges of the Scaled Agile Framework®

The final section of the questionnaire explored the success factors and challenges of the SAFe® agile development. The participants were asked to list the features of SAFe® that they liked most. Furthermore, the participants were asked to list the challenges they faced while working in the SAFe® agile environment. In addition, some enlightening ideas to overcome the challenges they met throughout the development. These were intentionally kept free text questions so that participants should not feel bound to specific areas but rather be able to answer from a broader perspective. Finally, the questionnaires were closed by the section where they were asked to rate the PI experience and to make any additional comments.

5.4.1 Success Factors of the Scaled Agile Framework®

A number of studies has been conducted in the past to identify the factors which are present in the development methodology and ensure the project success. For example studies exist in the form of literature reviews such as by Dikert et al. [2006] and also in the form of case studies conducted on the teams working using a certain methodology or development process (meta)model. One of the interesting studies conducted by Schatz & Abdelshafi [2005] was based on Primavera's development team's transition from traditional to agile method(ology). The study revealed that adopting Scrum significantly improved the work environment, enhanced trust and improved the relationships between team members. Additionally, working closely with product owners and stakeholders one could see how information sharing led to better business understanding. Another survey study conducted by Chow & Cao [2008] revealed that

factors such as an agile friendly environment, a high caliber team, strong customer involvement, correct delivery and proper practicing of agile software engineering techniques are critical success factors for a project. Dikert et al. [2016] conducted an extensive literature review on large scale agile projects and concluded that the choice and customisation of an agile approach, agile coaching and training, support from the management and embarking with pilot projects are also the key factors which affect the success of a large scale project.

Research question 1 aimed at finding out success factors of the Scaled Agile Framework®. In this study the Nordea Bank employees/participants came up with interesting reflective and critical comments for the question about the features of the Scaled Agile Framework® that they mostly appreciate. Many respondents valued the *"structured planning based on clear vision"*. According to PO4, the SAFe® approach ensures clear value stream identification, structured and delivery oriented planning. Also, it was somehow acknowledged that large scale organisations have special needs when it comes to agile delivery. Des4 and Des3 also shared similar feelings about structured timing and planning, incremental builds, better planning and prioritised work. One of the developers, Dev8 stated: *"SAFe® ensures reasonable stable environment and it incorporates Scrum at its core which I like a lot.*".

Further, the structured planning process, understandably, creates an atmosphere of trust and confidence within the team members and is clearly valued by most of them.

Hence, another interesting factor identified from the survey was the *"transparency and better visibility of work"* throughout the organisation. The project management tools such as CA Agile Central were utilised so that an authorised member could view the work flow of any team throughout the organisation. The visual boards, team boards and dependency boards, and shared demos at regular intervals provide transparency and better visibility of work throughout the organisation. Des1 commented that *"I think, SAFe has made the progress quite visible. Organisation runs on same cadence and we have shared demos"*. Des 3 also stated that "*SAFe® makes work more visible"* as a most liked feature. One of the Scrum masters SM3 also acknowledged the same by stressing *"transparency across the organisation"* as a key success factor. PO5 agreed with similar comments and added that *"SAFe® makes big picture more visible for the team members"*. Overall, many respondents value the idea of how SAFe® ensures transparent atmosphere and better visibility of work.

Notably, the respondents appreciated the "*involvement of top management and stakeholders"*. One of the testers, QA1 noted that: *"Higher management having a close look on dependencies and risk"*. SAFe® ensures the involvement of top management with the core team during the PI event and throughout the development cycle. Another remarkable comment was made by QA4, stating the following: *"Possibility to meet all the stakeholders and have fruitful discussion about future implementation and getting new improvement ideas from outside of team"*. Apparently, the atmosphere and idea of

working close to business people makes the team members enhance their shared understanding and awareness of the higher business goals. In addition, this minimises information gaps and helps in building trust between the software development team members and other stakeholders.

Other popular features were: better communication, tools and flexible working atmosphere. However, there were some negative responses such as those by Des5 saying that *"Oh, I have not experiences in any factor"*.

The following Table 7 lists the success factors that were identified through the questionnaire.

Success factors	
Planning	Stable planning, planning together with the team, planning risks and dependencies in advance, clarity of big picture.
Transparency and visibility	Transparent atmosphere, access to any team's workflow, team boards, dependency boards and CA agile central.
Involvement of top management and stakeholders	Possibility to interact and information sharing with all stakeholders and business people, shared demos at regular intervals, involvement of stakeholders during PI event.
Communication and coordination	constant information sharing, daily meetings, sprint activities, visual boards.

Table 7. Success Factors of the Scaled Agile Framework®

5.4.2 Challenges of the Scaled Agile Framework®: How agile is SAFe®?

Similar to the success factor studies, many studies have been conducted in the past to identify the challenges faced while adopting agile methodologies. According to Nerur et al. [2005] "*Most organisations cannot ignore the agile wave, but for those steeped in traditional systems development, adoption of agile methodologies will likely pose several challenges*". Cristal et.al [2008]. conducted a study to identify challenging implications of the Scrum agile method in a globally distributed environment. The major challenges identified and reported were: lack of formal documents, inadequate team structure, communication issues and management not used to agile practices.

Research question 2 aimed at finding out challenging for the development process factors of the Scaled Agile Framework®. In this study, the participants were asked to state the challenges faced by them while using the Scaled Agile Framework® methods, techniques and tools as a foundation for software project development activities. One of the major challenges highlighted by many respondents is the nature of SAFe® which is *"not truly agile"*. The Scaled Agile Framework® has a planning process, where approximately three months of planning is conducted in advance during the PI. While many members identified the stable planning as a success factor, some of the respondents identified it as a challenge to face with the use of the Scaled Agile Framework®. According to Dev11, "*Planning 3 months ahead is not very agile at all*". Another developer P8 observed and stated that SAFe® is: "*Not very adaptable to*

changes, many teams spend lot of time in planning and there is hardly any room for changes".

Furthermore, respondents concluded that *"rigidity and lack of innovation"* are another two challenging factors of the Scaled Agile Framework®. Following the book version of the framework poses a standard set of meetings and activities to be performed. Some respondents felt that this created a closed 'cage' environment with less room for innovation. Although there is a sprint associated with innovation, most of the times it is used for wrapping up remaining last PI and planning next PI. QA6 also emphasised that the framework is *"Too rigid and there is no room for improvements and innovations"*. In the same direction, an interesting remark was made by Des5 saying that suddenly *"knowledge does not scale and a horrible cage is born"*.

The next challenging factor identified was *"dependency between the teams"*. In a large scale organization, where the teams are distributed, the framework ensures that every team works on same cadence and dependencies can be planned during the PI event. However, in practice, there can be reasons for the teams' deviation from the plans and, hence, the dependent teams inevitably suffer. QA5 was outspoken concerning the *"Huge numbers of teams, how to get all in sync. Teams often miss the schedules because of dependencies between teams"*. QA3 also expressed similar opinion regarding dependency management and synchronising teams.

Finally, most of the designers mentioned that there is no structured requirement gathering process documented in SAFe®. One of the designers Des1 commented that *"SAFe® talks about how things are done after requirements have been specified, but it doesn't talk about the process of how requirements are specified. Since UX design is mostly about specifying the requirements, large part of SAFe® ends up missing UX design. I think that's a challenge."*.

Some other challenges mentioned by the respondents were: legacy systems, lack of involvement in major business decision which are done at program and portfolio levels, and complicated architecture.

Table 8 lists the summary of the major challenges of the Scaled Agile Framework®, as identified through the survey.

Challnging factors	
Not truly agile	advance 3 month planning, less scope of changing requirements.
Rigidity and lack of innovation	Rigid practices, less room for innovation and improvements
Dependency between teams	Teams may sometimes deviate from plan affecting the depending teams, synchronising whole organisation is challenging.
Design process	Missing standard design and requirement gathering process.

Table 8. Challenging factors of the Scaled Agile Framework®

5.5. Improvement Ideas: How SAFe® could be Safer in Use and Application!?

Research question 3 aimed at collecting and presenting improvement ideas in order to make a safer use for SAFe® (!) and for all the Scaled Agile Framework® (sub)processes. The participants were asked to answer the free text question and provide suggestions in order to improve the SAFe® process. The following points are the summary of the responses provided by the agile development teams' members.

1. *More innovation*. Many members had paid attention to the fact that more room is needed for innovations. One of the developers Dev12 mentioned that there should be focus on achievable deadlines and enough opportunity to innovate. Admittedly, the innovation sprint exists but it essentially gets consumed in quality assurance and upcoming PI preparation. PO3 shared similar opinion and also appealed to more room for continuous improvements but also trying out new things, which can make the scaled agile processes better.
2. *Better learning*. Another area to improve, suggested by many respondents, was the training function. Although SAFe® has training sessions it is essential that everyone is exposed to training before entering the team level layer in the organisation. Moreover, the training sessions could be focused on on the roles of the participants. In order to effectively work with this framework and its methods, it should be ensured that everyone gets trained and training sessions could be improvised. Dev10 stated that there should be heavy training for top management to create and prioritise on the features. PO1 shared similar opinion and said that there should be clarity on what a PO is supposed to deliver and what are the defined tasks that they should take under their control.
3. *Dependency management*. SAFe® assumes that there will be dependencies. Identification, drafting and connecting them with red strings on the dependency boards during PI is the solution for dependency management. However, many respondents confessed that this is a major challenge and admitted that the planning usually fails because of too many dependencies between the teams. SM3 considered that dependency handling is a real challenge and that the framework should include a process on cross collaboration between agile trains to resolve dependencies. QA3 insisted that during the PI event, the dependencies should be treated as priority and should be discovered earlier on the first day so that teams get time to discuss and plan them properly.
4. *Design process and customer participation*. Designers came up with interesting ideas to improve the design process. Des4 stated that there should be measurable objectives for the design process and the design process should be a part of incremental planning. Des3 commented that strategic design work

should be embedded within SAFe® iterations rather than planned outside. Des4 also put emphasis on bringing customers to the centre of the design process.

6. *PI process*. As an improvement suggestion for the PI process many participants suggested on shortening the length of the planning event and prior preparation before the planning event. Dev2, Dev3 and Dev 4 proposed that with better planning beforehand, the PI event could be completed within 1 day. Similar opinions were shared by some of the Scrum masters, testers and product owners.

6. Discussion and Conclusions

In the context of this thesis an overall rating of the agile software framework SAFe® was achieved by the survey respondents at Nordea Bank. The respondents provided their reflections as co-employees, team members and agile software developers, and expressed their opinions regarding their ways-of-working with the SAFe® methodological principles. Figure 4 illustrates the overall experience of each role using the Scaled Agile Framework®.

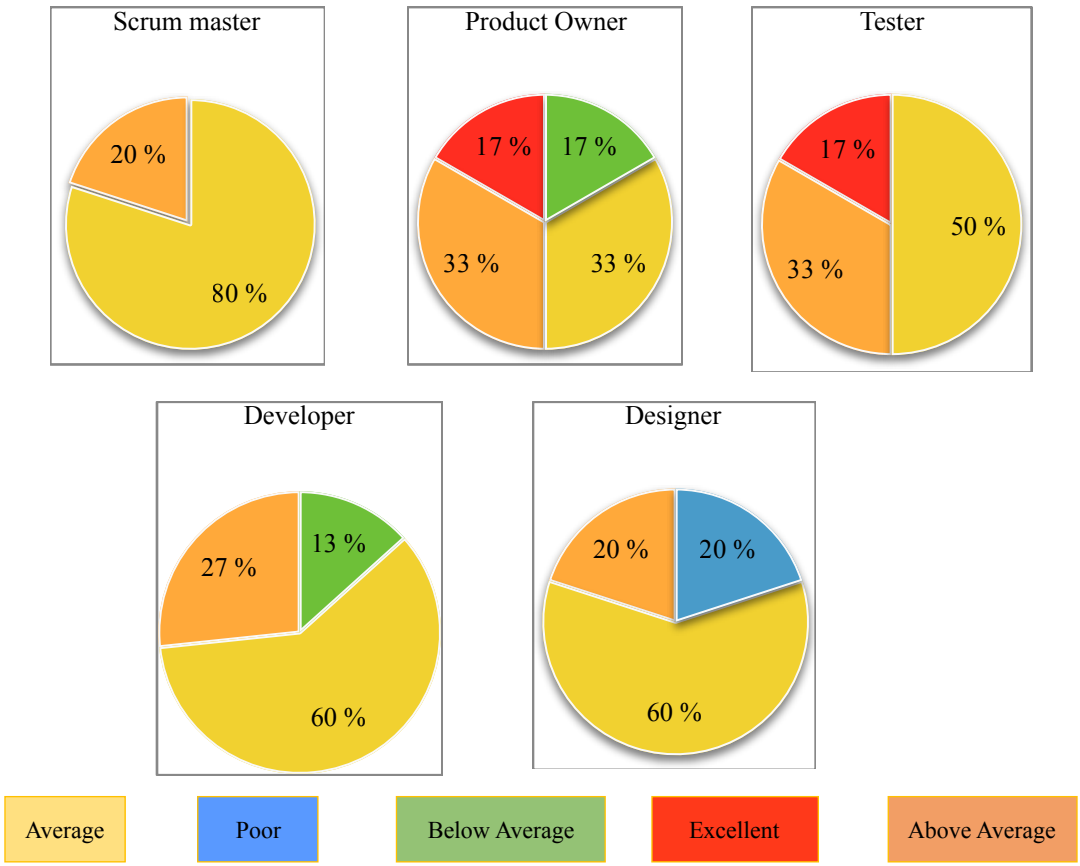


Figure 4. Overall Experience rating

6.1. A Big Picture

The creators of SAFe® strongly believe that SAFe® significantly increases the employment satisfaction. One of the business benefits of the methodology is that it ensures motivation and gives more power to the people. “Respect for people and culture” is also a pillar in the SAFe® house of Lean [Knaster and Leffingwell, 2015].

In this study at the banking sector and at Nordea branch at Helsinki, Finland some developers marked SAFe® as an above average experience while more than half of the developers found the SAFe® experience as average.Remarkably, a few developers rated it as a below average experience. It could be concluded that SAFe® does offer average experience for software developers in the banking sector. The majority of the Scrum masters rated it as an average experience and a few of them as an above average. In short, SAFe® seems to provide a good work environment for Scrum masters. The

ratings by product owners provided a mixed opinion. A few of them found the SAFe® experience as excellent while a few rated it below average. Nevertheless, the majority of them rated it as an average and above average experience. Thus, most product owners were satisfied while a few other encountered some other issues. Half of the testers rated the experience as average and some of them rated it as excellent, too. None of the testers rated it as a poor or below average experience. Hence, it can be derived that testers were mostly satisfied with the SAFe® method(ology). In addition, most of the designers rated it as an average experience. Some of them also rated it as poor. These are the only group of participants who have marked the experience as poor. Thus, it can be derived that SAFe® is rather failing to impress the designers' community.

In general the teams were satisfied with the structured planning process and the essence of Scrum embedded within the process. In Nordea bank Helsinki Finland, since the majority of the participants/software developers were experienced with Scrum and a few other agile methods, the adoption of SAFe® did not appear to be a difficult transition. Admittedly, none of the survey participants mentioned that it was difficult for them to understand the framework. However, some of them emphasised that the training sessions could be more focused and improved. According to the respondents, it must be ensured that everyone gets trained before starting to use the SAFe® methodology so that they could understand the concepts and use them effectively rather than treating the agility as a burden over regular work.

In general everyone appreciated the better visibility of the work after integrating SAFe® tools within the system. The meetings consumed a significant amount of time of an iteration. It is essential that the time is utilised efficiently and helps the teams to perform better to reach the goals. The participants seemed moderately happy towards the meetings and did not mark it as a blocker or hindrance in their regular work. The meetings are a good place to track the work progress and retrospectives specially provide opportunities for discussing continuous improvement ideas for teams to work efficiently. The PI meeting received mixed reviews ranging from few poor ratings to excellent experience rating. The meeting is indeed an essential element of the process; however, some suggestions such as better pre-planning and the more manageable dependencies would make the PI event more effective for everyone.

Some other major issues and improvement ideas were related to integrating a structured design process as an element of SAFe® and better way of managing dependencies.

The following sections compare some of the benefits of SAFe® as claimed by experts and creators and contrast them to the results obtained from this study.

6.2. Alignment and Transparency

The SAFe® approach is based on optimising the system as a whole. According to Knaster and Leffingwell [2015] one of the key business benefits of this framework is

that it aligns the organisation towards a common agenda rather than optimising individual teams. SAFe® makes the information flow and tracking of the work much easier. When both management and teams are aligned and developments done using common cadence, everyone will be focused on one purpose, that of making the system better.

This research study supports this commitment and the research outcomes indicate that team members appreciate the clarity of vision and synchronous operations of business and team level. The team members were happy that they could collaborate with business stakeholders and continuously get improvement ideas from them. The constant knowledge sharing among the business people and teams helps them to align their tasks towards achieving the business goals. The involvement of middle and executive/top management in PI events and during shared demos makes the progress of the teams crystal clear for the management and removes the possibility of negative surprises towards the delivery.

6.3. Communication and Coordination

The SAFe® approach is based on agile values and being such stresses the concept of interactions during documentation. One of the questions that SAFe® answers is: “How do we create an environment which fosters collaboration and relentless improvement and unblocks intrinsic motivation of people who do this work?” [Knaster and Leffingwell, 2015].

The Scaled Agile Framework® promotes constant communication during meetings such as daily stand ups, sprint planning and retrospectives. Apart from verbal communication participants are engaged in preparing boards and using tools such as CA agile central to visualise and track the work progress. The research study results indicate that the majority of the participants were satisfied with collaboration practices and listed it as one of the success factors. Notwithstanding, some of the participants also seemed to be dissatisfied in this area. The major challenge faced by the participants had to do with the fact that the design team is integrated with the Scrum team; hence, daily communication with them and getting design suggestions does not happen effectively. Moreover, according to the respondents’ words, the existence of too many teams and too many people also seems to affect (the performance of) some team members. In particular, the participants had expressed their concern that when there are teams with multiple dependencies, especially outside the agile train, then the communication process becomes demanding and perhaps obsolete. Thus, it can be concluded that although following the principles of SAFe® fosters an environment where collaboration should be easy, there could also be problems in agile practice due to some factors such as the distributed teams, too many dependencies and the attitudes of individual team members.

6.4. Dependency Management

The SAFe® framework, as a disciplined process metamodelling approach [Berki, 2016] defines strategies to manage and minimise the dependencies among the agile software process teams, programs and value streams. However, the results of this study indicate that the dependency management should be optimised more. The banking sector participants expressed that need and mentioned that in most of the times their teams have had many dependencies so any minimisation strategy does not seem to work very effectively. If one team deviated from the plan, then the dependent teams have also had to suffer. Conclusively, the SAFe® process metamodelling approach should treat the dependency as an impediment and should include some framework instructions for cross collaboration of the different agile trains. This could be helpful for managing dependencies effectively. Although the dependencies are planned in advance, if they are not resolved during development, there should be possibilities for the teams to incorporate new requirements as a backup option.

6.5. Conclusive Remarks

The main objectives of this study had been to identify success factors and challenges associated with the use and deployment of the Scaled Agile Framework® in the banking sector and Nordea Bank in particular. Moreover, the aim was to identify and present some improvement suggestions by the stakeholders towards the improvisation of the agile software process management. The study utilised the survey research methodology along with observation to answer the research questions. The survey was finally conducted by using online questionnaires for participants working as team level members at Nordea bank.

Research question 1 was: *"What are the success factors of the Scaled Agile Framework® at team level?"*. The research found out that structured planning, transparency, visibility of work, communication and coordination between teams and involvement of management and stakeholders are the key success factors of the Scaled Agile Framework®.

Research question 2 was: *"What are the challenging factors of the Scaled Agile Framework® at team level?"*. The research study revealed that rigidity, lack of innovation, dependency between teams and lack of design process are the major challenges faced by team level members.

Research question 3 was: "*What could be possible improvement suggestions for betterment of the SAFe® process at team level?"*. Based on the responses of the questionnaire and personal observations, future focus on innovation issues, effective training sessions and dependency management improvement are some of the ideas which could improvise the gaps identified in the process. Moreover, there appear to be some lacking (sub)processes such as end to end delivery, design process embedded

within a PI and more customer involvement. Some ideas to incorporate these missing sections were also suggested by the study participants [See section 5.5].

The big picture in figure 4 (see figure 4 on page 44) signifies the overall experience level of the participants from each category. The developers and Scrum masters seem to have average satisfaction and some of the testers, in particular, rate their experience as excellent. However, the product owners exhibit mixed feelings and designers in particular are not very impressed with the framework use.

Analysing the communication and coordination aspect, the team members were, in general, satisfied with the meetings and communication practices offered in this framework. However, it is evident from the results that if the teams are large and distributed, it sometimes becomes difficult for team members to share knowledge as compared to face to face communications in smaller teams. The SAFe® offers multi-train architecture with multiple teams working under release trains. In these situations of cross communication between the trains, difficulties may arise. Although SAFe® presents the PI event where the teams could meet up and handshake on dependencies, it could be difficult to establish coordination if the depending team belongs to different train or if the teams are not co-located. The team members appreciated the effectiveness of tools used for management and communication purposes; however, there could be hindrances due to personal communication skills or attitudes of team members. Another case where teams gather up are meetings, and participants in general are happy towards the meetings' function and functionality. Additionally, to increase the effectiveness of the meetings, pre-planning of meeting agenda should be done beforehand by the coordinators. It would make the meetings more focused and the meetings' participants would derive maximum benefits from such stakeholders' gatherings.

The PI event rating activity and subsequent results exhibit mixed feelings and rankings (see Table 4 on page 35)by all the participants. It is evident from the results that there were/are some pitfalls of the events instances that should be improved. The participants mainly demanded i) the shortening of the length of the event and better pre-planning. It was indicated that it is better to have short sessions in general; however the PI Planning event is a time and place where all the teams belonging to an agile train meet up and plan together. Even with a pre-plan, the actual team planning and dependencies could take more time than expected and it is safe to keep the PI event length as 2 days. However, in such long events, it is common for participants to loose focus and decrease in overall productivity. Hence, to refresh the participants and to increase their productivity, there should be room for energisers or fun activities within the event. This would lighten the mood and the general cognitive overload and foster better interaction and information/knowledge grasp among different teams. As a result, the participants would be comfortable to interact and handshake on resolving specific dependencies.

Factors such as stable planning, transparency and involvement of management stakeholders were/are especially appreciated by agile team members in the Nordea case study. SAFe® seemed/seems to excel in the area of increasing development project visibility throughout the organisation. From the beginning of the PI Planning event, SAFe® ensured/ensures maximum usage of visual boards and over communication of vision throughout the development lifecycle. A number of shared demos, daily meetings and constant backlog refinements were/are conducted in the presence of management and, notably, this prevents any negative surprises for some management layers. Evidently, all authorised stakeholders can track the progress of the/any project and are well aware of any risks, defects and other issues that may come up. These are excellent factors because the management does not get any surprises after development and, hence, they can plan their sales and other business decisions timely, based on the project's development and progress.

There were/are some areas in SAFe® deployment where the Nordea Bank case study participants faced challenges and they suggested some improvements. Accordingly, there should be some proper training for new members joining the organisation so that they can understand the importance of the framework use and implement it effectively. Additionally, in a large scale organisation, like the Nordea Bank, it is common to have dependencies between the teams. The evidence from the case study results has been strong enough to establish that the dependency management process needs improvement. Establishing dependencies in the PI Planning Event and marking them on boards using red strings should not be the end of the process. There could and should be mechanisms to show how to track dependencies throughout the iterations and, if they exist, subsequent software quality management strategies should be applied to minimise the number of the dependencies. Further, if one team deviates from the plan, the dependent teams should have some backup option to prevent their plans from failing. Thus, preventive (and not simply reactive) maintenance should take place.

Another major focus area revealed to be the effort to improve the agile design process. The design team does not follow the common development rhythm as other teams and hence, the design tasks are usually not included in the plan. This process should be optimised so that the designers could collaborate more and work together within the teams. It would be helpful to have designers as Scrum team members and constantly reviewing the deliveries from the design perspective.

The above stated case study research results were useful for the Nordea Bank's software development project team, which has adopted the Scaled Agile Framework® SAFe® in the recent years. This study gathered and analysed responses of the participants of the development project team. The study and analysis collectively provide a clear snapshot of the benefits and challenges faced by the currently involved participants following the SAFe® principles in application.

Moreover, and without loss of generality, any organisation moving towards scaling agile or adopting agile can refer to the results of this study to consider in advance the benefits and challenges that agile scaling can pose on the team members and on the software process management. This study could also be used by practitioners and researchers studying the scaling agile frameworks and methodologies comparing the results in terms of software quality properties and software quality features management.

6.3. Limitations of this Research Study

This study was carried out in order to analyse the success factors and challenges of SAFe® adoption at teamwork level in the banking domain, and being such it has certain limitations, by its scope and nature. The results derived are based on a limited number of participants, who had a specific role in the development process. The study does not focus on program level, value stream or portfolio level members of SAFe®. The results would have been different if the business layer people had also participated.

The study was conducted by using an online questionnaire. The questionnaire was only used once and the results were based on the circumstantial situation of the participants at that particular time period and stage in the software development project. If personal interviews or group interviews were used as a research method, the results could have been different. Moreover, if the survey or interview had taken place for a coupe of times within a year, the responses from the same dataset of the participants might have been changed.

The number of resulting responses do not constitute a sufficient sample to draw conclusions regarding the efficiency of the Scaled Agile Framework® application and applicability in industrial settings. Nevertheless, the survey questions were set in order to form a base to provide indicative answers to the research questions of this MSc thesis. In so doing, one ca say that there is a wide scope to further study the SAFe® deployment as an adaptable and customised framework in the banking sector and similar sectors. Understandably, from various viewpoints the results would have been different depending on the research questions; and would have been analysed and interpreted differently! Moreover, the participants belong to the banking domain, while other domains such as healthcare or gaming could possibly appear to have different success factors and consider different challenges.

The researcher’s objectivity/subjectivity is a critical issue for social researchers, especially those using qualitative and interpretivist approaches and/or are insiders or outsiders to the social reality they study. One must accordingly recognise that the socio-technical work environments of the banking systems, as whole, are highly interactive systems within the society. Though the thesis author utilized both a questionnaire and observation for this research study, in the context of this thesis only the questionnaire responses by the team members mostly count for investigating the attitudes,

experiences, success factors, challenges and overall satisfaction while working with the SAFe® approach. The personal observations of the thesis author are not taken into consideration for the analysis because observations require personal (to a degree 'subjective') interpretation. Otherwise, someone could have argued on how the thesis author, who has also had personal experience as a team member/developer in the Nordea bank, remained objective and unbiased when providing the analysis, reporting and comments on the final research outcomes.

6.4. A Coherent Proposal for Further Study

The Scaled Agile Framework® provides an extensive scope for various stakeholders to explore from various angles in the future research and development. This research study could be extended to include participants from other similar or different organisations and compare and contrast the responses. A similar study could be conducted to gain insights of the following issues of importance in SAFe®: program, portfolio and value stream level members together with the team level members. Such studies should be conducted in the future in the organisations using SAFe® and belonging to different domains such as healthcare and automotive. It would be attention-grabbing to compare the banking sector stakeholders' responses to other sectors' stakeholders experiences with SAFe®. In addition, there is a worthy of note scope to study SAFe® on a global scale by studying the effectiveness of SAFe® in different cultures and countries. Studying also SAFe® by using longitudinal research design would be intriguing because it would probably result in producing a new knowledge base about the applicability and suitability of SAFe® and its effectiveness globally.

At practical level, the current study presented some challenges faced by the team members and their own ideas and reflections towards the framework's improvement. Future studies could focus on exploring these factors in detail and eventually the future researchers and metamodellers/developers could provide realistic and realisable solutions to overcome the existing challenges. Further research and development could be conducted on topics such as the following: better dependency management, refactoring PI events, embedding design into SAFe® processes or/and avoiding cognitive overload with amusing effectiveness of the meetings.Last, case studies on other available scaling frameworks such as Disciplined Agile Delivery and Large Scale Scrum should be accomplished in order to bring about new knowledge on their application; so the results of this case study can also be compared to those results.

6.5. Summarised Points and Lessons Learned: Using Scaled Frameworks for Agile Software Process Improvement in the Banking Sector

Agile software development is a popular approach that serves as a methodological paradigm shift utilised by many organisations in recent years. The agile software process is considered (and in some cases proved) to be beneficial over traditional development methods because it encourages flexibility, welcomes changing

requirements, involves customers and, most importantly, values developers/employees. The agile methodology, when applied in the context of large scale organisations, it is termed as “scaled agile”.

Large scale organisations are different from small scale co-located organisations. There are many factors such as distributed (sometimes geographically dispersed) teams, legacy systems, different automated tools and many processes used by sub-organisations. In order to overcome the challenges posed by large scale organisations and align the system as a whole, a standard full proof holistic (systemic) approach must be taken into account.

There are many frameworks available that claim to solve the scaling issues and commit to apply agility in large scale teams. Some of the most popular frameworks in application are: Scaled Agile Framework® (SAFe®), Disciplined Agile Delivery (DAD) and Large Scale Scrum (LeSS).

This study was focused on Scaled Agile Framework® which was introduced by Dean Leffingwell in 2011, to provide a solution for scaling agile development. Nordea Bank, which is one of the leading banking giants adopted SAFe® in year 2014. The creators and promoters of this framework state that it has several benefits such as alignment of large scale complex system towards a common mission, quality improvement, faster time to market and better/increased employee satisfaction.

Nordea bank is an example of a large scale organisation which is distributed in many locations, has multi-cultural teams and several legacy systems because it is an old organisation. Every sub section works using different technologies and development methodologies. In this scenario, it is hard to align everyone and introduce agility instantly. Most of the employees have been working therein for many years and have been using traditional development approaches and systems. Moreover, banking is not only about one product/service/process, but several processes and servicess combining in parallel a plethora of employees such as the: sales staff, customer care and financial advisors and the list can go on. Usability along with other quality considerations are of paramount significance to the banking sector. In addition, being involved in money and finances, it is essential to keep regular checking with legal rules and security issues [Mishra et al., 2019]. This is a classic example of a domain where the process of the software development highly impacts the final product/service quality and, in turn, this affects the overall profits and losses.

In recent era, bank (and other) customers want high quality products and prompt, usable and secure services. Thus, customers’ needs and convenience cannot be ignored by any organisation surviving in today’s competitive market. Agile methods are most suitable in that sense because they value customers' needs throughout the software development process. The SAFe® approach was introduced to Nordea because the main objective of the bank is to provide high end digitalised banking experiences for their valued customers in the form of web-based online services and mobile applications.

The SAFe® framework addresses the needs for such a complex banking organisation and provides a standard pathway for incorporating agility along with process and product/service quality.

The SAFe® framework divides the organisation into three or four level layers with specifies set of roles and responsibilities. The highest layer is the portfolio level, where high level business strategies and vision(s) are formulated. The value stream and program level members work on specific values and direct the team level members for accomplishing the business objectives. The team level acts like a foundation and consists of roles such as product owners, testers, developers and Scrum masters. The teams gather for the PI planning event and plan the features for upcoming iterations based on the business objectives. The management stakeholders are constantly involved during the planning sessions. The objective here is to create the plan, agree on the dependencies and assumptions, and identify the risks and issues associated with the plan. The management knows in advance about the plan and possible risks and are well prepared, accordingly. Later on, the product owner works with the teams regularly and facilitates the teams to attain the planned objectives. The shared demos and the tools provided by the framework enhance the transparency of the tasks carried out by all the teams. In short, no work is hidden and everyone at all the levels are well aware of the objectives and ongoing project progress. In order to ensure a quality product, the developers perform unit testing and peer reviews during a continuous integration process. The testers conduct system testing and, lastly, the product owner conducts acceptance testing. Furthermore, prior to market release, internal pilot testing is conducted within employees or some percentage of customers. These (sub)processes add to the layers of quality to the product when it reaches the hands of the customers. Summarising and concluding, the SAFe® approach as applied in the Nordea Bank case study optimised the whole process of development, project management, quality assurance so that the overall quality of product/process improved significantly. There is, thus, adequate evidence of the suitability of the same or similar scaled agile frameworks in the banking or similar application domains.

Additionally, in terms of teamwork, this study revealed worthy of note facts about the team members' perspectives regarding the framework. The most appreciated factors offered by SAFe® seem to be: structured planning, transparency, visibility of work, communication and coordination between teams and involvement of management and stakeholders. The challenges identified by the respondents are: the rigidity of the framework, lack of innovation possibilities, dependency management and lack of a proper design process. Notwithstanding, regardless the identified weaknesses in the applicability, in the holistic picture of SAFe® use and deployment, most of the respondents seem to be happy or neutral towards this framework. Right now, The framework's adoption registers a win situation within Nordea Bank's agile software

process and a successful choice to support and implement Nordea's IT and digitalization strategy for its banking products and services.

References

[Ambler, 2008] Scott W. Ambler, Agile software development at scale. In: Meyer B., Nawrocki J.R., Walter B. (eds), *Balancing Agility and Formalism in Software Engineering. Lecture Notes in Computer Science*, Vol **5082**, 2008, Springer, Berlin, Heidelberg.

[Ambler, 2012] Scott W. Ambler and MarkLines, *Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise*. *IBM Press (2012)* ISBN-13: 978-0132810135, 2012.

[Ambler, 2013] Scott W. Ambler, Going beyond scrum disciplined agile delivery, October 2013. Also available as *http://disciplinedagileconsortium.org/Resources/Documents/BeyondScrum.pdf*.

[Aydinli et al., 2016] Dilem Aydinli, Eleni Berki, Timo Poranen and Ioannis Stamelos, Management anti-patterns in IT innovation projects. In: *Proceedings of the 20th International Academic Mindtrek conference, ACM Proceedings,* 2016*,* pp. 1-10.

[Beck et al., 2001] Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland and Dave Thomas, Manifesto for Agile Software Development, 2001. Also available as *http://www.agilemanifesto.org*/.

[Berki et al., 2004] Eleni Berki, Elli Georgiadou and Mike Holcombe(2004), Requirements engineering and process modelling in software quality management – towards a generic process metamodel. *The Software Quality Journal*, Vol **12**, 2004, pp. 265-283. Also available as *https://link.springer.com/article/10.1023/B:SQJO.0000034711.87241.f0*.

[Berki, 2006] Eleni Berki, Examining the quality of evaluation frameworks and metamodeling paradigms of information systems development methodologies. Book chapter. Duggan, E. & Reichgelt, H.(Eds). *Measuring Information Systems Delivery Quality*, Idea Group Publishing:Hershey, PA, USA, Mar 2006. ISBN 159140858-X, pp. 265-289.

[Berki et al., 2007] Eleni Berki, K. Siakas and Elli Georgiadou 2007, Agile quality or depth of reasoning? Applicability vs. suitability with respect to stakeholder's needs. Book Chapter. Stamelos, I. & Sfetsos, P. (Eds), *Agile Software*

Development Quality Assurance. IRM Press and Idea Group Publishing: Hershey, PA. ISBN 978-1-59904-216-9. pp. 23-55.

[Bohem & Turner, 2005] B. Boehm and R. Turner**,** Management challenges to implementing agile processes in traditional development organizations. *Softw. IEEE,* Vol **22** (5), 2005, pp. 30-39.

[CA Agile Central] *https://www.ca.com/us/products/ca-agile-central.html*, Copyright © 2018 CA Technologies. All rights reserved.

[CA Flowdock] *https://www.ca.com/us/products/ca-flowdock.html*, Copyright © 2018 CA Technologies. All rights reserved.

[Checkland, 1999] Peter Checkland, *Systems Thinking, Systems Practice,* John Wiley, 2 Nov. 1999, 330 pages. Business and Economics.

[Chow & Cao, 2008] Tsun Chow and Dac-Buu Cao**,** A survey study of critical success factors in agile software projects. *Journal of Systems and Software,* Vol **81**, 2008, pp. 961-971.

[Cristal et al., 2008] Mauricio Cristal, Daniel Wildt, and Rafael Prikladnicki, Usage of scrum practices within a global company. *In Global Software Engineering, 2008. ICGSE 2008. IEEE International Conference*, pp. 222–226.

[Dikert et al., 2016] Kim Dikert, Maria Paasivaara and Casper Lassenius**,** Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software,* Vol **119**, 2016, pp. 1869-1890.

[Dingsøyr & Moe, 2013] Dingsøyr Torgeir and N.B. Moe, Research challenges in large-scale agile software development. *ACM Software Engineering Notes 2013.* Vol **38***(*5*),* pp. 38-39.

[Dingsøyr & Moe, 2014] Dingsøyr Torgeir and N.B. Moe, Towards principles of large-scale agile development. In: Dingsøyr T., Moe N.B., Tonelli R., Counsell S., Gencel C., Petersen K. (eds). *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation. XP 2014. Lecture Notes in Business Information Processing,* Vol 199, 2014, Springer, Cham.

[Dingsøyr et al., 2014] Dingsøyr Torgeir, Tor E. Faegri and Juha Itkonen, What is large in large-scale? A taxonomy of scale for agile software development. In:

Jedlitschka A., Kuvaja P., Kuhrmann M., Männistö T., Münch J., Raatikainen M. (eds). *Product-Focused Software Process Improvement. PROFES 2014. Lecture Notes in Computer Science*, Vol **8892**, 2014, Springer, Cham.

[Elliot, 2000] J. J. Elliot, Design of a product-focused customer-oriented process. *Information and Software Technology*, Vol **42** (14), 2000, pp. 973-981.

[Eriksson & Törn, 1991] Inger Eriksson and Aimo Törn, A model for IS quality, (information systems). *Software Engineering Journal*, Vol **6**(4), pp. 152-158. Also available as *https://ieeexplore.ieee.org/document/87372/*.

[Galliers & Land, 1987] Robert D. Galliers and Frank F. Land, Choosing appropriate information systems research methodologies. *Communications of the ACM*, Vol **30**, 1987.

[Glass, 2001] R. Glass R, Agile versus traditional: Make love, not war. *Cutter IT Journal*, December, Vol **6**(1), 2001, pp. 12–18.

[Highsmith, 2002] Jim Highsmith, *Agile Software Development Ecosystems.* Addison Wesley, 2002.

[Hummel et al., 2013] Markus Hummel, Christoph and Rosenkranz and Roland Holten, The role of communication in agile systems development: An analysis of the state of the art. *Business and Information Systems Engineering* Vol **5**(5), 2013, pp. 343–355.

[Kauffeld and Lehman W., 2012] Simone Kauffeld and Nale Lehmann-Willenbrock., Meetings matter effects of team meetings on team and organizational success. *Small Group Research*. Vol **43**, 2012, pp. 130-158.

[Knaster and Leffingwell, 2015] Richard Knaster and Dead Leffingwell, *SAFe 4.0 Distilled: Applying the Scaled Agile Framework for Lean Software and Systems Engineering,* 1st Edition, Addison-Wesley Professional, pp. 384, 2015.

[Kothari, 2008] C. R. Kothari, *Research Methodology Methods and Techniques*, New Delhi, New Age International, 2008.

[Kunwar, 2013] Sundar Kunwar, Metamodelling and evaluating eXtreme programming. MSc thesis, 2013. Also available as *https://tampub.uta.fi/bitstream/handle/10024/84754/gradu06859.pdf?sequence=1*.

[Laanti, 2014] Laanti Maarit, Characteristics and principles of scaled agile. In: Dingsøyr T., Moe N.B., Tonelli R., Counsell S., Gencel C., Petersen K. (eds). *Agile Methods. Large-Scale Development, Refactoring, Testing, and Estimation. XP 2014. Lecture Notes in Business Information Processing,* Vol **199**, 2014, Springer, Cham.

[Larmen & Vodde, 2017] Craig Larman and Bas Vodde (2017), *Large-Scale Scrum: More with LeSS,* Addison-Wesley professional, 2017.

[Layman et al., 2006] Lucas Layman, Laurie Williams, Daniela Damian, and Hynek Bures, Essential communication practices for extreme programming in a global software development team. *Information and Software Technology,* Vol **(489)** (2006), pp. 781-794.

[Leffingwell et al., 2016] Dean Leffingwell, Alex Yakyma, Richard Knaster, Drew Jemilo, Inbar Oren, *SAFe® 4. 0 Reference Guide: Scaled Agile Framework® for Lean Software and Systems Engineering*, Addison-Wesley Professional, 2016.

[Leffingwell, 2016] SAFe 4.0, Introduction overview of the scaled agile framework® for lean software and systems engineering, A Scaled Agile, Inc. White Paper, July 2016.

[Li, 2012] Jinjin Li, Agile software development, Technics University Berlin, Berlin, Germany, 2012. Also available as *https://www.snet.tu-berlin.de/fileadmin/fg220/courses/WS1112/snet-project/agile-software-development_li.pdf*.

[Li et al, 2013] Linfenf Li, Eleni Berki, Marko Helenius & Saila Ovaska, Towards a contingency approach with whitelist and blacklist-based anti-phishing applications: what do usability tests indicate? *Behaviour and Information Technology, BIT* , Vol **33**(11), pp. 1136-1147.

[Luong & Rogelberg, 2005] Loung Alexandra and Steven G. Rogelberg, Meetings and more meetings: The relationship between meeting load and the daily well-being of employees. *Group Dynamics: Theory, Research, and Practice*, Vol **9**, 2005, pp. 58-67.

[Maaninen & Berki, 2004] Ari Manninen and Eleni Berki, An evaluation framework for the utilisation of requirements management tools - maximising the quality of organisational communication and collaboration. In: Edgar-Nevill, D., Ross, M. & Staples, G. (Eds). *New Approaches to Software Quality. Software Quality*

Management XII. Proceedings of BCS Software Quality Management 2004 Conference, University of KENT, Canterbury, 5-7 Apr, pp. 139-160. ISBN 1-902505-56-5.

[Mishra et al, 2019] Priya Mishra, Eleni Berki, Juri-Petri Valtanen, Teemu Lepistö, Tomi Suomela, Quality deployment and use of the Scaled Agile Framework® – managing teamwork and software quality in the banking sector. In the Proceedings of Ross, M. Staples, G. (Eds) the British Computer Society *Software Quality Management 2019 International Conference, Solent University, Southampton*, Apr 2019. (Pending).

[Misra et al., 2009] Subhas C. Misra, Vinod Kumar and Uma Kumar**,** Identifying some important success factors in adopting agile software development practices. *Journal of Systems and Software,* Vol **82**, 2009, pp. 1869-1890.

[Nakakoji et al., 2010] Kumiyo Nakakoji, Yunwen Ye and Yasuhiro Yamamoto, Supporting expertise communication in developer-centered collaborative software development environments. In: Mistrík I., Grundy J., Hoek A., Whitehead J. (eds) *Collaborative Software Engineering*. Springer, Berlin, Heidelberg, 2010.

[Nelson et al., 2000] Kay M. Nelson, Deb Armstrong, Mari Buche and Mehdi Ghods, Evaluating the CMM level 3 KPA of intergroup coordination: A theory-based approach. *Information Technology and Management (Jan 2000),* Vol **1**, 2000, pp. 171-181.

[Nerur et.al, 2005] Sridhar Nerur, Radhakanta Mahapatra and George Mangalaraj Challenges of migrating to agile methodologies, *Communications of the ACM*, Vol **48** (5), 2005, pp. 72-78.

[Nordea Case study, 2017] Nordea A Uniform Heartbeat with Help from Scaled Agile Framework® and IJI Copyright © 2015 Ivar Jacobson International. Also available as http://www.scaledagileframework.com/wp-content/uploads/delightful-downloads/2017/09/Nordea-Case-Study.pdf.

[Nordea Bank website] Nordea Bank Website, *https://www.nordea.fi/*.

[Poppendieck and Poppendieck, 2003] Mary Poppendieck and Tom Poppendieck, *Lean Software Development – An Agile Toolkit for Software Development Managers*, Addison-Wesley, Boston, 2003.

[Raut et al., 2015] Laukik Raut, Rajat B. Wakode and Pravin Talmale, Overview on kanban methodology and its implementation. *International Journal for Scientific Research & Development,* Vol **3**, 2015, pp. 2518-2521.

[Rasmusson, 2010] Jonathan Rasmusson, *The Agile Samurai: How Agile Masters Deliver Great Software*, The Pragmatic Bookshelf, Raleigh, Texas, 2010. Also available as *http://file.allitebooks.com/20160410/The%20Agile%20Samurai.pdf*.

[SAFe®/Scaled Agile Framework®] Scaled Agile Framework® web pages, *https://www.scaledagileframework.com/* (*Last accessed June 2018).*

[Schatz & Abdelshafi, 2005] Bob Schatz and Ibrahim Abdelshafi, Primavera gets agile: A successful transition to agile development. *IEEE Software*, Vol **22**(3), 2005, pp. 36-42.

[Shalloway et al., 2010] Alan Shalloway, Guy Beaver, and James R. Trott, *Lean-Agile Software Development: Achieving Enterprise Agility*. Boston, MA: Pearson Education, 2010.

[Soloveva, 2017] Elena Soloveva. A remedy perspective for emerging antipatterns in agile software projects in small and medium enterprises. MSc thesis, 2017. Also available as http://tampub.uta.fi/bitstream/handle/10024/102551/1513599675.pdf?sequence=1.

[Stamelos & Sfetsos, 2007] I. Stamelos & Panagiotis Sfetsos. *Agile Software Development Quality Assurance,* 2007*,* IRM Press and Idea Group Publishing: Hershey, PA. ISBN 978-1-59904-216-9.

[Stray et al., 2012] Viktoria G. Stray, Nils B. Moe, and Aybuke Aurum, Investigating daily team meetings in agile software projects. *38th EUROMICRO Conference on Software Engineering and Advanced Applications*, 2012, pp. 274-281.

[SurveyPlanet, 2017] *https://surveyplanet.com/*, (*Last accessed June 2018*).

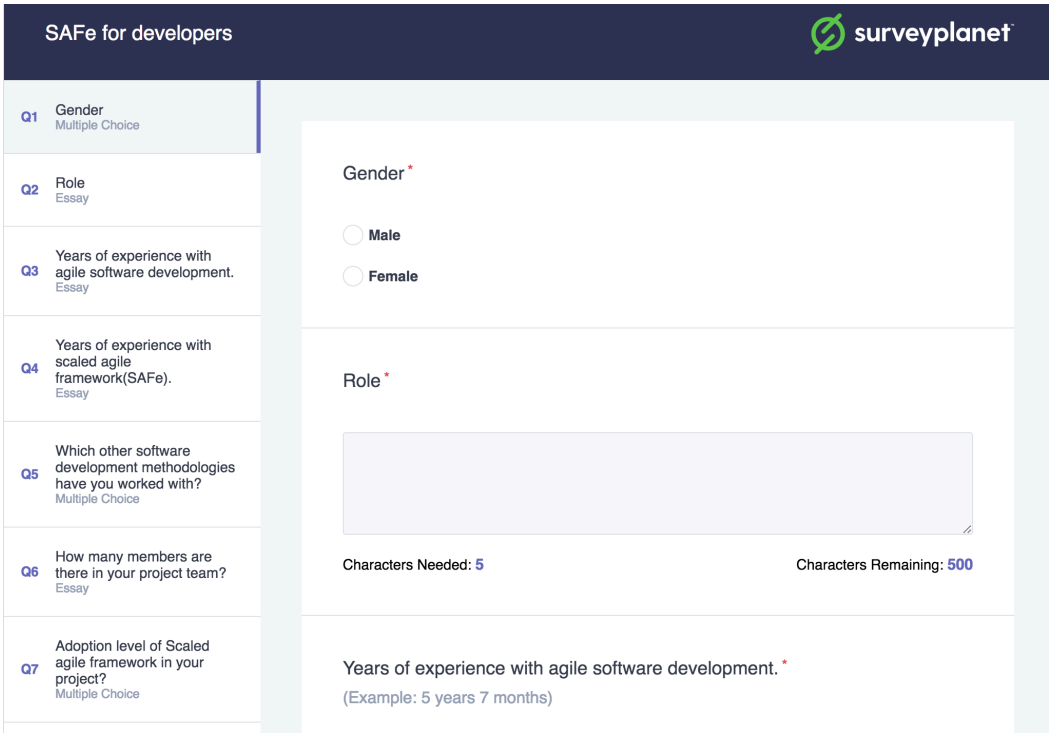
[Taymor, 2013] Emerson Taymor, (2013). *Agile Handbook*, 2013. Also available as *http://agilehandbook.com/agile-handbook.pdf.*

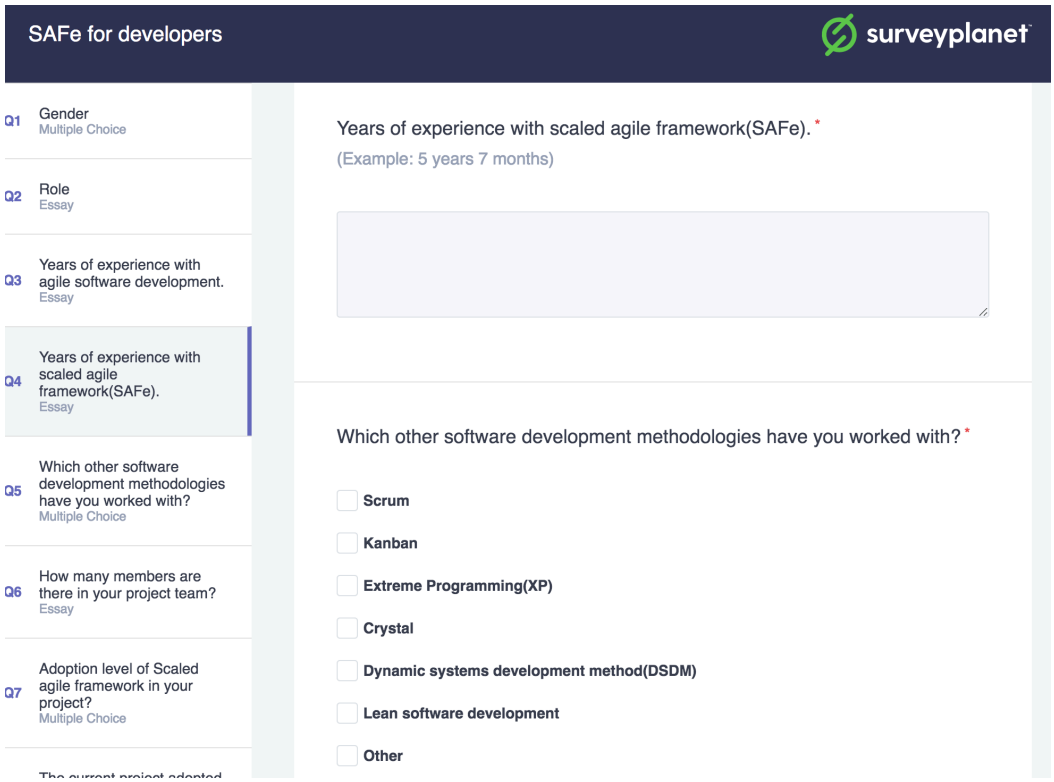
[Wears et, al. 2007] Robert L. Wears, Shawna J. Perry, Stephanie Wilson, Julia Galliers and James Fone, Emergency department status boards: user-evolved artefacts for inter- and intra-group coordination, *Cogn Tech Work*, Vol **9**, 2007, pp. 163-170.

[Whitworth & Biddle, 2007] Elizabeth Whitworth and Robert Biddle, The social nature of agile teams. *Proceedings of the AGILE 2007*, 2007, pp. 26–36.

[Williams and Cockburn, 2003] Laurie Williams and Alistair Cockburn, Agile software development: It's about feedback and change. *IEEE Computer*, Vol 36, 2003, pp. 39-43.

Appendix


SAFe for developers
surveyplanet
Q1 Gender
Multiple Choice
Q2 Role
Essay
Q3 Years of experience with agile software development.
Essay
Q4 Years of experience with scaled agile framework(SAFe).
Essay
Q5 Which other software development methodologies have you worked with?
Multiple Choice
Q6 How many members are there in your project team?
Essay
Q7 Adoption level of Scaled agile framework in your project?
Multiple Choice
Gender *
Male
Female
Role *
Characters Needed: 5
Characters Remaining: 500
Years of experience with agile software development. *
(Example: 5 years 7 months)


SAFe for developers
surveyplanet
Q1 Gender
Multiple Choice
Q2 Role
Essay
Q3 Years of experience with agile software development.
Essay
Q4 Years of experience with scaled agile framework(SAFe).
Essay
Q5 Which other software development methodologies have you worked with?
Multiple Choice
Q6 How many members are there in your project team?
Essay
Q7 Adoption level of Scaled agile framework in your project?
Multiple Choice
Years of experience with scaled agile framework(SAFe). *
(Example: 5 years 7 months)
Which other software development methodologies have you worked with? *
Scrum
Kanban
Extreme Programming(XP)
Crystal
Dynamic systems development method(DSDM)
Lean software development
Other

SAFe for developers

surveyplanet

Q1 Gender
Multiple Choice

Q2 Role
Essay

Q3 Years of experience with agile software development.
Essay

Q4 Years of experience with scaled agile framework(SAFe).
Essay

Q5 Which other software development methodologies have you worked with?
Multiple Choice

Q6 How many members are there in your project team?
Essay

Q7 Adoption level of Scaled agile framework in your project?
Multiple Choice

Rate the ease of communication with other team members (scrum master, testers, designers, other experts). *

- Poor
- Below average
- Average
- Above average
- Excellent

If you find it tough to coordinate with other team members, can you elaborate the reason?

(In case you find it convenient just mark NA)

Characters Needed: 2 Characters Remaining: 500