SELECTING A SCALED AGILE APPROACH FOR A FIN-TECH COMPANY

L.A. Christopher[1*] & M. de Vries[1]

ARTICLE INFO

Article details
Presented at the 31st annual conference of the Southern African Institute for Industrial Engineering (SAIIE), held virtually from 5-7 October 2020.

Available online 11 Nov 2020

Contact details
* Corresponding author
laurenalisachristopher@gmail.com

Author affiliations
1 Department of Industrial and Systems Engineering, University of Pretoria, South Africa

ORCID® identifiers
L.A. Christopher
https://orcid.org/0000-0003-4782-9601

M. de Vries
https://orcid.org/0000-0002-1715-0430

DOI
http://dx.doi.org/10.7166/31-3-2432

ABSTRACT

Agile methodologies were created for small co-located teams; but their benefits are now needed for projects with multiple, geographically dispersed teams. This raises the issue of scaled Agile — the modification of Agile to suit grander endeavours. This article introduces a fin-tech company which is building a digital banking platform and is already running Scrum, with the crucial need to scale its Agile approach because numerous globally distributed development teams are working on this project. A systematic inquiry at the enterprise, using interviews and surveys, raised issues about inter-team communication, collaboration, and dependencies, all of which link to issues in the literature about scaling Agile. Using a systematic process of extracting knowledge on Agile-at-scale frameworks from the literature, a comparison of existing frameworks (SAFe, Less, DAD, and Nexus) was used to determine which was best suited, with Nexus being suggested for the enterprise. We conclude by suggesting case study research to experiment further with Nexus.

OPSOMMING

Ratse metodologieë is geskep vir klein, naby-geleë spanne, maar hul voordele is nou nodig vir projekte met meerdere, geografies verspreide spanne. Daar is dus ŉ behoefte om ratse metodologieë aan te pas om grootskaalse projekte te akkommodeer. ŉ Finansiële-tegnologie maatskappy wat ŉ digitale bankplatform ontwikkel, en reeds van Scrum gebruik maak, word in die artikel bekendgestel. Die maatskappy het ŉ dringende behoefte om sy bestaande ratse benadering op te skaal omdat daar veelvuldige, globaal verspreide ontwikkelingspanne aan dié projek werskaf. ŉ Sistematiese navraag by die maatskappy, deur middel van onderhoude en vraagstukke, het uitdagings met die tussenspan-kommunikasie, samewerking en afhanklikheid uitgelig. Hierdie uitdagings is bekende tekortkominge van grootskaalse ratse benaderings. Deur middel van ŉ sistematiese proses om die kennis van grootskaalse ratse metodologieë vanuit die literatuur te onttrek is daar ŉ vergelyking getref tussen bestaande raamwerke (SAFe, Less, DAD en Nexus). Nexus is identifiseer as die mees geskikte raamwerk. Laastens word ŉ gevallestudie voorgestel om die saak verder, met die gebruik van Nexus, te ondersoek.

1 INTRODUCTION

With the ever-increasing demand for the world to stay connected, software development is becoming a necessity for business in today's market [1]. This industry is ever-changing, and conventional development methodologies, such as the *waterfall* methodology, are diminishing [2]. Their long-term planning and structure rendered projects useless, because the market had already changed between the time that planning had started and when development was initiated [3]. Traditional methods, based on hierarchy and bureaucracy, slow down decision-making [1]. Agile, created to combat problems associated with traditional methods, has been in the workplace since the 1990s [3, 4]. Quick adaptability and the 'fail-fast' approach made Agile popular in software development [1].

Agile encourages face-to-face interaction between team members (including customers), rather than using tools and technology [1, 3]. It emphasises development and product quality over extensive documentation, self-organising teams using their members' strengths, and the adaptability to change using constant iteration [1, 3]. Agile has changed since its inception, it was initially created around ideologies such as *software before documentation*; now taking into account more practical aspects and considering business factors, such as governance [4], it has been the base for frameworks such as Scrum, Kanban, and Extreme Programming (XP) [1, 5]. Agile projects are ideal when teams are small, self-organising, co-located, and where customers are on-site; any issues can be communicated and dealt with almost immediately [5]. In 2002 the scalability factor within Agile was raised, becoming a focus topic [3]. In a consistently changing, technologically driven world with a high demand for software development, the reality is that large globally distributed teams working on major projects need Agile methodologies to optimise their project management [4]. Some scaled Agile frameworks have been developed, such as Scrum of Scrums (SoS), Scaled Agile Framework (SAFe), and Nexus [1, 6, 7].

This study was initiated by its principal researcher, a business analyst working at a highly tech-driven fin-tech company (referred to as *FinTechEnterprise*), working on their biggest product development project in the last three years. This product, their digital banking platform (DBP), raised the need to evaluate the way in which Scrum was implemented and how to scale it, using FinTechEnterprise's environment and this project as a 'real-world' case study. This paper focuses on the challenges of adapting Scrum to suit a large group involving globally distributed teams; using literature-based scenarios and results, as well as using the interview results of experts in the field. The study extracts factors to be considered when selecting a scaling Agile framework, indicating their contribution to elevating the issues within the DBP project.

The article is structured as follows. Section 2 presents the research method, which used problem instance validation and a systematic literature review (SLR). Section 3 inquires into FinTechEnterprise and issues regarding the project, with Section 4 using the SLR results to propose a solution. Section 5 summarises the article, looking at a way forward. Section 6 provides the acknowledgements, and Section 7 provides the references.

2 RESEARCH METHOD

Yin's [8] guidance on case study research states that using method and data triangulation increases the validity of claims — e.g., using both qualitative and quantitative methods, and using multiple data sources. This section elaborates on the qualitative and quantitative research methods that were followed in gathering data, which were chosen after research and analysis of the various methods that are commonly used. The chosen methods were used to analyse the problem instance at FinTechEnterprise, and to perform a systematic review of the existing literature, in the search for a solution.

2.1 Problem instance validation

Diagnosis techniques are not well-established for Agile frameworks, with no commonly used methods to evaluate the chosen method [9]. Therefore, two generic problem diagnosis strategies were used for this study: (1) root-cause analysis (RCA), and (2) systems dynamics using a *causal-loop-diagram* (CLD). An RCA identifies, analyses, and assists in understanding the underlying reason for the issues being faced, leading to the creation, testing, and monitoring of a solution [10, 11]. There are various RCA methods [10]. The *five whys* was chosen owing to its quick and efficient evaluation of the current enterprise and identification of the root causes. A CLD visually demonstrates the effects that certain variables have on each other [12], using arrows to show links and direction, with symbols of polarity, for either a positive or a negative influence [13]. Several data-gathering strategies were used to triangulate the analysed results, including interviewing, participant observation, and a survey.

Observation was used because the primary researcher was working on the DBP (digital banking platform) project and was therefore able to benefit, as stated by Kawulich [14] — i.e., to detect nonverbal cues, such as emotions, that might not be communicated in a confrontation-avoiding interview. This allowed for a deeper understanding from the interviews, relating and critically assessing what was said, as there was a better sense of the environment.

Interviews conveyed multiple viewpoints — i.e., both business and development perspectives. Interviewing is a qualitative data collection method that follows the general process of a pre-interview contact to ask and inform the interviewee about the interview, drafting questions, conducting the interview, and transcribing [15]. Interviews assist in understanding personal opinions and experiences within the environment [16]. The one-on-one interviews between the researcher and the interviewees were conducted in closed rooms for privacy. Following Guest, MacQueen and Namey's [17] guidelines on thematic analysis, a structural codebook was used to cover themes such as the work environment, pros and cons, concerns, and suggested solutions for the DBP project. Several project participants were interviewed, including leading employees from both business and development. For a *business perspective*, the first interviewee was the *product development lead*; the second was the *lead technical manager* who heads the technical side of the project. These individuals were responsible for making key decisions in this project, and so understood the project in its entirety. The *senior business analyst* who was interviewed exercised the role of connecting business requirements with development during the system and process design. Two *scrum masters* involved in this project (scrum master 1 and scrum master 2) were interviewed who had implemented and championed Scrum in the company, and who would lead the scaling of Agile.

A *survey* was sent to an external scrum master (scrum master 3) who was not involved in FinTechEnterprise and was included purely to provide Agile knowledge. A survey is a fixed list of questions to participants, and is a standardised method of collecting data [16]. The e-mailed survey allowed the participant to answer the questions honestly, without the added pressure of time or interview prompting. Scrum master 3 has experience in DAD and scaled Agile framework implementations, enabling him to provide an expert opinion on Agile and scaling. The questions in the survey were based on experiences of Agile in general, and were not about FinTechEnterprise. Unlike the information gathered via interviews on the context-specific use of Agile, where emotions and nonverbal cues were accessible, the survey was used simply for additional cross-validation.

2.2 Systematic enquiry for a solution

To identify a possible solution from the literature, a systematic literature review (SLR) was used. The research protocol for the SLR was the eight-step procedure of Okoli and Schabram [18]: (1) the purpose of the SLR, (2) protocol and training, (3) searching the literature, (4) practical screening, (5) quality appraisal, (6) data extraction, (7) synthesis of the study, and (8) writing a review. The *purpose* of this SLR was to analyse Agile frameworks, extracting *factors that need to be considered when choosing an appropriate framework*, including lessons learnt from existing implementations.

The *research protocol* was defined in terms of a key word selection, data sources, criteria for practical screening, and criteria for quality appraisal, following Okoli and Schabram [18]. The *key words* were incorporated in a single search string — i.e., Kw:("Scaling Scrum"OR"Scrum"OR"Scaling Agile"OR"Scaled Agile Framework"OR"Disciplined Agile Delivery"OR"Nexus"OR"Large-Scale Scrum"OR"Agile"OR"Large-Scale Agile"). For the protocol of *data sources*, we included the University of Pretoria's online library, and books and journals from reliable electronic databases: World Cat, IEEE Xplore, and SpringerLink.

Practical screening included peer-reviewed online articles written in English, and English-language books that contained any of the key words in their titles. The peer-review criteria were included only for the initial research, and not for data obtained from backward snowballing. Only research from the year 2000 onwards was considered, as the scalability of Agile emerged as a topic in 2002 [3], thus requiring screening a few years prior to it for an understanding of its origins. Any non-English documentation or research with no connection to the key words used for this SLR was excluded. Since a high number of free sources were available, any sources that needed a payment and any duplicate articles were excluded immediately. For the primary sources found, *any non-peer-reviewed* online articles were also excluded. From the selected extracted *articles*, backwards snowballing was completed. According to titles, keywords, and abstracts, including the language and year range from the *inclusion* and *exclusion* criteria, articles related to the research questions and key words were selected for further consideration. Articles, blogs, and websites found as a result of backward snowballing, although not peer-reviewed, were also considered. Non-peer-reviewed sources were included, as they were used to create content for peer-reviewed articles, and could therefore be considered reliable sources. A comprehensive but focused pool of research was thus created.

The *quality appraisal* process was taken from Kitchenham *et al.* [19], using the evaluation process from York University's Centre for Reviews and Dissemination (CDR) and the Database of Abstracts of Reviews of Effects (DARE) criteria. The quality assessment, based on four questions, was answered by 'yes', 'partially', or 'no', giving a score per answer of 1, 0.5, and 0 respectively [19]. Each source, shortlisted via the practical screen, was scored, excluding sources with an aggregated score lower than 2. This allowed for the quality of the sources to be compared with each other, and ensured that all sources were of the same high calibre. The four quality assessment questions were:

1. Is the selected article clearly focused on Agile and its frameworks?
2. Is the scalability of Agile clearly addressed?

3. Is there reference to frameworks/approaches that can be used in order to utilise Agile in a large-scale operation (means of scaling Agile up)?
4. Is there empirical evidence that a suggested framework/approach (as a solution) sufficiently addresses some of the problematic factors associated with Agile?

No source scored lower than 2. The four sources that scored 2 were only used for the SLR results, which validated a *class-of-problems* exist, as they scored in the first two questions containing information on Agile and touching on the concept of scalability, but not suggesting frameworks/approaches solutions. The 24 sources that scored above 3 were considered for the SLR results that related to *suggested solution areas,* as they included frameworks/approaches as solutions.

The data extraction process, which began with an initial search method using the University of Pretoria's online library and keywords, is illustrated in Figure 1. The inclusion criteria (from the practical screen protocol) were then used to indicate date range and peer-reviewed articles only. Owing to the high number of sources and wide range of keywords, as well as researcher capacity constraints, only the first five pages of results per keyword search were considered. Since the result set indicated little empirical data on scaled Agile and their frameworks (in terms of comparisons of features or evaluation metrics/tools [4, 20, 21, 22, 23, 24]), lower quality sources (blogs and web pages) were also included to ensure a more comprehensive coverage of work from practitioners and theorists.

A *data extraction form* captured standard article information (author, title, year of publication, and publication type), following Kitchenham [25], and the extracted codebook themes related to the extracted source were based on Guest, MacQueen and Namey [17]. A second analyst was involved during the coding process, thus improving the reliability of the process. The primary researcher adapted the intercoder agreement guidelines from Guest *et al.* [17], created the codebook, and then chose three sources *(*the first three pages) to provide training to the second analyst. The primary researcher created the master set of data coding, while the second analyst created a secondary set of data coding, and the two were compared. The percentage of agreement for all the codes was 85 per cent. An intercoder agreement of 80 per cent or higher is considered acceptable [17], and therefore no adjustments were made to the codebook. The main source topics were either non-scaled Agile (NSA) or scaled Agile (SA), with the extracted themes being 'AgileOrigin', 'OriginFW', 'ScaleNeed', 'ScaledFW', 'ScaledFactors', 'ScaledPros&Cons', and 'ScaledExamples'.

The *data synthesis* was completed in a qualitative and quantitative manner, and represented graphically. These representations assisted with comparing the amount and type of data collected regarding the class-of-problems and its solutions, simplifying the data analysis. The theme frequency in a source was taken as '1', regardless of how many times it occurred in that source [17]. The quantitative data was investigated by mapping the year of publication against the extracted themes, revealing that, even though scaling Agile was recognised as early as 2003, it began to be acknowledged from 2010, which correlates with scaling Agile being raised as a top research topic at the 2010 XP conference [22]. The total number of sources per theme were 13, 18, 18, 23, 18, 20, and 14 for 'AgileOrigin', 'OriginFW', 'ScaleNeed', 'ScaledFW', 'ScaledFactors', 'ScaledPros&Cons', and 'ScaledExamples' respectively. Figure 2 shows that 39 per cent of the data fell under the non-scaled Agile topic and 61 per cent under scaled Agile; the largest percentage related to the general mentioning of various scaled frameworks.

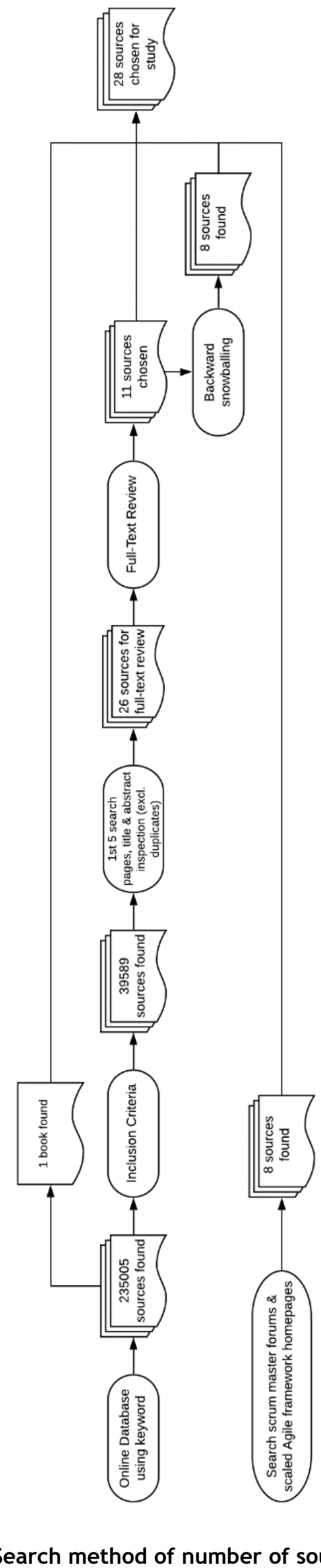
Online Database using keyword
235005 sources found
1 book found
Inclusion Criteria
39589 sources found
1st 5 search pages, title & abstract inspection (excl. duplicates)
26 sources for full-text review
Full-Text Review
11 sources chosen
Backward snowballing
8 sources found
28 sources chosen for study
Search scrum master forums & scaled Agile framework homepages
8 sources found

Figure 1: Search method of number of sources found

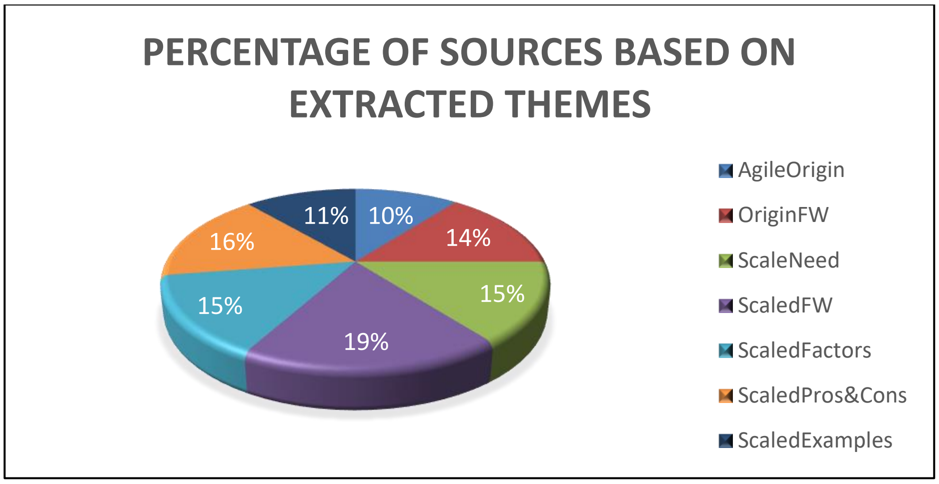

Figure 2: Percentage of sources, based on extracted themes

Figure 3 graphically represents the data synthesis in a qualitative manner, showing the number and type of sources per extracted theme. The quality of the sources, grouped into 'high', 'medium', and 'low', is based on the type of publication used.

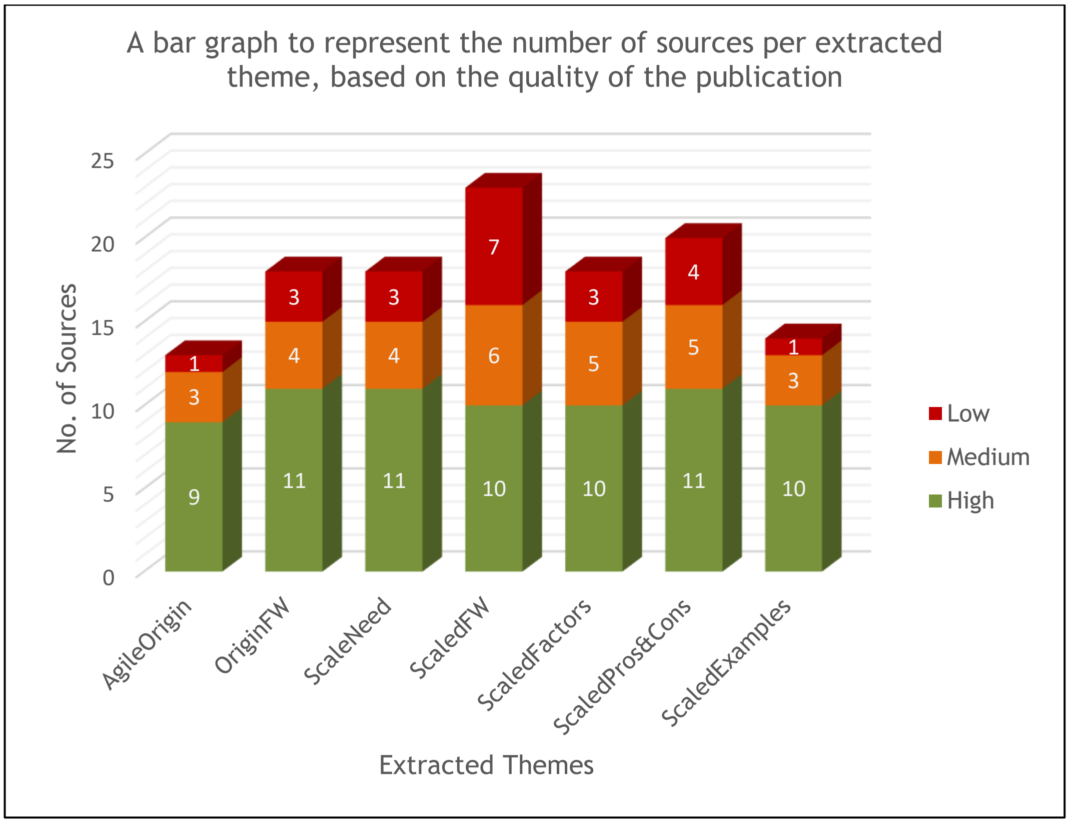

Figure 3: Quality of data sources per extracted theme

High-quality publications were regarded as peer-reviewed articles/documents, books, or Master's theses, because they were academically reviewed. Medium-quality sources were non-peer-reviewed conference papers, guides, reports, case studies, and white papers, because these might not have been reviewed, but might have evidence to support their claims, or were used as the standard for implementation/use (in the case of guidelines). The low-quality sources were from blogs or web pages, with no validation or review by anyone, and with no academically supported evidence.

3 PROBLEM INQUIRY RESULTS

FinTechEnterprise was already using Scrum, and the evident benefits included the following: team members were working well together achieving regular incremental releases; and there was a noticeably positive change from the initial *waterfall* system development approach. However, with a Scrum implementation that was focused heavily on the teams, factors outside those teams were not considered. Bottlenecks were not dealt with — an issue, according to scrum master 1, that was resolved by the creation of a leadership board that stated:

“The leadership board was an overarching board that was placed just before the product backlog where all our initiatives for the next couple of years would be listed and recorded. This gave us an additional redundant step because we were just creating silos again.”

Using the interview transcripts, the ‘five whys’ method was performed in order to identify the root cause (Figure 4).

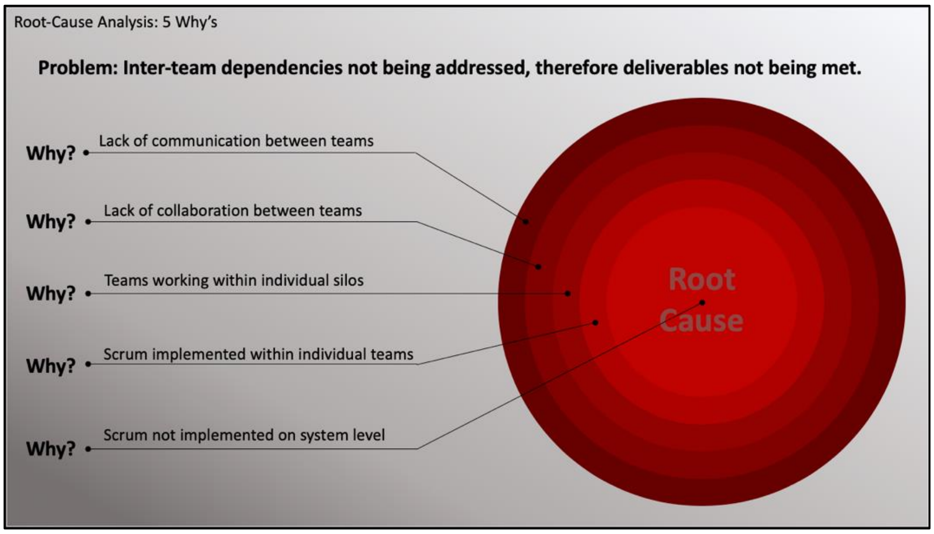
Root-Cause Analysis: 5 Why's
Problem: Inter-team dependencies not being addressed, therefore deliverables not being met.
Why? Lack of communication between teams
Why? Lack of collaboration between teams
Why? Teams working within individual silos
Why? Scrum implemented within individual teams
Why? Scrum not implemented on system level
Root Cause

Figure 4: Five whys results represented visually

The ‘five whys’ originated in the main concern that inter-team dependencies were a weakness in this build, as mentioned by the lead technical manager. This led to the first ‘why’ addressing the lack of communication between the teams, as stated by the product development lead:

“…the inter-team dependency though was where things started to break down a little bit.”
“We had many teams, all working in isolation, there were dependencies that they had amongst each other; but those dependencies weren’t clearly communicated and understood, as a result impacted their own deliverables.”

Teams were not communicating, and so inter-team dependencies were not known; this led to the second ‘why’, the lack of collaboration — also raised in the quote from the product development lead. With teams working in isolation, there would be very little or no collaboration, resulting in the teams working in silos, as noted earlier by scrum master 1, resulting in the third ‘why’. The fourth ‘why’ brought clarity and reasoning to the isolation and lack of collaboration, as Scrum was executed in the company at an individual team level, as stated earlier by scrum master 1 and pointed out by a senior business analyst:

“We had Scrum within our small groups, that’s about it.”

The final ‘why’, identifying the root cause, was that the Scrum implementation was not system-based, but rather had a team-based foundation. This was validated by scrum master 1 and confirmed by scrum master 3:

“What didn’t work was that we realised that Scrum, although many people think it’s a team-based thing, it’s actually a systems-based thing. If you fixed the team, or you improve them with Scrum, you’re going to hit a glass ceiling at some point because you can only optimise them so much before the organisational gravity pulls them back into bad habits or impacts their productivity. So, we optimised the team and once the team reached a certain level of productivity, we realised that factors outside of the team started impacting the team so that they couldn't grow more. They reached an optimum productivity level and then it kind of dipped because there wasn't any motivation. That was one of the big things we had to fix. Scrum wasn't only a localised team thing; it was a much bigger problem. If you wanted to fully tap into the power and benefits of Scrum, you needed to look at the system as well, to look at bottlenecks before and after your function."

"If teams adopt Agile as a framework without the understanding and support of the organization, the structures and processes within the organization will not change to support the implementation, and the initiative will struggle to survive — teams cannot work in isolation, it must be a system wide adoption."
Using the CLD (Figure 5), the variables of *knowledge of the inter-team dependencies*, *collaboration*, and *communication among the teams* are linked. They all impact each other positively in a reinforcing loop. Increasing communication and collaboration would therefore increase knowledge of the inter-team dependencies.

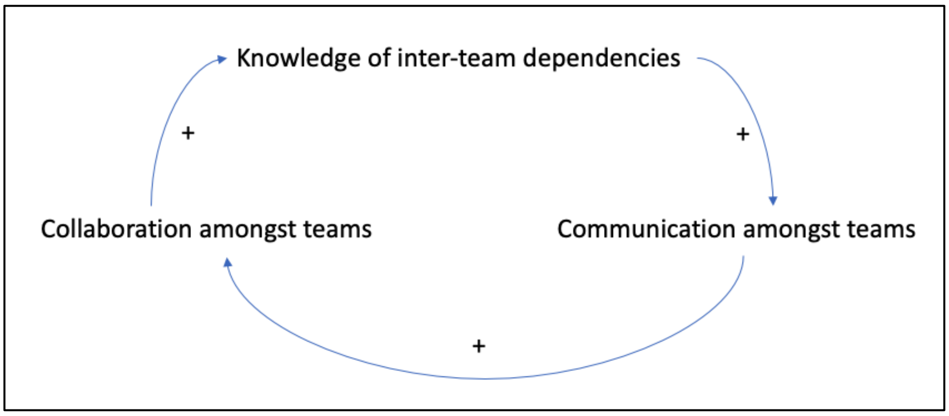
Knowledge of inter-team dependencies
+
+
Collaboration amongst teams
Communication amongst teams
+

Figure 5: Causal loop diagram

Teams worked in silos because the Scrum implementation was team-focused, resulting in good team work but a lack of knowledge outside their own projects, and a lack of transparency between teams. As scrum master 2 explained:

"Also, it was nice because teams were going along nicely but we had no visibility of what everyone was doing, so this team would deliver something, but other teams didn't know what they were delivering."

The banking build exposed the lack of clarity in the environment, and showed that inter-team dependencies were not catered for, as raised by the product development lead and the lead technical manager:

"We had many teams, all working in isolation, there were dependencies that they had amongst each other; but those dependencies weren't clearly communicated and understood, as a result impacted their own deliverables."

"... the inter-team dependency though was where things started to break down a little bit. It was hard to coordinate between teams, especially when deliverables were due, and a team was dependent on two or three other teams. They might be able to work directly with one team, because we have two teams in India and then a couple teams in South Africa; if you were here and you were dependent on everyone here it was relatively easy to get things going. If you were dependent on teams in India, then you were locked to their dev cycle; they had already decided what they were going to do in their sprint before, they had no idea that you were looking for anything from them."

Teams were also globally distributed between India, South Africa, and Mauritius, increasing the project's complexity. Having many teams working together, building one product, was something that needed to be addressed, but was not considered in the current system. When asked what caused the company to consider a scaled solution, the product development lead stated:

"The size of the project, and the many teams involved, there were 6 systems we needed to build … It [Scrum] made sense for what they were doing at that point in time. Small, minor, changes on systems, focused on them and executed them, not a lot of involvement with other people; versus what we do now, where it's massive systems big overall, and complete redesign…"

Transparency among teams and clear communication are vital in addressing inter-team dependency issues, identifying the need for a solution in this project.

4 SOLUTION SELECTION

Agile, originally created for small groups of co-located developers working on a project [20, 23, 26, 27, 28, 29] with the benefits of improved efficiency, was required by larger enterprises [23]. The reality of software development in large organisations is that teams are larger than the Agile design originally intended, with numerous globally distributed teams working on one project [4, 6, 21, 29, 30, 31]; hence the need for scaled Agile. A larger group creates a greater risk of management becoming overwhelmed trying to keep track of the amount of work, losing visibility and control of the project, and a paucity of requirements analysis and architectural planning [22, 23, 26, 32, 33, 34]. Other challenges include: coordination among teams, comprehension of the bigger picture, increased resistance towards change, consistency of coding standards, and requiring documentation — which goes against Agile principles, but is needed with the number of people involved, including external parties [22, 23, 26, 27, 30, 32, 34, 35].

Various frameworks are aimed at scaling Agile. The main ones discussed in the literature are Scrum of Scrums (SoS), Scaled Agile Framework (SAFe), Disciplines Agile Delivery (DAD), Large Scale Scrum (LeSS), and Nexus [4, 6, 20, 21, 22, 24, 26, 27, 28, 29, 31, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44]. According to the 2018 Agile Report on scaled Agile approaches, 29 per cent of organisations use SAFe, 19 per cent use SoS, 10 percent use internally created methods, five percent use DAD, and five percent use LeSS. Enterprise Scrum, Lean Management, and Agile Portfolio Management are at three percent each, Nexus at one percent, and none use Recipes for Agile Governance in the Enterprise [34]. Frameworks have their strengths and weaknesses; and even though SoS is the most widely used scaled Agile solution, there seems to be a key issue with it. With the large number of people attending the Scrum-of-Scrum meetings from the various teams, the meetings became too long, and too few tasks overlapped in each individual group's set of work to be considered useful [22, 26, 27]. Therefore, as a part of the main framework discussion and comparison, SoS was excluded. With SoS excluded, SAFe, DAD, and LeSS are the most used scaled Agile approaches [28, 34], and so they were included in the framework evaluation and analysis exercise. Looking at LeSS and Nexus, Coleman [36] stated that "both options provide a much better alternative to Scrum of Scrums".

As FinTechEnterprise was using Scrum, Nexus was included in the framework evaluation. Even though it is not a widely used scaled Agile approach, the reasoning was that it is specifically a scaled Scrum approach. With Scrum already a popular framework in FinTechEnterprise, and a main issue with scaling Agile being resistance to change, investigating a framework that uses what the environment already knows and uses implies less change, and therefore less resistance to change, increasing the probability of success. There is a lack of assessment criteria when trying to choose which scaled Agile framework is best for an enterprise environment [24], as stated below.

"Many noted the lack of any assessment model for conducting such a comparison to guide critical decisions on adopting specific large-scale agile frameworks."

Each framework's background is given for evaluation. Table 1 provides summarised information extracted from the literature on the pros, cons, complexity, and level of inter-team communication for each framework. Success factors and challenges are also considered, providing for a better assessment.

SAFe, a scaled Agile framework, combines Scrum, Lean, Kanban, and a mix of Agile methodologies [4, 40]. SAFe has various levels that cater for different enterprise needs and complexity levels, specifying levels with clearly defined roles, processes, and organisational structures [20, 40, 45]. LeSS is a scaled Scrum framework with two variations catering for different numbers of Agile teams involved on one project — one for two to eight teams, and the other for more than eight teams [29, 45]. DAD, a mixture of Scrum and Lean methodologies, focuses more on the delivery of IT solutions, taking into account the technical aspects, and with a larger scope and more roles than Scrum [42]. Nexus, similar to Scrum-at-scale, using elements and fundamentals, can be used with three to nine teams and one product backlog [6, 31, 36, 44]. Nexus looks at collaboration and dependencies between teams, and aims to deliver an integrated increment every cycle, focusing on quality and speed of deliverables [31, 36, 38].

Table 1: Scaled agile framework evaluation and analysis

	SAFe	LeSS	DAD	Nexus
Pros	• More defined roles (ideal management) [46]. • Quicker time-to-market, shorter lead times [17, 40]. • If scaled Agile is needed and the enterprise still uses traditional methods, it provides an easier transition into Agile because of its structure [43]. • Flexible in order to support small to extremely large enterprises [39, 45]. • Keeps business goals in focus[39].	• Lightweight, flexible, and easy to adapt to enterprise environment [4, 45]. • Extends Scrum to scale level, but not overly complicated [45]. • Customer-centric, which leads to happier customers and thus better business [29].	• Flexible and adaptable [42]. • Shorter delivery times [42]. • Optimisation for the entire organisation and not just a certain team/group [42]. • Gives good guidance for which processes would suit a project best, architecture, and development operations [41, 43].	• Increased visibility and decreased inter-team dependencies [38]. • Decreased lead time [33, 37, 38]. • Lightweight and easy to set up [33, 36]. •
Cons	• It is one of the most complex of the scaled Agile frameworks. [4] • It is very structured, which leads to an increase in hierarchy and a decrease in flexibility and agility, making it less adaptable. Some have referred to it as the 'new waterfall' [1, 4, 39, 41, 43]. • It is very high level as it keeps the enterprise-vision in mind; but this can cause longer planning and therefore cycles [39]. • There is a lot of guidance as to how to implement and start SAFe, but very little guidance once it is implemented [24, 41].	• Owing to the minimalistic approach, there is less structure within the organisation and more reliance on the mindset and communication between people [29]. This can be seen as a high-risk and negative factor. • It is aimed at more advanced practitioners [36].	• Not very descriptive about how to do certain elements; this can lead to disorganisation [41].	• It is still new, so it is still adapting and changing; this can be seen as a negative because issues have not been picked up yet [41]. •
Complexity	• High [4]	• Medium [4]	• Medium [4]	• No analysis results available.
Inter-team communication	• High [41]	• High [41]	• High [41]	• Medium [41]

The SRL shows that there is a class of problems — i.e., a need for scaled Agile solutions in enterprises working on big projects, as well as the issues that surround scaling Agile. Key issues include managing more people, lack of visibility and control for management, the absence of analysis and architectural planning requirements, coordination between teams, consistent coding standards among teams, documentation needs, resistance to change, and each independent team's understanding of the 'bigger picture'. Similar findings are evident in earlier research [32]. FinTechEnterprise raised some of these issues about the project, especially regarding the coordination between teams, which is the major problem instance. The class of problems is well-summarised [38]:

"While the teams were working on one product and had one product owner, they suffered from being organised as component teams and dealing with cross-team dependencies."

The suggested solution areas included four scaled Agile frameworks (SAFe, LeSS, DAD, and Nexus), which were evaluated for their pros, cons, complexity, and level of inter-team communication (summarised in Table 1), and paired with the challenges and success factors of scaling Agile in order to make an informed framework selection. The challenges and success factors were taken from industry implementation and learnings.

SAFe, the most popular scaled Agile framework, has various levels and caters for numerous needs; but because of its comprehensive nature it has been labelled as complex and as resembling the waterfall approach. LeSS, a lighter Scrum-based framework, is more flexible owing to its minimal approach; however, with very little structure, it relies on abstract concepts such as mindset and communication (a known weak factor in FinTechEnterprise). DAD combines methodologies into a flexible framework that goes beyond the Scrum scope to include delivery, ensuring a full-cycle framework. It lacks information about a few of its features, which could cause confusion. Nexus, an extension of Scrum, is a lightweight framework that is easy to set up, and assists with inter-team dependencies. However, it is still new and still changing, making it slightly risky to use, as not all of its potential issues have been realised and addressed.

Referring to Table 1, SAFe is rated as high complexity, with LeSS and DAD rating as medium. Nexus cannot be accounted for, as there was no information. Inter-team communication is a major aspect to consider; all the frameworks score this as high, except for Nexus, which is rated medium. However, with a major issue in scaled-Scrum solutions being resistance to change, and with FinTechEnterprise already running Scrum in its environment, the choices were narrowed to LeSS and Nexus — scaled-Scrum solutions, not combining additional methodologies. This would result in less change and therefore less resistance, leading to a better success rate.

Considering the objective of the study, LeSS and Nexus are lightweight and flexible, are both an extension of Scrum, and can both be used by the same number of teams. LeSS's advantage is that it has a high level of inter-team communication, whereas Nexus is medium level; and LeSS is more customer-centric, whereas Nexus is more orientated to inter-team collaboration. Comparing their cons, LeSS's major disadvantage is its minimalist approach, placing responsibility on the team members to talk; but communication and collaboration is the issue in question at FinTechEnterprise. Nexus is new, and may not have had all its flaws exposed; but it promises improved visibility within a project and decreases inter-team dependencies — which aligns with the main criterion for selecting a scaled Agile approach for FinTechEnterprise.

5 CONCLUSION AND FUTURE RESEARCH

This study identified a problem instance that featured at FinTechEnterprise, which was also evident as a class of problems in the literature. Our analysed problem instance results indicated that a solution was needed to increase communication and collaboration between teams. Using an SLR and extracting knowledge about Agile-at-scale frameworks from the literature, we short-listed four existing frameworks (SAFe, LeSS, DAD, and Nexus) as possible solutions for the enterprise. We evaluated the solutions, comparing their pros, cons, complexity, and inter-team communication. A factor that contributed significantly to the decision was that of trying to mitigate resistance to change, especially with the size of the build and the number of people involved. Because the enterprise was already running Scrum as their Agile framework, and had good results, our data analysis indicated that Nexus, a scaled version of Scrum, would be the best-fit framework for this fin-tech company.

Future research could include a Nexus implementation in FinTechEnterprise, monitoring and documenting the practical implementation. Taking these learnings and comparing them with their theoretical version, the case study should provide insight into and knowledge about what was changed to suit a practical environment. These insights would be helpful to others with a similar business environment who need to scale-up Agile.

ACKNOWLEDGEMENTS

We are grateful to FinTechEnterprise for allowing us to use their environment, and to those who agreed to participate in the study.

REFERENCES

[1] **Denning, S.** 2015. Agile: It's time to put it to use to manage business complexity. *Strategy & Leadership*, 43(5), pp. 10-17.

[2] **Hanssen, G.K.** 2011. Agile software product line engineering: Enabling factors. *Software: Practice and Experience*, 41(8), pp. 883-897.

[3] **Williams, L. & Cockburn, A.** 2003. Agile software development: It's about feedback and change. *Computer*, 36(6), pp. 39-43.

[4] **Ebert, C. & Paasivaara, M.** 2017. Scaling agile. *IEEE Software*, 34(6), pp. 98-103.

[5] **Reifer, D.J., Maurer, F. & Erdogmus, H.** 2003. Scaling agile methods. *IEEE Software*, 20(4), pp. 12-14.

[6] **Prikladnicki, R., Lassenius, C., Tian, E. & Carver, J.C.** 2016. Trends in agile: Perspectives from the practitioners. *IEEE Software*, 33(6), pp. 20-22.

[7] **Bittner, K., Kong, P. & West, D**. 2018. The Nexus Framework for scaling Scrum: Continuously Delivering an integrated product with multiple Scrum teams. Boston, MASS: Pearson.

[8] **Yin, R.K.** 2018. *Case study research and applications: Design and methods*, 6th ed. Los Angeles, California: SAGE Publications.

[9] **Gill, A.Q. & Henderson-Sellers, B.** 2006. Measuring agility and adoptability of agile methods: A 4 dimensional analytical tool. In *The IADIS International Conference on Applied Computing,* IADIS Press, pp. 503-507.

[10] **Bresky, N.** 2007. Book review, 'Root cause analysis: Simplified tools and techniques'. *Technometrics*, 49(3), p. 364.

[11] **Connelly, L.M.** 2012. Root cause analysis. *Medsurg nursing: Official Journal of the Academy of Medical-Surgical Nurses*, 21(5), pp. 316, 313.

[12] **Spector, J.M., Christensen, D.L., Sioutine, A.V. & McCormack, D.** 2001. Models and simulations for learning in complex domains: Using causal loop diagrams for assessment and evaluation. *Computers in Human Behavior,* 17(5), pp. 517-545.

[13] **Schaffernicht, M.** 2010. Causal loop diagrams between structure and behaviour: A critical analysis of the relationship between polarity, behaviour and events. *Systems Research and Behavioral Science*, 27(6), pp. 653-666.

[14] **Kawulich, B.B.** 2005. Participant observation as a data collection method. *Forum: Qualitative Sozialforschung/Forum: Qualitative Social Research*, 6(2),.

[15] **Byrne, M.** 2001. Interviewing as a data collection method. *AORN Journal*, 74(2), pp. 233-235.

[16] **Harrell, M.C. & Bradley, M.A.** 2009. *Data collection methods: Semi-structured interviews and focus groups*, Santa Monica, CA: Rand National Defense Research Inst.

[17] **Guest, G., MacQueen, K.M. & Namey, E.E.** 2012. *Applied thematic analysis.* Thousand Oaks, California: Sage.

[18] **Okoli, C. & Schabram, K.** 2010. *A guide to conducting a systematic literature review of information systems research*. Available from: http://sprouts.aisnet.org/10-26 [accessed Sept 2018].

[19] **Kitchenham, B., Brere, O.P., Budgen, D., Turner M., Bailey, J. & Linkman S.** 2008. Systematic literature reviews in software engineering — A systematic literature review. *Information and Software Technology*, 51(1), pp. 7-15.

[20] **Paasivaara, M.** 2017. Adopting Aafe to scale Agile in a globally distributed organization. In *2017 IEEE 12th International Conference on Global Software Engineering (ICGSE)*, pp. 36-40.

[21] **Dikert, K., Paasivaara, M. & Lassenius, C.** 2016. Challenges and success factors for large-scale Agile transformations: A systematic literature review. *The Journal of Systems & Software*, 119, pp. 87-108.

[22] **Paasivaara, M. & Lassenius, C.** 2014. Deepening our understanding of communities of practice in large-scale Agile development. In *2014 Agile Conference*, pp. 37-40.

[23] **Paasivaara, M., Lassenius, C., Heikkilä, V.T., Dikert, K. & Engblom C.** 2013. Integrating global sites into the Lean and Agile transformation at Ericsson. In *2013 IEEE 8th International Conference on Global Software Engineering*, pp. 134-143.

[24] **Conboy, K. & Carroll, N.** 2019. Implementing large-scale Agile frameworks: Challenges and recommendations. *IEEE Software*, 36(2), pp. 44-50.

[25] **Kitchenham, B.** 2004. *Procedures for performing systematic reviews. Joint Technical Report*. Keele, UK: Keele University.

[26] **Paasivaara, M., Lassenius, C. & Heikkilä, V.T.** 2012. Inter-team coordination in large-scale globally distributed scrum: Do scrum-of-scrums really work? In *Proceedings of the 2012 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, pp. 235-238.

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

[28] **Kalenda, M., Hyna, P. & Rossi, B.** 2018. Scaling Agile in large organizations: Practices, challenges, and success factors. *Journal of Software: Evolution and Process*, 30(10), pp. 1-25.

[29] **The LeSS Company B.V.** 2019. *Introduction to less.* Available from: https://less.works/less/framework/introduction.html [Accessed 25 May 2019].

[30] **Reifer, D.J., Maurer, F. & Erdogmus, H.** 2003. Scaling Agile methods. *IEEE Software*, 20(4), pp. 12-14.

[31] **Schwaber, K.** 2018. Nexus™ guide, in *The definitive guide to scaling Scrum with Nexus: The rules of the game*. Available from: Scrum.org.

[32] **Fourie, L. & De Vries, M.** 2017. Exploring enhancements to the Agile approach for mid-sized enterprises in the services sector. *South African Journal of Industrial Engineering*, 28(3), pp. 29-39.

[33] Scrum.org. 2018. *Cathay Pacific Airways makes Nexus their official scaling framework for it: Increases product delivery by 200%.* https://scrumorg-website-prod.s3.amazonaws.com/drupal/2018-09/Cathay-Pacific-Airways_Sept2018.pdf

[34] **CollabNet & VersionOne.** 2018. *The 12th state of Agile report*. Available online [accessed 3 November 2020]: https://stateofagile.com/#ufh-i-613553652-12th-annual-state-of-agile-report/7027494

[35] **Cohn, M.** 2010. *Succeeding with Agile: Software development using Scrum.* Boston, MA: Pearson Education.

[36] **Coleman, J.** 2018. *How do Nexus and LESS differ?* https://www.business2community.com/strategy/how-do-nexus-and-less-differ-02137501.

[37] **Scrum.org.** 2018. How net health scales Scrum with the Nexus framework. Available from: https://scrumorg-website-prod.s3.amazonaws.com/drupal/2018-02/Case-Study_NetHealth_February2018_web.pdf [accessed 1 June 2019].

[38] **Scrum.org**. 2017. *Security software product company uses nexus ™ framework*. https://www.scrum.org/resources/security-software-product-company-uses-nexus-framework [accessed 1 June 2019]

[39] **QASymphony.** 2018. *The pros and cons of the scaled Agile framework (Safe)*. Available from: https://www.qasymphony.com/blog/pros-cons-scaled-agile-framework-safe/ [accessed 1 June 2019].

[40] **Scaled Agile.** 2018. *Safe® 4.6 introduction overview of the scaled Agile framework® for lean enterprises.*

[41] **Dolman, R. & Spearman, S.** 2017. *Ask: Agile scaling knowledge — The matrix*. Available from: http://www.agilescaling.org/ask-matrix.html [accessed 18 May 2019].

[42] **Project Management Institute.** *Introduction to Disciplined Agile Delivery (DAD).* Available from: https://www.pmi.org/disciplined-agile/process/introduction-to-dad#:~:text=Disciplined%20Agile%20Delivery%20(DAD)%20is,enterprise%20aware%2C%20and%20is%20scalable. [accessed 1 June 2019].

[43] **Francino, Y.** *Large-scale Agile frameworks compared: Safe vs DAD.* Available from: https://techbeacon.com/app-dev-testing/large-scale-agile-frameworks-compared-safe-vs-dad [accessed 1 June 2019].

[44] **Verma, R.** 2017. *I'm not calling your baby ugly — Two ways and 25 dimensions to compare Agile scaling frameworks.*https://smoothapps.com/index.php/2017/04/im-not-calling-your-baby-ugly-2-ways-and-25-dimensions-to-compare-agile-scaling-frameworks-less-safe-nexus/.

[45] **Kalenda, M.** 2017. Scaling Agile software development in large organizations. Master's thesis in *Faculty of Informatics*, Masaryk University. Available from [accessed 3 Novemebr 2020]: https://is.muni.cz/th/410499/fi_m/masters_thesis.pdf

[46] **Larman, C. & Vodde, B.** 2010. *Practices for scaling Lean & Agile development*: *Large, multisite, and offshore product development with large-scale Scrum.* Boston, MA: Pearson Education.