

# **Extent of Large-Scale Agile Transformation within Momentum Investments Software Delivery Teams**

**Neil Karamagi Coutinho  
1967341**

**A consultancy report submitted to the Faculty of Commerce, Law and  
Management, University of the Witwatersrand, in partial fulfilment of the  
requirements for the degree of Master of Business Administration**

**Johannesburg, 2022**

**Protocol number: WBS/BA1967341/408**

## DECLARATION

I, **Neil Karamagi Coutinho**, declare that this consultancy report is my own work except as indicated in the references and acknowledgements. It is submitted in partial fulfilment of the requirements for the degree of Master of Business Administration in the Graduate School of Business Administration, University of the Witwatersrand, Johannesburg. It has not been submitted before for any degree or examination in this or any other university.

---

Signed at .....JOHANNESBURG.....

On the .....27<sup>th</sup>..... day of .....February..... 2023....

## **ACKNOWLEDGEMENTS**

This particular MBA journey has not been the most traditional route with multiple unexpected personal challenges that tested every facet of my being. First and foremost, to God be the glory for carrying me through this all!

I am eternally grateful for the people that pushed and supported me, notably my family. My parents have been one of my biggest supporters in this journey – providing prayers and encouragement throughout this whole journey. I am truly thankful to them for always sticking by my side. Mwebale muno muno!

Most notably, I would like to thank my devout partner, Mo, who has had to deal with both my absences through this course and picking up many of the family duties I placed on her shoulders. She has been the true rock in our family providing immeasurable support at the most difficult times. I truly love you and can only hope I can make it up to you through the rest of our lives.

To my daughter Namara - born at the start of this journey – you have been my muse more than you will ever know.

To my supervisor Greg Lee, thank you for your patience through this all and helping to get me over this last hill.

Lastly, thank you so much to my employer Momentum Investments to whom this report would have no basis. The time off given to work on this and the access to their data will always be remembered. To my many colleagues and friends in this wonderful company who so graciously afforded me their time – I thank you all.

## **SUPPLEMENTARY INFORMATION**

Supervisor: Dr Gregory Lee

Word count †: 12 147

Supplementary files: Momentum Investments Consent to Access Letters

Momentum Investments/ Wits NDA

Publish Anonymized Project Data Set

† Including Executive Summary, References, etc

## LIST OF TABLES

Table 1: Defect vs Feature Proportion on a Portfolio Level Over Time .....	45
Table 2: Average Defect vs Feature Proportion per Team Over Time .....	46
Table 3: Average Days to Resolve Defects/Features Over Time .....	46
Table 4: Two-way ANOVA Test for Feature Task Priority vs Average Resolution Time (Days) over Time Period .....	47
Table 5: Two-way ANOVA Test for Defect Task Priority vs Average Resolution Time (Days) Over Time Period .....	48
Table 6: Two-way ANOVA Tests P-values for Feature and Defect Priority vs Average Resolution Time (Days) Across Teams Per Year.....	49

## LIST OF FIGURES

Figure 1: Waterfall Methodology Sequence and Deliverables.....	6
Figure 2: Agile Lifecycle.....	8
Figure 3: Adoption of Large-Scale Agile Frameworks .....	11
Figure 4: SAlFe® for Lean Enterprises: Portfolio Configuration .....	12
Figure 5: Hypothesised Quality Drivers in Agile .....	16
Figure 6: Proportion Defects vs Features Over Time .....	23
Figure 7: Proportion Average Feature vs Defect Count Per Team Over Time .....	25
Figure 8: Average Number of Days to Resolve Task .....	26
Figure 9: Defect and Feature Resolution Time Comparison to 2013 Baseline.....	27
Figure 10: Agile Testing Matrix .....	33
Figure 11: Economic View of Prioritisation .....	34

## **EXECUTIVE SUMMARY**

Large-scale Agile transformations expand on Agile principles of flexible, speedy, incremental and iterative development by applying these principles on an enterprise level framework. Momentum Investments has been on a multi-year Agile transformation process within its software delivery teams implemented with Scaled Agile Framework for Enterprise (SAFe®) as their large-scale Agile framework of choice.

For Momentum Investments, it therefore becomes imperative to take stock as to what degree its multiple software delivery teams have benefited from this process. Specifically, from a software quality and software value standpoint to help define the company's future strategies.

A longitudinal quantitative study was performed on multiple years of team project data collected through Momentum Investments internal project management tool. Key metrics investigated were feature and defect density levels, task resolutions times and task priority.

Key findings showed that Momentum Investments has realised benefits in terms of task resolution times as well as a marginal improvement in the overall levels of features and defects resolved. However, concerns regarding improvement and refinement of quality assurance processes, task prioritisation and task metrics completeness were revealed.

The findings recommend that Momentums Investments should take a more holistic refinement and improvement approach to quality assurance and testing practices based on the Agile testing matrix. Additionally, economic views and systems thinking should play a more pivotal role in task prioritisation activities. Furthermore, teams should be more accurate and diligent in better recording task metrics for future analysis.

Momentum Investments has so far benefited from its large-scale Agile transformation process. This can further be improved upon by its Agile sponsors, at all levels in the portfolio, taking a keener approach implementing, monitoring and evaluating its delivery processes in line with the given recommendations.

Keywords:

Agile, SAFe, Scaled Agile, transformation, teams, software quality, software value, large-scale

## Table of Contents

<b>DECLARATION.....</b>	<b>II</b>
<b>ACKNOWLEDGEMENTS .....</b>	<b>III</b>
<b>SUPPLEMENTARY INFORMATION .....</b>	<b>IV</b>
<b>LIST OF TABLES.....</b>	<b>V</b>
<b>LIST OF FIGURES .....</b>	<b>VI</b>
<b>EXECUTIVE SUMMARY.....</b>	<b>VII</b>
<b>1: INTRODUCTION.....</b>	<b>1</b>
1.1 Purpose .....	1
1.2 Background .....	1
1.3 Research Problem Statement .....	4
1.4 Significance of this Study .....	4
1.5 Scope of this Study .....	4
<b>2: LITERATURE REVIEW.....</b>	<b>5</b>
2.1 Purpose .....	5
2.2 Agile Software Development .....	5
2.3 Large-Scale Agile Transformation .....	9
2.3.1 Scaled Agile Framework .....	11
2.4 Software Quality .....	13
2.5 Software Value .....	16
2.6 Summary .....	18
<b>3: RESEARCH METHODOLOGY.....</b>	<b>19</b>
3.1 Introduction .....	19
3.2 Research Design.....	19
3.3 Population and Sample.....	20
3.3.1 Population .....	20

3.3.2 Sample and Sampling Method.....	21
<b>3.4 Measures.....</b>	<b>21</b>
<b>3.5 Data Collection.....</b>	<b>21</b>
<b>3.6 Data Analysis .....</b>	<b>22</b>
<b>4: FINDINGS AND IMPACT.....</b>	<b>23</b>
4.1 Feature vs Defect Tasks as a Proportion of Deliverables.....	23
4.2 Average Time to Resolve Features and Defects.....	25
4.3 Prioritisation of Tasks.....	28
4.4 Summary of Findings and Impact.....	29
<b>5: RECOMMENDATIONS.....</b>	<b>31</b>
5.1 Introduction .....	31
5.2 Greater Focus on Quality Assurance and Testing.....	31
5.3 Improved Prioritisation .....	34
5.4 Better Data .....	35
<b>6: CONCLUSION .....</b>	<b>36</b>
<b>REFERENCES .....</b>	<b>38</b>
<b>ANNEXURES .....</b>	<b>44</b>
<b>Validity and Reliability.....</b>	<b>44</b>
External Validity .....	44
Internal Validity .....	44
Reliability.....	45
<b>Findings.....</b>	<b>45</b>

# **1: INTRODUCTION**

## **1.1 Purpose**

The purpose of this report is to evaluate whether the various software delivery teams within Momentum Investments have realised the benefits of long-term, large scale Agile transformation. This will be primarily done from a software quality and software value perspective. The objective of this report will be to provide recommendations to Momentum Investments in line with these findings.

## **1.2 Background**

Success, for many businesses, is based on whether a project was completed in line with pre-defined and agreed deliverables, and if it was delivered on time and under budget to the expected quality. However, the information technology sector is notorious for numerous failed and delayed projects (Kona, 2015).

There are varied opinions regarding the success of information technology projects and whether investments in IT yield positive returns. This is largely due to many organisations' failure to accurately pinpoint, track and evaluate benefits from investments in IT projects (Zwikael & Smyrk, 2012).

Surveys performed over the last couple of decades have shown that between 60% and 80% of all organisations failed in accomplishing their IT strategies due to not delivering on the expected outcomes (Kaplan & Norton, 2008). In numerous cases these projects fail as a result of being delivered late, project deliverables being out of scope or of subpar quality and/or poor software performance (Kona, 2015). Given that South African financial institutions spend billions of Rands annually on IT investments, growing at an average of 12% annually (Tarrant, 2016), it becomes imperative that IT strategies are managed accordingly, more so now than ever.

Organisations must strive to ensure the success of their IT investments in order to effectively execute on their chosen strategy and realise their visions. To achieve this, project management teams must clearly define the merits of what determines a project to be successful (Serra & Kunc, 2015).

Currently the software industry trends towards utilizing traditional project management methodologies due to their process driven way-of-work and these traditional methodologies capability to define a predictable and steady delivery pace. However, these very characteristics tend to result in delayed responsiveness to ever-changing user demands and needs. Furthermore, software deliverables tend to lack expected functionality and are delivered later than the business would ideally like when implemented on the back of these traditional methodologies (Kona, 2015).

The introduction of Agile software development practices was an attempt to seek less cumbersome, quicker and more responsive software development (Abrahamsson et al., 2017). Agile is defined as a flexible, iterative development model and encourages continuous liaison with the client as well as short delivery cycles (Mbuya, 2016). Agile has become a trend word in many an organisation seeking to gain the benefits of greater flexibility and speed to market espoused by many practitioners. Various companies have found that the adoption of Agile practices has resulted in better quality software from a usability standpoint, faster delivery time frames and improved responsiveness to changing user requirements (Kona, 2015).

However, Agile transformation requires time, patience and dedication to yield positive results. Furthermore, Agile requires a significant mindset shift within the organisation over a period of time in order to break and overcome entrenched habits (Goldstein, 2013). Unfortunately, larger enterprises tend to be far less accepting to transformation. Bhasin (2012) also observed that larger enterprises experience far greater challenges adopting Agile in balancing transparency, flexibility, and collaboration. It is also recommended that larger enterprises should be particularly conservative when defining expectations for Agile

transformation, notably when adopting Agile methodologies on large projects (Reifer et al., 2003).

Though numerous benefits of Agile methodologies have been documented, limited information exists concerning the negative aspects and challenges organisations face adopting Agile methodologies – more especially the process of Agile adoption (Kona, 2015). Furthermore, minimal information is available regarding long-term benefits for organisations that adopt and practice Agile software development methodologies (Abrahamsson et al., 2017). Mbuya (2016) observed that only 13% of companies surveyed had Agile experience that exceeded six years. As more companies mature on their Agile journey, they will undoubtedly want to gauge how far along a transformation journey they have come.

Momentum Investments is one such organisation that has been on an over 5-year Agile transformation journey. This makes it one of the more mature Agile organisations in South Africa according to the State of Agile in South Africa 2018 report (IQbusiness, 2018). Currently Momentum Investments comprises multiple individual software delivery teams all practicing Agile and the entire portfolio running Scaled Agile Framework for Enterprise (SAFe®) as their large-scale Agile framework of choice.

Given the number of teams, all at different stages of their Agile journey, it becomes critical for Momentum Investments to understand where it is on its Agile transformation journey to be able to carve the path of its next steps.

### **1.3 Research Problem Statement**

This study aims to determine to what extent Momentum Investments has achieved large-scale agile transformation within its software delivery teams, over a multi-year implementation period, from a software quality and software value viewpoint.

### **1.4 Significance of this Study**

This study aims to help evaluate the degree of success of Agile transformation, particularly from a software quality and software value viewpoint, within Momentum Investments. This is so as to enable stakeholders to better quantify this transformation process with the hope that this springboards current and future business initiatives to both improve, correct and advance their Agile journey.

This is important as many companies tend to not perform thorough cost-benefit analysis or post-implementation analysis to assess whether IT projects effectively executed on delivery (Serra & Kunc, 2015). Additionally, a lack of industry recommended methods to evaluate, monitor and determine the benefits from Information Technology investments and projects is a challenge for numerous organisations (Terlizzi & Albertin, 2017).

Given that approximately only 16% of companies in South Africa practicing Agile have more than 5 years of experience (IQbusiness, 2018), it is also my hope that this study may potentially assist other organisations in South Africa planning on embarking on their own Agile journey.

### **1.5 Scope of this Study**

This study was limited to Momentum Investments software delivery teams. This study examined 6 years' worth of team delivery/project data from 2013 – 2019.

## **2: LITERATURE REVIEW**

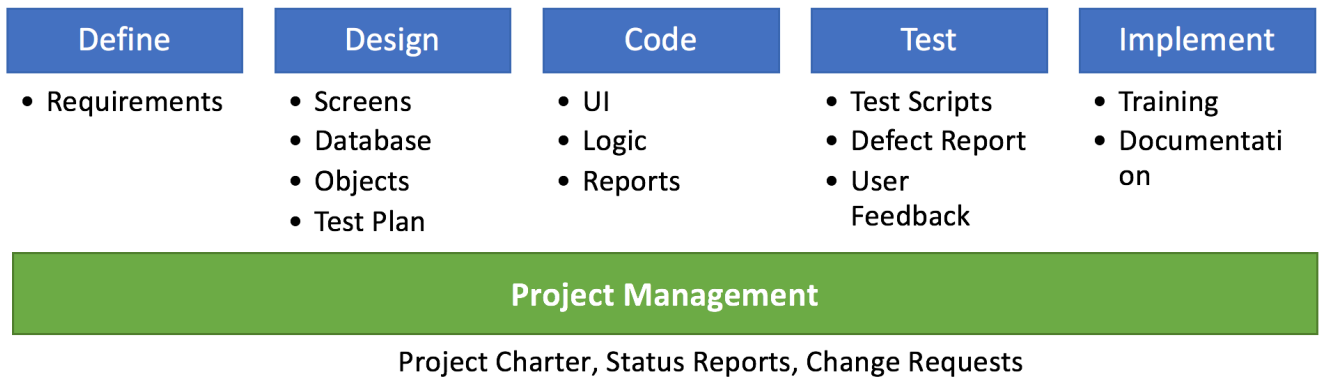
### **2.1 Purpose**

The purpose of this literature review was to find existing works/studies that would assist in addressing the research problem(s) by delving into the following topics: (1) Agile Software Development (2) Large Scale Agile Transformation, with special emphasis on the Scaled Agile Framework (3) Software Quality and (4) Software Value.

### **2.2 Agile Software Development**

Software development projects are notorious for cost and schedule overruns, as well as quality and usability problems (Kwak & Stoddard, 2004). Traditional software development methodologies were seen as too cumbersome, bureaucratic, and inflexible (Lindvall et al., 2004). Schwaber and Beedle (2002) also determined that traditional software development approaches are significantly time consuming before deliverables are produced. This delay can then mean that both client expectations, priorities and market conditions might change.

“Waterfall” methodology is a term coined by Winston Royce in the 1970s to define a structured progression between phases. In software development, as shown below in Figure 1, it also defines a traditional engineering lifecycle with a sequential software engineering methodology to produce deliverables (Awad, 2005) .



**Figure 1: Waterfall Methodology Sequence and Deliverables**

*Source: Awad (2005)*

Frustrated with the then status quo, a group of 17 experienced software engineers came together in February 2001 and conceived the Agile manifesto (Beck et al., 2001a) :

- **Individuals and interactions** over processes and tools
- **Working software** over comprehensive documentation
- **Customer collaboration** over contract negotiation
- **Responding to change** over following a plan

The manifesto stated that though the practices on the right are valued, they are not as highly valued as the practices on the left **in bold** (Beck et al., 2001a).

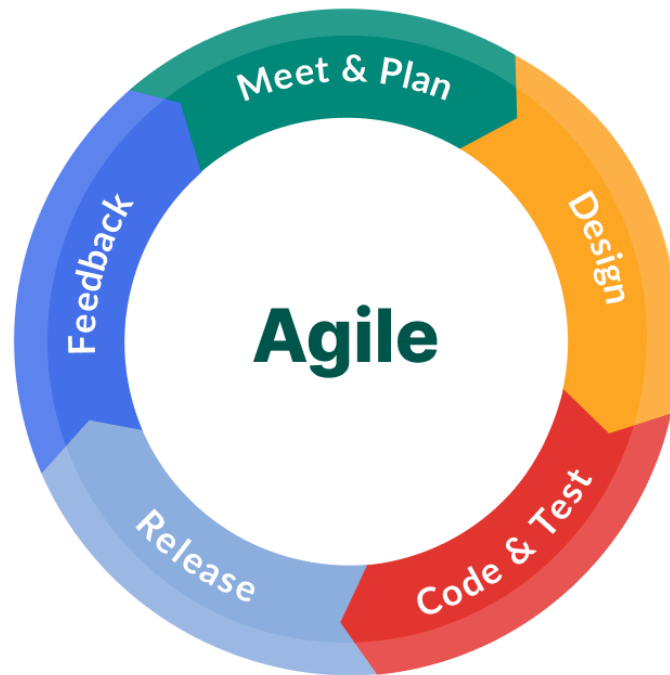
These values of the Agile manifesto informed the following twelve principles for Agile software delivery (Beck et al., 2001b):

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business representatives and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity. The art of maximising the amount of work not done is essential.
11. The best architectures, requirements, and designs emerge from self-organising teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.

Abrahamsson et al. (2017) define the core tenants of Agile methods as an “unforgiving honesty of working code, effectiveness of people working together with goodwill, and a focus on teamwork”. Agile methodologies follow an iterative development model. Agile is characterised as being team-oriented, flexible and able to more easily accommodate changes throughout the software development life cycle. Additionally, it strongly encourages continuous liaison with the client as well as short delivery cycles (Kona, 2015; Mbuya, 2016).

As shown below in Figure 2, in the Agile software development life cycle development, design, coding and testing are done simultaneously during short, time boxed periods, typically less than 4 weeks, known as sprints (Korhonen, 2012). Additionally, the testing and quality assurance activities occur earlier on and more frequently in the Agile development software cycle than traditional process-driven “Waterfall” methodologies (Zhou & Xu, 2008). This allows for a faster and more flexible response to change, thereby supporting earlier and quicker development of working code (Turk et al., 2002). Shorter turnaround times and flexibility also means that features delivered correlate more to customer needs, thereby increasing customer satisfaction (Korhonen, 2012).



**Figure 2: Agile Lifecycle**

*Source: Dreissigacker (2022)*

These benefits resulted in a number of smaller organisations pursuing Agile, notably as a mechanism to try lower their development costs (Lindvall et al., 2004). Some early Agile researchers deemed that Agile was inherently suitable for small and medium sized teams and that it had limited support for larger teams (Chow & Cao, 2008; Turk et al., 2002). Turk et al. (2002) further elaborated that Agile at its heart is “management in the small”, ideally suited for small and medium sized teams given that coordination, control, and communication mechanisms are more effective in such environments. However, the very same attributes that made Agile attractive to smaller teams also appealed to larger organisations (Lindvall et al., 2004) and the number of larger organisations adopting Agile practices has increased significantly over the years (Mbuya, 2016).

However, despite a growth in the number of larger organisations shifting to Agile, several studies have shown that many do experience challenges and impediments adopting Agile, notably with putting benefits into practice to gain competitive advantage (Barlow et al., 2011; Mbuya, 2016). Awad (2005) and

Fowler (2005) similarly observed that this difficulty introducing Agile processes typically stems from the organisation having entrenched traditional organisation structures counter to recommended Agile practices. These structures are largely a result of big upfront design thinking, therefore, to adopt Agile successfully would require both significant changes in the functions of the organisation as well as team and personnel changes. One specific organisational area that Fowler (2005) highlights is Quality Assurance and Testing – particularly more prone to resistance due to a greater focus and attention required from these departments and roles through every iteration in the Agile process.

It is often wrongly assumed that Agile methodology is only applicable to development teams, when in reality it involves all levels of the organisation (Bhasin, 2012). If a large organisation's objectives are to produce software functionality quicker and faster, the adopted Agile methods have to scale to meet their requirements (Reifer et al., 2003). Large-scale Agile transformation and its various frameworks are ways in which organisations have gone about addressing these implementation concerns.

### **2.3 Large-Scale Agile Transformation**

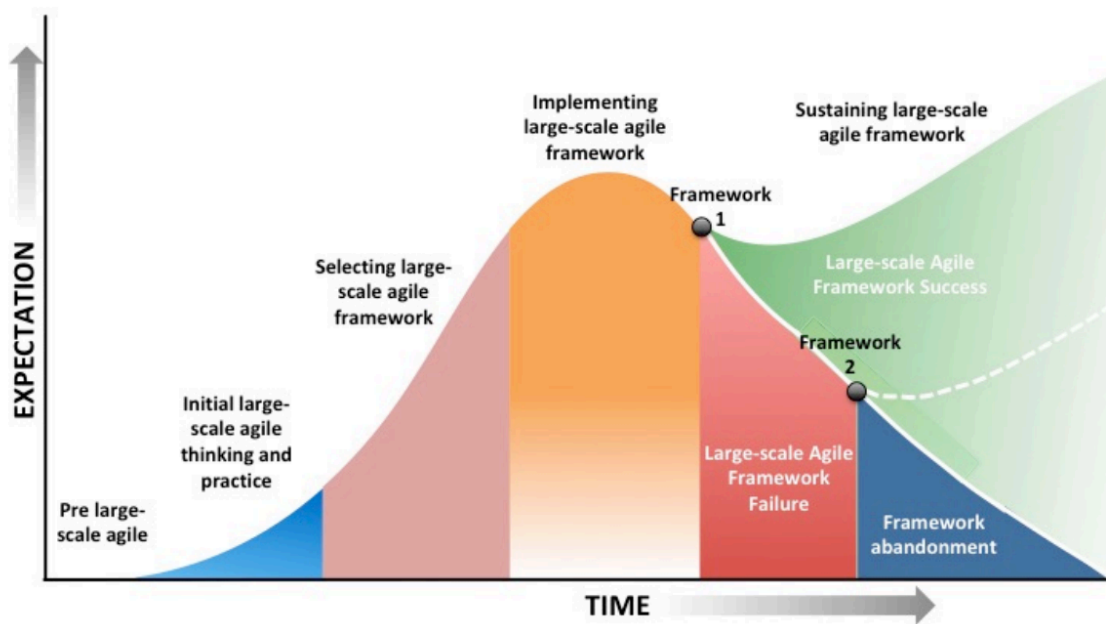
Large-scale Agile development can be characterised as over 50 persons, across more than 5 teams, working to deliver the same product (Dingsøyr & Moe, 2014). Laanti (2012) proposes that for large organisations, Agile adoption should not be done only at a certain level of the organisation, such as at the lowest levels where development teams are typically found, but rather all levels of the organisation must change to a new way of working. The author does concede though that smaller successes can lead into organisational learning to spur wider Agile adoption. Lal and Clear (2017) advise that the predominant goal of an Agile scaled method implementation should be an organisational shift to better align technical activities with the overall enterprise objectives.

A few case studies have been published showing that large organisations have reaped the benefits of embracing Agile methodologies in both minimizing time

to market and the flexibility to make requirement changes (Papadopoulos, 2015). The author of one such case study does warn that organisational adoption of an Agile framework is complex, notably those already with traditional processes and ways of working (Papadopoulos, 2015). Papadopoulos (2015) goes on to state that activities need to be closely monitored to identify and mitigate problems and thereafter establish customised Agile guiding principles that match the project's needs.

Ambler (2006) also strongly supports that Agile can work for distributed teams in larger organisations. However, the author does note that collaboration and communication are critical to its success. Furthermore, to scale effectively, some upfront modelling should be practiced, but in an Agile manner, and larger teams should be broken down into smaller teams as they are not nimble enough to respond to change. Additionally, these smaller teams should be coordinated effectively to gain the most of scaled Agile.

However, Gandomani et al. (2013) propose that an Agile transformation should rather be executed on a comprehensive framework, one that takes into consideration factors such as people, organisation, management, process and technology. The authors go on to state that without a disciplined strategy an organisation's Agile development will be limited. In Figure 3, Conboy and Carroll (2019) similarly espouse that such frameworks incorporate various tool sets that assist to establish predefined workflow patterns and routines. However, the authors do note that empirical data showcasing the use, effectiveness and challenges as result of adopting such frameworks is limited.



**Figure 3: Adoption of Large-Scale Agile Frameworks**

*Source: Conboy and Carroll (2019)*

Ebert and Paasivaara (2017) determined that the largest and greatest challenge adopting scaled Agile is both organisational and individual mindset. The authors recommend that rather than tailoring a scaled Agile framework to suit the organisation, focus should rather be on ensuring that Agile culture is entrenched throughout both management and engineering.

### **2.3.1 Scaled Agile Framework**

Scaled Agile Framework for Enterprise (SAFe®), is the framework of choice for Momentum Investments and 65% of other large organisations in South Africa implementing scaled Agile (IQbusiness, 2018). It is a comprehensive framework for implementing Agile at an enterprise level (Leffingwell, 2019). It describes a framework for the structure of an organisation to best enable effective Agile software delivery.

As shown in Figure 4 below, SAFe® breaks down product portfolios, programs and teams into separate levels in order to best cater to differing concerns.

SAFe® then assesses an enterprise on a scorecard based on four measures: quality, value delivery, efficiency and agility (Leffingwell, 2019).



The primary goal of SAFe® is to provide a detailed framework intended to deliver and develop enterprise level solutions within a defined time frame. It does this by firmly incorporating Agile approaches in a manner specifically designed to overcome challenges experienced by large scale organisations (Leffingwell, 2019; Poth et al., 2020). In SAFe®, these significant organisational investments and development initiatives that are typically cross cutting across multiple teams are defined as Epics (Scaled Agile, 2022).

The originators of SAFe® highlight numerous benefits of this framework such as an up to 50% increase in productivity, a more than 50% improvement in time to market and quality, as well as a quantifiable increase in employee engagement and job satisfaction (Poth et al., 2020)

According to Pitkänen (2015), though the creators of the framework claim to have numerous successful implementations, there is a lack of academic studies supporting this claim. Paasivaara (2017) fielded a study with Comptel, a globally distributed software development company, that did yield good results by adopting the SAFe® framework in two business lines. However, the author does note that research on how these frameworks are adopted in practice is seriously lacking (Paasivaara, 2017).

Paasivaara et al. (2018) also note that though scaling frameworks, such as SAFe®, provide pertinent insights on how large-scale Agile approaches manifest themselves, they fail to explain the process of a large-scale Agile transformation.

## **2.4 Software Quality**

Agile practitioners tout improved software quality as one of the major strengths of Agile methodologies (de Azevedo Santos et al., 2011). The Shine Technologies “Agile Methodologies Survey” surveyed 131 organisations and found 88% of these organisations stated that quality was either better or significantly better as a result of adopting Agile practices (Shine technologies, 2003). When Agile methods have a positive impact on software quality it in turn

has a positive impact on productivity and cost that results in improved business value (Bhasin, 2012).

Sfetsos and Stamelos (2010) highlight that Agile development redefines quality assurance work by ensuring that quality is embedded throughout the entire development process. Additionally, the frequency of these quality assurance exercises and activities occur both earlier on and more frequently in Agile software development (Zhou & Xu, 2008). Furthermore, the entire team is responsible for ensuring that testing forms an integral part of every stage in the software development lifecycle (Goldstein, 2013). Van Cauwenberghe (2003) also notes that these integral testing practices ensure that defects are not only determined earlier but also resolved quicker – meaning they are inherently cheaper to fix in the long run.

The values of the Agile manifesto support this notion by defining working software as the primary measure of progress (Beck et al., 2001b). Kruchten (2001) observed that prioritising “actual working code” throughout the software development lifecycle enables the product to react and adapt more readily to changing consumer needs and demands. Quality in Agile projects is therefore a key contributor to project success (Abbas et al., 2010).

Barlow et al. (2011) found that studies investigating Agile methodologies in relation to quality on a larger organisational scale are sparse with none on a global scale. Abrahamsson et al. (2010) also argue that empirical evidence remains quite limited regarding long term quality improvements at scale. Similarly, Mbuya (2016) observed that correlations between improved software quality and Agile methodology could not be empirically validated, more specifically in a South African context. The author also notes that these limited studies in other global regions have either been inconclusive or contradictory – where both positive and negative improvements in product quality have been registered.

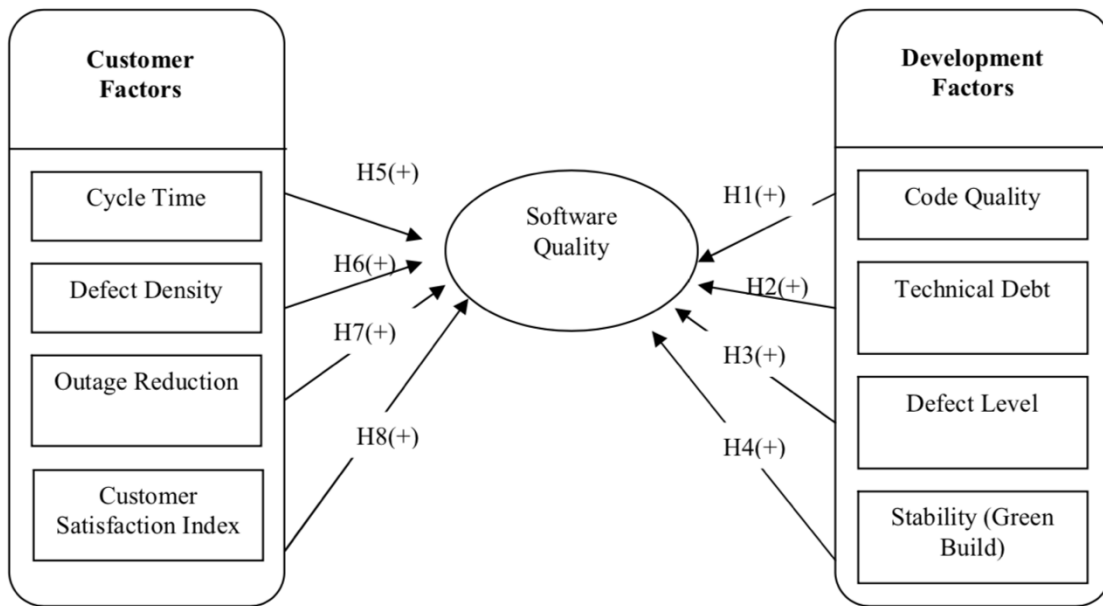
However, Bhasin (2012) observed that larger enterprises do experience challenges implementing Agile and ensuring delivery quality, maintaining

required testing and quality assurance practices. Quality is a difficult concept to specifically define as various authors define it differently – therefore there are various subjective measures to gauge software quality (Mbuya, 2016).

Korhonen (2012) observed that there are changes in recorded defect data with the adoption of Agile practices. Furthermore, by exploring defect data, feedback on the progress of Agile transformation can be obtained and then used to analyse and gauge both the engagement and success of personnel in the organisational Agile transformation process. Papadopoulos (2015), in a separate study, also observed lower defect levels with a team that adopted Agile compared to another team that did not.

Similarly Li et al. (2010) saw no significant reduction in defect densities tracking a team's defect data over several months as the team adopted Agile practices. Likewise, Bhasin (2012) could not derive and prove a significant quantifiable benefit regarding software quality through Agile development by organisations. However, Li et al. (2010) observed that software quality and release date control did improve as a result of better testing efficiency.

As show in Figure 5 below, Bhasin (2012) hypothesised that the Agile influence on quality can be studied and measured by observing various impacted parameters – notably Defect Density, Defect Reduction, Defect Level, Cycle Time Improvement of Delivery and various other development and customer factors.



**Figure 5: Hypothesised Quality Drivers in Agile**

*Source: Bhasin (2012)*

As a result of improved software quality, code is of a higher quality level meaning that faults are determined earlier resulting in not only a faster delivery rate, but a measurable decrease in observed defects as well as time to resolve defect incidences. This translates to fewer customer reported issues and improved customer satisfaction (Bhasin, 2012). Furthermore, changes are cheaper to implement in the long run as a result of well-designed and improved software quality as a result of well implemented Agile software development practices (Cockburn, 2000).

## 2.5 Software Value

A review of the research literature on Agile methodologies and software engineering reveals that the bulk of these studies focus on the development process, rather than the end product itself (Mbuya, 2016). According to Kwak and Stoddard (2004), the most critical factor in commercial software product development is time to market. The current business environment requires organisations to develop and evolve their software systems at a much faster speed (Ramesh et al., 2006). The State of Agile report for South Africa 2018

also confirms this notion with businesses stating accelerated product delivery as the primary reason for adopting Agile (IQbusiness, 2018).

Reifer et al. (2003), in their findings, also found that focusing on deploying releases, with whatever functionality exists and is ready at the time, according to pre-established, brief and time-boxed schedules is most advantageous to the organisation. The Standish Group International (2015) found in their study of several thousand projects that the delivery of software components early and often, within short time frames, increased the success rate of those projects. The Standish Group also found that approximately 45% of features in an application are never used. This also motivates why frequent small releases are recommended to validate the demand for these features with clients and prevent unnecessary software complexity for unneeded functionality (Awad, 2005).

A regular frequent release cadence enables the customer to provide quicker feedback and enforces the team to make production releases and maintenance cycles as cheap and efficient as possible. Coupled with face-to-face customer testing this ensures that the specifications and functionality of the system are current and in line with business requirements. This organisational discipline helps to significantly mitigate the high cost of fixing errors (Van Cauwenberghe, 2003).

Dingsøyr and Lassenius (2016) highlight an increased focus on value in improvement trends relevant to software development. The authors also noted an increased interest towards utilizing empirical means to understand value. Turk et al. (2002) put forward that software delivery teams should come to understand what stakeholders perceive as value. Yatzeck (2012) offers a very similar argument in that “Lead/Lag” indicators should be considered in an Agile transformation where “lag” measures are values that motivate stakeholders, such as increased speed to market or higher code quality.

Awad (2005) highlights the case study of a Singapore lending project which was deemed unfixable - however by adopting Agile methodologies and focusing on

simple code, better testing, and most importantly frequent small releases, the project not only delivered to client requirements but also earlier and under budget. Ultimately, delivered working code was the real measure of success. Kona (2015) in his research also found that respondents gauged the delivery of high priority features, automated testing, and regular delivery of software as the most important factors for a successful project from a technical standpoint.

## **2.6 Summary**

This literature review covered how Agile came to be as response to the challenges of traditional software development methodologies. It detailed the values and principles that initially informed Agile delivery and formed the core of Agile frameworks to follow. The benefits of an Agile way of working not only initially attracted smaller teams but larger enterprises as well.

The review went on to explore how larger teams initially did start to reap the benefits of Agile delivery, however additional overhead drove the need for more comprehensive scaled frameworks that took the entire organisation into account. The Scaled Agile Framework for Enterprise (SAFe®) is one such scaling framework, and though it provides recommendations on how teams can structure and run themselves, it doesn't necessarily define the process in which that should occur. Therefore, though it does suggest metrics that can be used as a scorecard to gauge Agile transformation, it is ultimately the responsibility of the organisation to determine how to best to execute on their own Agile transformation.

Issues of software quality and software value were explored and how these could be used to track how well an organisation has executed on their Agile transformation – notably in evaluating certain metrics such as defect management and resolution as well as software release cadence.

## **3: RESEARCH METHODOLOGY**

### **3.1 Introduction**

The focus of this research is to determine the extent of large-scale Agile transformation within the software delivery teams at Momentum Investments over a multi-year period (2013 – 2019). A quantitative approach is therefore a suitable methodology to track this change by analysing project data over this period.

Quantitative research is used to answer questions about relationships among measured variables with the purpose of explaining, predicting, and controlling phenomena (Leedy & Ormrod, 2005). A quantitative project is one that is best addressed by understanding what factors or variables influence an outcome (Cresswell, 2005).

### **3.2 Research Design**

The time-based nature of this research makes a longitudinal quantitative study an appropriate choice (Davis-Kean et al., 2015; McCaston, 2005). These types of studies are driven by the desire for precise comparisons of treatments or by intrinsic interest in age or time-related changes (Cook & Ware, 1983).

According to Guo and Klein (2016), longitudinal research has the potential to depict organisational and societal shifts due to information technology implementation and adoption. Venkatesh and Vitalari (1991) recommend that longitudinal research offers the researcher a means to study change, adaptation, and evolution in the information systems research area. Pinsonneault and Kraemer (1993) also propose that questions about impact, policy effects, development and implementation all involve a time dimension and therefore require measurement over time.

Though longitudinal studies do have their benefits, if the time lag is long, many external variables might affect the dependant variables, therefore a causal relationship might be difficult to establish (Pinsonneault & Kraemer, 1993).

Dikert et al. (2016) note that challenges in large-scale Agile development have not yet been studied sufficiently utilising secondary studies. It is known that given the time, expense and effort to collect primary data, secondary data sets allow researchers a convenient alternative to test longitudinal questions than it would take if collecting primary data themselves (Davis-Kean et al., 2015).

Secondary data is data that is examined and used for research other than which the data was initially collected for (Vartanian, 2010). It can be advantageous in saving time and money collecting data. However, since the data collection is complete, the method of collection and quality of data can be difficult to determine (McCaston, 2005).

Secondary data, collected by other researchers or organisations, allow for more rigorous, diverse, and longitudinal analysis (Davis-Kean et al., 2015).

Secondary data analysis also lends itself well to trend analysis as it offers a relatively easy way to monitor change over time (McCaston, 2005).

The design of this research constitutes descriptive statistics of secondary data, obtained from an internal project management tool, as part of a longitudinal study encompassing several years' worth of project data for all the software delivery teams within Momentum Investments.

### **3.3 Population and Sample**

#### ***3.3.1 Population***

The population for this study is all the software development teams within Momentum Investments practicing Agile software delivery over this time period. This particular population comprises 46 individual software delivery teams over this period. However, it must be noted that multiple new teams formed, and older ones dissipated over the multi-year period given the differing durations of various company projects.

### **3.3.2 Sample and Sampling Method**

No sampling method is required as the scope of this report covers all software delivery teams practicing Agile within Momentum Investments. Since the number of teams is small enough and the research design is reliant on secondary data recorded from all the software delivery teams, data for each and every team was analysed.

### **3.4 Measures**

For the purposes of this study, secondary data formed the research instrument. As stated, secondary data is a useful tool for longitudinal analysis (Davis-Kean et al., 2015; McCaston, 2005).

This secondary data constitutes 6 years' worth of project data for all the software delivery teams within Momentum Investments from 2013 to 2019. This project data currently tracks various task metrics related to teams and comprises approximately 100 000 task records.

### **3.5 Data Collection**

Consent was granted and obtained from the head of the department to utilise data from the internal project management tool at Momentum used by the software delivery teams. This was done under the stipulations that the identity of all teams and team members were kept confidential to protect identities and the assurance that no information disadvantages teams/team members in this study. Furthermore, all identifying information regarding projects and features developed in these teams was anonymized to ensure that no private intellectual information of Momentum Investments was inadvertently exposed.

This data was extracted, processed, coded, cleaned, and all sensitive information removed. Since this study does deal with secondary data, concerns of data reliability were warranted. To ensure reliability all data artefacts used and produced regarding process notes, raw data, coded data as well as any categories for data reconstruction and synthesis were recorded for auditability purposes.

One limitation currently observed was that the project data collected only started as the Agile transformation commenced in Momentum Investments. This unfortunately does not give us a baseline value to compare metric trends from before the Agile transformation proceedings took place. To address this limitation, the earliest recorded data in 2013 was used as the baseline value.

### **3.6 Data Analysis**

This study was limited to metrics in relation to software quality and software value. Descriptive statistics of trends on a portfolio and team level were performed on the data to track metric changes over time.

Current data-points analysed on a portfolio and team level were:

- Feature versus defect count over the multi-year period
- Time to develop a feature over the multi-year period
- Time to resolve a defect over the multi-year period
- Time to develop features versus work priority (High/Medium/Low) over the multi-year period and on a yearly basis
- Time to resolve defects versus work priority (High/Medium/Low) over the multi-year period and on a yearly basis

These descriptive statistics were analysed in line with the theory presented in the literature review looking at averages and trends over time on both a portfolio and team level. Additionally, ANOVA tests were conducted to determine statistically significant differences for time to resolve tasks versus work priority on a multi-year period as well as on a year-by-year basis amongst teams.

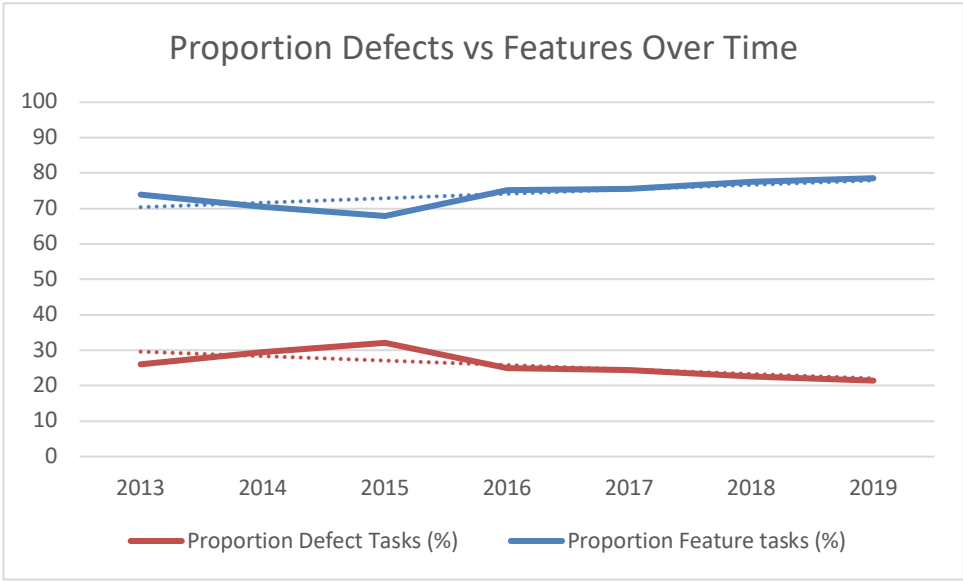
Concerns regarding Validity and Reliability in this research are addressed in the annexure.

# 4: FINDINGS AND IMPACT

## 4.1 Feature vs Defect Tasks as a Proportion of Deliverables

The literature proposes that by exploring observable changes in recorded defect and feature data with the adoption of Agile practices, feedback on the progress of Agile transformation can be obtained to gauge both organisational engagement and success. Ideally, defect density and levels should also see an observable improvement with Agile adoption (Bhasin, 2012; Papadopoulos, 2015).

If we graph the proportion of defects vs features tasks delivered over time for the entire Momentum Investments portfolio from Table 1 in Figure 6 below, we observe an overall positive trendline for the proportion of tasks that are features vs a negative trendline for proportion of tasks that are defects.



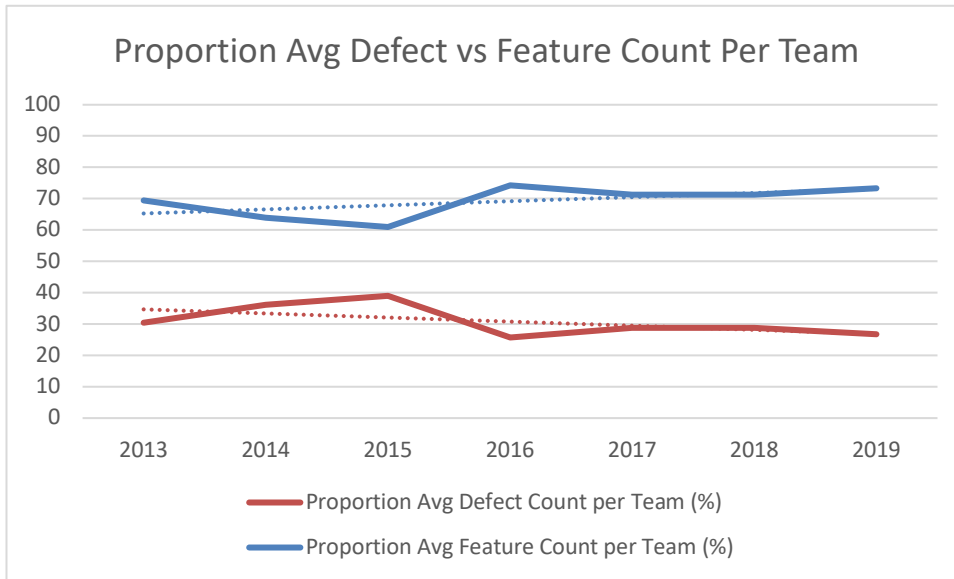
**Figure 6: Proportion Defects vs Features Over Time**

These overall trendlines align with the proposed literature indicating that Agile adoption has resulted in an observable improvement in more features and less defects comprising deliverables produced by various teams. However, we do note another interesting observation; between 2013 and 2015 we see that the proportion of defects recorded increased over time whilst the proportion of

features reduced, before we see a steady improvement from 2016. This overall change over time aligns with the Bridges Transition Model which identifies 3 phases in a transition process (Tremolada, 2015). The period of 2013-2015 highlights the Ending where the transitions starts, usually characterised by scepticism, uncertainty and frustration exemplified by poorer performance from the 2013 baseline (Tremolada, 2015). The year 2015 marks the Transition/Neutral Zone where resistance, confusion and anxiety are at their peak, but is the crux point for change (Tremolada, 2015). From 2016 we see the New Beginning phase at play where the benefits of the Agile transformation start to bear fruits in fewer defects and more features delivered overall - suggesting that the organisation is engaging well with the change (Tremolada, 2015). This suggests an endogenous change, whereby the change occurred from inside the system due to outside actions (i.e. Agile transformation) affecting change on variables inside the system (Doherty, 2022).

However, though this aligns with the theory, the proportion of feature tasks shows an overall 5% improvement whilst the defect tasks saw a 5% decline overall at the end of the 5-year period. The originators of SAFe® highlight an up to 50% increase in productivity, so while it is an improvement, it does fall on the lower end of this productivity improvement scale. Momentum Investments would need to gauge if this is deemed acceptable and potentially use this as a benchmark in defining the ideal proportion of features and defects as overall portfolio deliverables from an Agile transformation perspective.

If we graph the proportion of average defect vs feature count per team over time from Table 2 below in Figure 7 we similarly observe an overall positive trendline for the proportion of average feature counts per team vs a negative trendline for the proportion of average defect counts per team. However, the data trend over time also supports the Bridges Transition Model, as explained above, where teams went through an Ending phase (2013-2015), Neutral/Transition Zone (2015) and New Beginnings phase (2016 -2019).

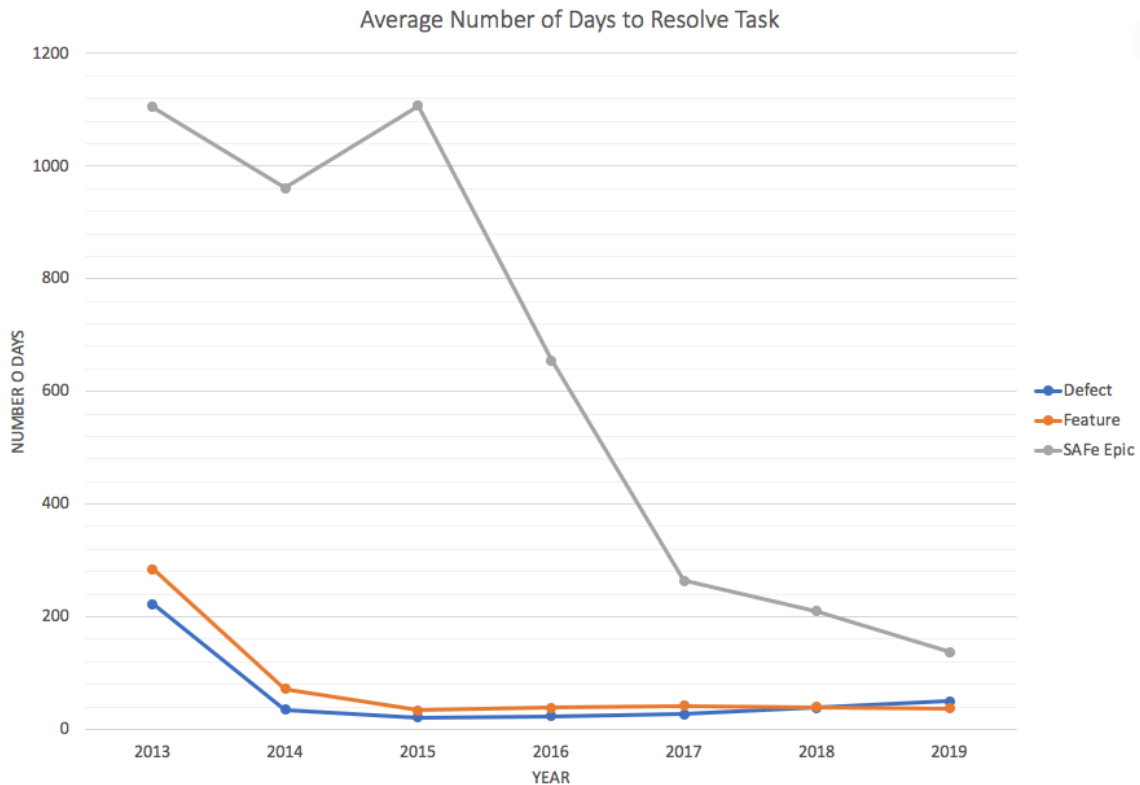


**Figure 7: Proportion Average Feature vs Defect Count Per Team Over Time**

Analysis on a team-by-team level over time for comparison purposes to determine better and poorer performing teams overall by feature vs defect task proportion was difficult to perform. This was because multiple new teams formed, and older ones dissipated over the multi-year period given the differing durations of different company projects.

#### 4.2 Average Time to Resolve Features and Defects

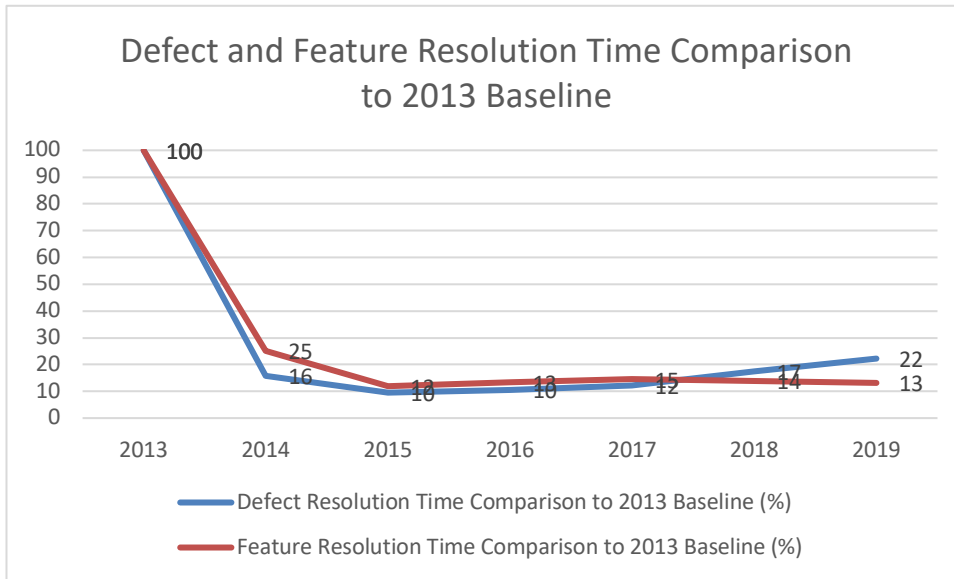
Agile influence on value can be studied and measured by observing various impacted parameters - notably improvement of delivery as measured by cycle time of deliverables (Bhasin, 2012). Agile methodologies espouse an emphasis on releasing working functionality as readily as possible to best benefit the organisation (Reifer et al., 2003). A regular frequent release cadence enables the customer to provide quicker feedback and also enforces the team to make production releases and maintenance cycles as cheap and efficient as possible (Van Cauwenberghe, 2003).



**Figure 8: Average Number of Days to Resolve Task**

On a portfolio level, we see in Figure 8 above that the average number of days to resolve Defects, Features and SAFe® Epics over time has trended downward in line with Agile transformation expectations, whereby faster delivery is touted as one of the principal benefits.

Furthermore, when we look at the improvement below in Figure 9, specifically in regard to defect and feature resolution time over time from Table 3, we see that compared to baseline numbers seen in 2013, both defects and features on average take 15% of the time they took to originally resolve. This suggests a significant improvement in the cadence that tasks are completed.



**Figure 9: Defect and Feature Resolution Time Comparison to 2013 Baseline**

Unfortunately, analysis on a team-by-team level over time for comparison purposes to determine better and poorer performing teams overall by resolution times was difficult to perform. This was because multiple new teams formed, and older ones dissipated over the multi-year period given the differing durations of various company projects.

Figure 9 above shows that, though improved, feature resolution times have plateaued over the last few years. One greater concern is that though the defect resolution is improved, both the graph above and Table 3 show an upward tick in the resolution time of defects, more than doubling from their lowest numbers in 2015 from 21 to 49 days in 2019. It can be argued that the growing complexity of customer software solutions over time could validate this increase, however this also does raise concerns of if whether quality assurance measures have degraded over time in some teams in Momentum Investments. One method of evaluation would be to evaluate if potentially higher priority defects are resolved with greater expediency than lower priority defects to validate if this increased number is a concern.

### 4.3 Prioritisation of Tasks

Kona (2015) in his research found that respondents gauged the delivery of high priority features as one of the most important factors for a successful Agile project. By evaluating the average delivery time of tasks based on tasks priority over time, we can evaluate if this is also the case for Momentum Investments.

Performing a two-way ANOVA tests comparing both the feature and defect task priority against resolution times over the multi-year period, as recorded in Table 4 and Table 5, P-values of greater than 0.05 for both observations failed to reject the null hypothesis. Therefore, we could conclude that there is no significant difference in level of task priority and resolution time when tasks are taken into consideration.

This suggests that there is an equal amount of time being spent in resolving feature and defect tasks across the various software teams regardless of the level of priority. Though complexity of low and high priority tasks can be argued (which unfortunately the data does not sufficiently provide), it is something to further investigate as to why this is the case across the portfolio.

If we expand this ANOVA test analysis to look at feature and defect task priority against resolution times on a year-on-year basis across teams from 2013 to 2019, we similarly observe in Table 6 that the *majority* of P-values are greater than 0.05 year-on-year. This too indicates that overall, year-on-year, there is no significant difference in level of task priority and resolution time across teams when tasks are taken into consideration.

Furthermore, removing external factors such as task complexity, dependencies etc (that cannot be gauged given current data), it is of interest as to why higher priority features/defects are not resolved faster than lower priority feature/defects given Agile advocates for prioritisation of higher value features in its processes.

#### **4.4 Summary of Findings and Impact**

The analysis of several years of multiple Momentums Investments software team project data provides us some insight into the outcomes of a large-scale Agile framework journey.

From a positive point of view, the Agile transformation process has resulted in significantly shorter average resolution times of about 15% of the baseline delivery times recorded in 2013. Additionally, the proportion of tasks delivered in the portfolio as features has seen a positive increase whilst the proportion of tasks that are defects has seen a reduction. From an Agile perspective, this suggests that two factors are being entrenched into these teams. Firstly, teams are practicing more frequent releases suggesting a quicker cadence in order to deliver valued features to customers and also receive quicker feedback. Secondly, Agile promotes quality assurance and testing as critical parts of the development process, ideally meaning that fewer defects are generated and are quicker to resolve. The result being that teams can use more of their time delivering features that customers value rather than resolving systems issues.

However, there are certain factors that do remain a potential point of concern. Despite the proportion of features increasing and defects decreasing, there has only been an overall 5% improvement. The originators of SAFe® highlight an up to 50% increase in productivity, and whilst the definition of productivity can be argued, a 5% improvement is a metric that would need to be evaluated by Momentum Investments stakeholders as a baseline for the future.

Furthermore, though delivery cadence has increased, defect resolution times across the portfolio have seen a doubling from an all-time low. This suggests that potentially quality assurance processes may have room for improvement to better ensure software quality. Feature resolution times across the portfolio have also plateaued for several years after the introduction of Agile, however Momentum Investments would need to define if that is a satisfiable and sustainable velocity or if there is also room for improvement here to deliver features quicker to customers.

Lastly, the data shows that across the defined timeframe, even though features and defects are assigned priority, there appears to be no significant difference in the resolution times despite the differences in priority levels. Since Agile promotes the delivery of higher priority features to customers, this is something for Momentum Investments to investigate and gauge what external factors may be inhibiting this that the current data does not capture. Additionally, high priority defects have the same delivery cadence as lower priority defects. Since working software is the primary measure of progress in Agile transformation, this means that the quality of customer software is impeded.

## **5: RECOMMENDATIONS**

### **5.1 Introduction**

Large-Scale Agile transformations within large organisations notably focus on minimizing time to market and greater flexibility (Papadopoulos, 2015). The SAFe® framework expands on this by scoring enterprises on quality, value delivery, efficiency and agility (Leffingwell, 2019). By assessing several years of project data, we observed that the Agile journey of Momentum Investments software delivery teams has reaped benefits through this SAFe® implementation. This is shown in significant improvements in task resolution times as well as marginal improvements in overall feature delivery and defect management. That being said, there are several areas that the data highlights as areas of improvement that the organisation can review to further their Agile journey.

### **5.2 Greater Focus on Quality Assurance and Testing**

Fowler (2005) states that quality assurance and testing are roles where greater attention and work is required as part of the Agile process with each iteration. The data from Momentum Investments shows that though initially defect resolution times showed a considerable drop with the advent of Large-Scale Agile transformation, this value has started to trend up over time. Pham and Pham (2012), and many other authors, recommend test automation as a critical contributor to success in Agile transformation.

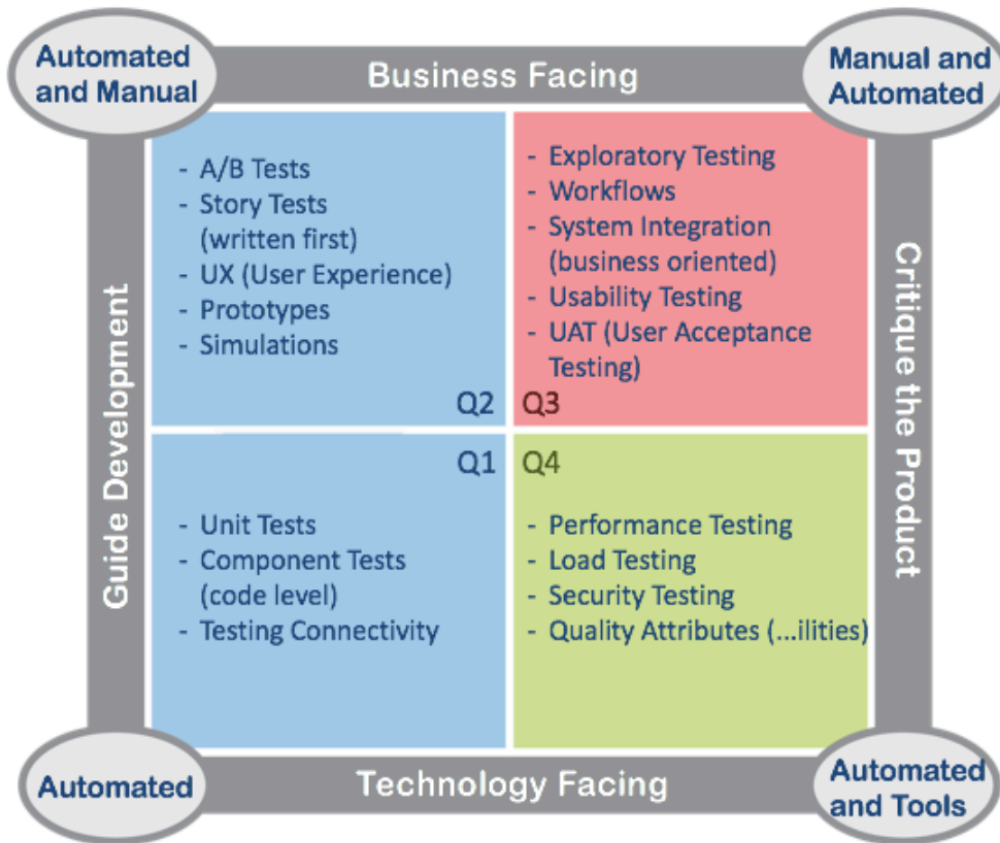
Reifer et al. (2003), in their research, determined that continuous integration and automated build-and-test processes were valued most by participants in Agile transformations, notably as these processes allow quicker feedback both testing new features whilst ensuring that existing functionality remains intact. Furthermore, functional testing ensures the analysis and specification of the system is up-to-date and precise with business requirements.

Though Momentum Investments does practice and promote automated testing as part of its software delivery practices, given the uptick in defect resolution times, a more critical focus on quality assurance and testing practices as part of the delivery pipeline would be recommended to mitigate future defects. This would ideally mean that defects levels should reflect a continued decline, meaning that more time is spent developing new and improved features to clients rather than fixing existing functionality.

Van Cauwenberghe (2003) also notes that these integral testing practices ensure that defects are not only determined earlier but also resolved quicker – meaning they are inherently cheaper to fix in the long run. The result being that average resolution times for defects (barring external factors) should also see a downward trend due to better quality software that is easier to correct. All of this should then have a positive impact on productivity and cost that ultimately delivers improved business value. Furthermore, improved testing translates to a faster delivery rate of customer features, which in turn improves customer satisfaction (Bhasin, 2012).

To achieve this, Zhou and Xu (2008) recommend that quality assurance exercises and activities should occur both earlier on and more frequently in the delivery process. Momentum Investments should assess if indeed quality assurance and testing are occurring frequently enough and to the level required in the individual teams. By assessing the way of work for each team and implementing practical transparent checkpoints in both the development and testing processes that quality assurance has been completed will assist each team to gauge that it is being done correctly. Furthermore, automated build-and-test processes should be defined as benchmark standards for any future feature development or defect resolution processes.

Additionally, the SAFe® framework advocates for Agile testing as a continuous process in that every member of the team is a tester. As shown in Figure 10, the Agile testing matrix provides guidance on what and when to test. This fosters a comprehensive strategy to better ensure quality (Scaled Agile, 2021).



**Figure 10: Agile Testing Matrix**

*Source: Scaled Agile (2021)*

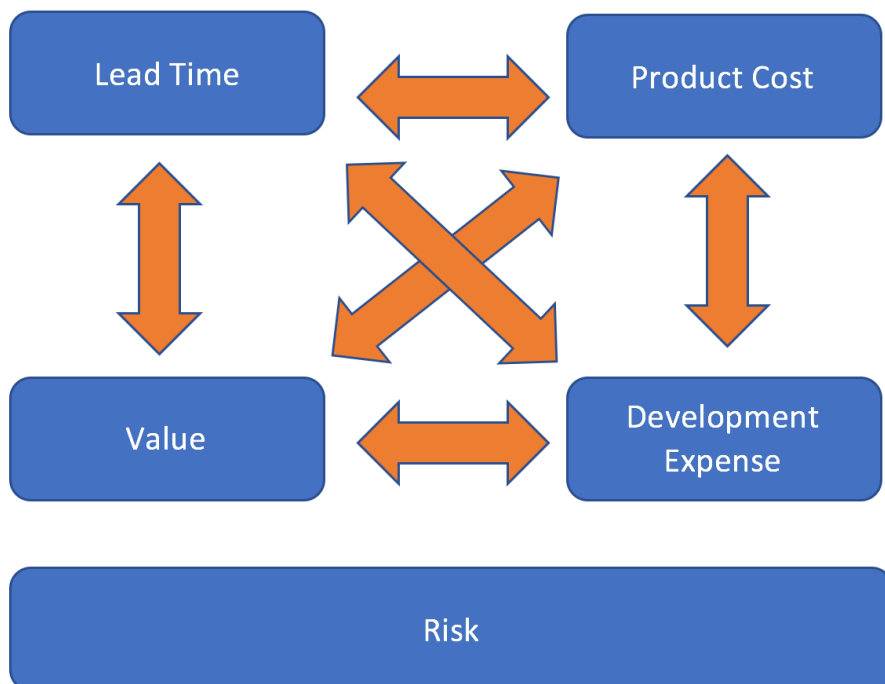
The horizontal axis of the matrix consists of technology facing or business tests. Technology facing tests is code written by developers to evaluate system functionality whilst business facing tests are based on business terminology and created to be understandable by the business user. The vertical axis on the left side guides development by recommending how developers should test their code, typically before and during implementation. The right side advises on ways to test to critique or evaluate the solution against user requirements and specifications (Scaled Agile, 2021).

This matrix provides a useful comprehensive guideline to assess if Momentum Investments current testing practices in its various software delivery teams address the various quadrants presented in the Agile testing matrix.

### 5.3 Improved Prioritisation

The findings indicate that in general across the Momentum Portfolio there appears to be no significant difference in the prioritisation of feature or defect tasks based on resolution times. Ideally, lower priority tasks should have longer resolution times as they would be deferred for higher priority tasks. Since Agile actively promotes the delivery of higher priority features to customers to both deliver value and encourage quicker feedback, focus should be spent determining what factors impact the delivery of higher priority tasks.

The SAFe® framework recommends practicing an economic view in defining and prioritising work in order to deliver features that the market demands. As shown below in Figure 11, economics should be the primary factor of job prioritisation in determining what tasks can be completed in the shortest amount of time to either produce the most revenue, cut expenses or reduce exposure to risk (Mattis, 2021).



**Figure 11: Economic View of Prioritisation**

*Source: Mattis (2021)*

Extending on this economic view, the next principle that SAFe® extols from a prioritisation viewpoint is systems thinking. Within larger organisations, the work one individual or team may deem to be the most important may not be the most important work for the organisation overall. Systems thinking encourages placing clients front and centre and evaluating the whole value stream, particularly in discerning the slowest or most problematic areas. This means prioritising and investing teams to other areas that are most beneficial to the organisation (Mattis, 2021). Delivering the most important features first is a critical component of an effective Agile delivery strategy (Chow & Cao, 2008).

#### **5.4 Better Data**

With Agile transformations, having the right and accurate data becomes essential for organisations to make informed decisions (Kudyba, 2021). This analysis, though able to determine several findings, found several data points that could not be thoroughly analysed because of inconsistent or lack of data in teams, such as task effort/complexity and task dependencies. Additionally, current datapoints analysed in teams could be better and more consistently tracked. This improvement will not only enable Momentum Investments to better gauge how they are performing periodically, but also fine tune team delivery processes.

Agile principles can only be effectively executed with strong leaders as change agents. It is usually perceived that Agile methodology is mainly applicable to development teams, however in reality it actually requires the entire organisation to adjust (Bhasin, 2012). If the Agile implementation is viewed as purely an IT undertaking, then the teams that need to both communicate clarity regarding business value and priority may not be cooperative. This can then hinder the Agile transformation process and the subsequent value that can be derived by the organisation (Ashmore & Runyan, 2014).

Within Momentum Investments it therefore becomes crucial that Agile leaders, sponsors and champions, both at the higher portfolio level and within teams, ensure that these principles and recommendations strongly define what work is

prioritised throughout the various work streams to ensure the greatest benefit to the organisation.

## **6: CONCLUSION**

Agile driven projects highlight the benefits of flexible, speedy, incremental and iterative development. This is done with the drive to deliver more cost-effective, higher quality, valuable features to customers quicker. Furthermore, it proposes improved responsiveness to ever-changing customer needs. Large-scale Agile transformations expand this notion by applying these principles on an enterprise level.

Momentum Investments multi-year, large scale transformation journey, using Scaled Agile Framework for Enterprise (SAFe®) as their framework of choice, has yielded several benefits. Company project data reveals that task resolution times have seen considerable improvements. Additionally, teams have better engaged with the Agile transition, as reflected by more of the portfolio's deliverables constituting features rather than corrections to defects in their software.

However, though this is the case, the data does shows potential areas of improvement for Momentum Investments. Firstly, the improvement and refinement of existing quality assurance and testing practices should be reviewed using the Agile testing matrix. This is recommended to both further reduce defect density levels as well as enable faster and more cost-effective defect rectification. Besides improved software quality, it also translates to more time for its teams to deliver valued customer features.

Secondly, task priority needs to be reassessed by evaluating team backlogs through an economic view and systems thinking viewpoint, as recommended by the SAFe® framework. This should enable the various software delivery teams in Momentum Investments to better deliver higher priority tasks that align to organisational goals.

Thirdly, teams should take a more diligent approach in improved and consistent tracking of task metrics to better enable the organisation to track performance and fine tune delivery processes.

To better achieve this, Agile sponsors at all levels in the organisation must take a proactive approach as change agents to ensure that these recommendations are implemented, monitored and evaluated continuously.

## REFERENCES

- Abbas, N., Gravell, A. M., & Wills, G. B. (2010). The impact of organization, project and governance variables on software quality and project success. 2010 Agile Conference, Orlando, FL.
- Abrahamsson, P., Oza, N., & Siponen, M. T. (2010). Agile software development methods: A comparative review. *Agile software development*, 31-59.
- Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2017). Agile software development methods: Review and analysis. *arXiv:1709.08439*.
- Ambler, S. (2006). *Supersize me*. <https://www.drdoobbs.com/supersize-me/184415491>
- Ashmore, S., & Runyan, K. (2014). *Introduction to agile methods*. Addison-Wesley Professional.
- Awad, M. (2005). A comparison between agile and traditional software development methodologies. *University of Western Australia*, 30, 1-69.
- Barlow, J. B., Giboney, J., Keith, M. J., Wilson, D., Schuetzler, R. M., Lowry, P. B., & Vance, A. (2011). Overview and guidance on agile development in large organizations. *Communications of the Association for Information Systems*, 29(2), 25-44.
- Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., & Sutherland, J. (2001a). *Manifesto for agile software development*. <http://agilemanifesto.org/>
- Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., & Sutherland, J. (2001b). *Principles behind the agile manifesto*. <http://agilemanifesto.org/principles.html>
- Bhasin, S. (2012). Quality assurance in agile: A study towards achieving excellence. 2012 Agile India, Le Meridien, Bengaluru, India.
- Blaxter, L. (2010). *How to research*. McGraw-Hill Education (UK).
- Chow, T., & Cao, D.-B. (2008). A survey study of critical success factors in agile software projects. *Journal of Systems and Software*, 81(6), 961-971.
- Cockburn, A. (2000). *Reexamining the cost of change curve*. [https://ronjeffries.com/xprog/articles/cost\\_of\\_change/](https://ronjeffries.com/xprog/articles/cost_of_change/)

- Conboy, K., & Carroll, N. (2019). Implementing large-scale agile frameworks: challenges and recommendations. *IEEE Software*, 36(2), 44-50.
- Cresswell, J. (2005). *Research design: Qualitative, quantitative and mixed methods approaches*. Sage Publications.
- Davis-Kean, P. E., Jager, J., & Maslowsky, J. (2015). Answering developmental questions using secondary data. *Child development perspectives*, 9(4), 256-261. <https://doi.org/10.1111/cdep.12151>
- de Azevedo Santos, M., de Souza Bermejo, P. H., de Oliveira, M. S., & Tonelli, A. O. (2011). Agile practices: An assessment of perception of value of professionals on the quality criteria in performance of projects. *Journal of Software Engineering and Applications*, 4(12), 700.
- Dikert, K., Paasivaara, M., & Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, 119, 87-108. <https://doi.org/https://doi.org/10.1016/j.jss.2016.06.013>
- Dingsøyr, T., & Lassenius, C. (2016). Emerging themes in agile software development: Introduction to the special section on continuous value delivery. *Information and Software Technology*, 77, 56-60.
- Dingsøyr, T., & Moe, N. B. (2014). Towards principles of large-scale agile development. International Conference on Agile Software Development, Rome, Italy.
- Doherty, D. (2022). *Exogenous vs. endogenous: What's the difference?* <https://www.zippia.com/advice/exogenous-vs-endogenous/#:~:text=Endogenous%2C%20therefore%2C%20means%20within%20a,means%20something%20outside%20the%20system>.
- Dreissigacker, U. (2022). *Why agile is so popular in project management*. <https://blog.ganttpro.com/en/why-agile/>
- Ebert, C., & Paasivaara, M. (2017). Scaling agile. *IEEE Software*, 34(6), 98-103.
- Fowler, M. (2005). *The new methodology*. <https://www.martinfowler.com/articles/newMethodology.html>
- Gandomani, T. J., Zulzalil, H., Ghani, A. A. A., & Sultan, A. B. M. (2013). Towards comprehensive and disciplined change management strategy in agile transformation process. *Research Journal of Applied Sciences, Engineering and Technology*, 6(13), 2345-2351.

- Goldstein, I. (2013). *Scrum shortcuts without cutting corners: Agile tactics, tools, & tips*. Addison-Wesley.
- Guo, Y. M., & Klein, B. D. (2016). Longitudinal studies in information systems research: Practices, findings, and gaps. MWAIS 2016 Proceedings, University of Wisconsin, Madison, Wisconsin.
- Hammer, E. R. (1977). Validity in longitudinal research. *Social Science*, 52(3), 158-168. <http://www.jstor.org/stable/41886176>
- IQbusiness. (2018). *The state of agile in South Africa 2018*. [https://iqbusiness.net/wp-content/uploads/2018/11/Agile\\_Report\\_2018.pdf](https://iqbusiness.net/wp-content/uploads/2018/11/Agile_Report_2018.pdf)
- Kaplan, R. S., & Norton, D. P. (2008). *The execution premium: Linking strategy to operations for competitive advantage*. Harvard Business Press.
- Kona, C. T. (2015). *The scrum framework for software development projects in financial services organisations in South Africa*. [Master's thesis, Witwatersrand University]. Johannesburg.
- Korhonen, K. (2012). *Supporting agile transformation with defect management in large distributed software development organisation*. [Doctoral Thesis, Tampere University of Technology].
- Kruchten, P. (2001). Agility with the RUP. *Cutter IT journal*, 14(12), 27-33.
- Kudyba, S. D. C., Agnel. (2021). *Build a better dashboard for your agile project*. <https://hbr.org/2021/07/build-a-better-dashboard-for-your-agile-project>
- Kwak, Y. H., & Stoddard, J. (2004). Project risk management: Lessons learned from software development environment. *Technovation*, 24(11), 915-920.
- Laanti, M. (2012). *Agile methods in large-scale software development organisations: Applicability and model for adoption*. [Doctoral thesis, University of Oulu].
- Lal, R., & Clear, T. (2017). Scaling agile at the program level in an Australian software vendor environment: A case study. Australasian Conference on Information Systems, Hobart, Tasmania, Australia.
- Leedy, P. D., & Ormrod, J. E. (2005). Practical research: Planning and design. 8th. *Upper Saddle River, NJ: Merrill Prentice Hall*.
- Leffingwell, D. (2019). *Welcome to Scaled Agile Framework® 4.6!* <https://www.scaledagileframework.com/about/>

- Li, J., Moe, N. B., & Dybå, T. (2010). Transition from a plan-driven process to scrum: A longitudinal case study on software quality. Proceedings of the 2010 ACM-IEEE international symposium on empirical software engineering and measurement, Bolzano-Bozen, Italy.
- Lindvall, M., Muthig, D., Dagnino, A., Wallin, C., Stupperich, M., Kiefer, D., & Kahkonen, T. (2004). Agile software development in large organizations. *Computer*, 37(12), 26-34.
- Mattis, A. (2021). *The challenge of economic prioritization – agility planning*. <https://scaledagile.com/blog/the-challenge-of-economic-prioritization/>
- Mbuya, D. P. (2016). *Agile and conventional methodologies: An empirical investigation of their impact on software quality parameters*. [Doctoral thesis, University of South Africa].
- McCaston, M. K. (2005). Tips for collecting, reviewing, and analyzing secondary data. *HLS Advisor*, 27, 1-9.
- Paasivaara, M. (2017). Adopting SAFe to scale agile in a globally distributed organization. 2017 IEEE 12th International Conference on Global Software Engineering, Buenos Aires, Argentina.
- Paasivaara, M., Behm, B., Lassenius, C., & Hallikainen, M. (2018). Large-scale agile transformation at Ericsson: A case study [journal article]. *Empirical Software Engineering*, 23(5), 2550-2596. <https://doi.org/10.1007/s10664-017-9555-8>
- Papadopoulos, G. (2015). Moving from traditional to agile software development methodologies also on large, distributed projects. *Procedia-Social and Behavioral Sciences*, 175, 455-463.
- Pham, A. T., & Pham, D. K. (2012). *Business-driven IT-wide agile (scrum) and kanban (lean) implementation: An action guide for business and IT leaders*. CRC Press.
- Pinsonneault, A., & Kraemer, K. (1993). Survey research methodology in management information systems: An assessment *Journal of Management Information Systems*, 10(2), 75-105. <https://doi.org/10.1080/07421222.1993.11518001>
- Pitkänen, A. (2015). *Agile transformation: A case study*. [Master's Thesis, Aalto University].

- Poth, A., Kottke, M., & Riel, A. (2020). Evaluation of agile team work quality. International Conference on Agile Software Development, Copenhagen, Denmark.
- Ramesh, B., Cao, L., Mohan, K., & Xu, P. (2006). Can distributed software development be agile? *Communications of the ACM*, 49(10), 41-46.
- Reifer, D. J., Maurer, F., & Erdogmus, H. (2003). Scaling agile methods. *IEEE Software*, 20(4), 12-14.
- Scaled Agile. (2021). *Agile Testing*.  
<https://www.scaledagileframework.com/agile-testing/>
- Scaled Agile. (2022). *Epics*. <https://www.scaledagileframework.com/epic/>
- Schwaber, K., & Beedle, M. (2002). *Agile software development with scrum. Series in agile software development* (Vol. 1). Prentice Hall Upper Saddle River.
- Serra, C. E. M., & Kunc, M. (2015). Benefits realisation management and its influence on project success and on the execution of business strategies. *International Journal of Project Management*, 33(1), 53-66.
- Sfetsos, P., & Stamelos, I. (2010). Empirical studies on quality in agile practices: A systematic literature review. 2010 Seventh International Conference on the Quality of Information and Communications Technology, Porto, Portugal.
- Shine technologies. (2003). *Agile methodologies survey results*.  
<http://www.shinotech.com/resources/ShineTechAgileSurvey2003-01-17.pdf>
- Smith, A. K., Ayanian, J. Z., Covinsky, K. E., Landon, B. E., McCarthy, E. P., Wee, C. C., & Steinman, M. A. (2011). Conducting high-value secondary dataset analysis: An introductory guide and resources. *Journal of general internal medicine*, 26(8), 920-929.
- Tarrant, H. (2016). 'Big Four' IT spending tops R30 billion a year.  
<https://www.moneyweb.co.za/news/tech/big-four-it-spending-tops-r30-billion-a-year/>
- Terlizzi, M. A., & Albertin, A. L. (2017). IT benefits management in financial institutions: Practices and barriers. *International Journal of Project Management*, 35(5), 763-782.

- The Standish Group International. (2015). *CHAOS report 2015*.  
[https://www.standishgroup.com/sample\\_research\\_files/CHAOSReport2015-Final.pdf](https://www.standishgroup.com/sample_research_files/CHAOSReport2015-Final.pdf)
- Tremolada, G. (2015). *Bridge's transition model*.  
<https://frontlinemanagementexperts.wordpress.com/2015/07/03/bridges-transition-model/>
- Turk, D., France, R., & Rumpe, B. (2002). Limitations of agile software processes. Third International Conference on Extreme Programming and Flexible Processes in Software Engineering, Alghero, Italy.
- Van Cauwenberghe, P. (2003). Refactoring or upfront design? Conference Proceedings XP 2001, 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering, Sardinia, Italy.
- Vartanian, T. P. (2010). *Secondary data analysis*. Oxford University Press.
- Venkatesh, A., & Vitalari, N. P. (1991). Longitudinal surveys in information systems research: An examination of issues, methods, and applications. *The Information Systems Challenge: Survey Research Methods; Harvard University Press: Location, UK*, 115-144.
- Yatzeck, E. (2012). *Lead and lag measures for agile transformation*.  
<https://www.thoughtworks.com/insights/blog/lead-and-lag-measures-agile-transformation>
- Zhou, Y., & Xu, B. (2008). Predicting the maintainability of open source software using design metrics. *Wuhan University Journal of Natural Sciences*, 13(1), 14-20.
- Zwikael, O., & Smyrk, J. (2012). A general framework for gauging the performance of initiatives to enhance organizational value. *British Journal of Management*, 23, S6-S22.

# ANNEXURES

## Validity and Reliability

### *External Validity*

External validity relates to the extent to which findings are likely to have broader applicability beyond the focus of a study (Blaxter, 2010) . Attrition has a potential impact on compromising external validity (Bergman & Magnusson, 1990), which given the nature of movement within an organisation becomes a factor that is very difficult to control over a given time period.

However, given this study did not look at a smaller sample group but rather the entire population, concerns of applicability to a broader population did not necessarily apply. However, the findings of this study may be of value to other agile practitioners as a revelatory study.

### *Internal Validity*

Leedy and Ormrod (2005) define internal validity as the extent to which its design and the data it yields allow the researcher to draw accurate conclusions about cause-and-effect relationships for the data. The internal validity of longitudinal studies are susceptible to attrition (Hammer, 1977), however given the use of a secondary data set, measures to reduce or address attrition cannot be pursued.

Smith et al. (2011) state that for large survey datasets, a good first assessment is reading the questions as they were asked. The importance in a validity study is determining how and why the question was asked and the data was collected. Certain questions have face value whilst others are open to interpretation. We strived to increase validity by ensuring that the metrics we gleaned from the data fit to our own requirements for this study.

## **Reliability**

Reliability is concerned with the way in which the research project has been carried out and whether it was done in such a way that subsequent studies would come up with essentially the same results (Blaxter, 2010).

Since this study does deal with secondary data, concerns of reliability are warranted. To ensure reliability all data artefacts used and produced regarding process notes, raw data, coded data as well as any categories for data reconstruction and synthesis were recorded for auditability purposes.

## **Findings**

<b>Year</b>	<b>Defect Count</b>	<b>Proportion Defect Tasks (%)</b>	<b>Feature Count</b>	<b>Proportion Feature Tasks (%)</b>
<b>2013</b>	214	26	609	74
<b>2014</b>	2714	29	6502	71
<b>2015</b>	3343	32	7069	68
<b>2016</b>	2776	25	8381	75
<b>2017</b>	2570	24	7948	76
<b>2018</b>	2425	23	8337	77
<b>2019</b>	2189	21	8020	79

**Table 1: Defect vs Feature Proportion on a Portfolio Level Over Time**

<b>Year</b>	<b>Average Defect Count Per Team</b>	<b>Proportion Average Defect Count Per Team (%)</b>	<b>Average Feature Count Per Team</b>	<b>Proportion Average Feature Count Per team (%)</b>
<b>2013</b>	18	31	41	69
<b>2014</b>	160	36	283	64
<b>2015</b>	197	39	307	61
<b>2016</b>	126	26	364	74
<b>2017</b>	107	29	265	71
<b>2018</b>	93	29	232	71
<b>2019</b>	81	27	223	73

**Table 2: Average Defect vs Feature Proportion per Team Over Time**

<b>Year</b>	<b>Defect Average Days Resolve*</b>	<b>Defect Resolution Time Comparison to 2013 Baseline* (%)</b>	<b>Feature Average Days Resolve*</b>	<b>Feature Resolution Time Comparison to 2013 Baseline (%)</b>
<b>2013</b>	<b>222</b>	100	<b>285</b>	100
<b>2014</b>	<b>35</b>	16	<b>72</b>	25
<b>2015</b>	<b>21</b>	10	<b>34</b>	12
<b>2016</b>	<b>23</b>	10	<b>38</b>	13
<b>2017</b>	<b>27</b>	12	<b>41</b>	15
<b>2018</b>	<b>39</b>	17	<b>40</b>	14
<b>2019</b>	<b>49</b>	22	<b>37</b>	13

\*Rounded to nearest whole number

**Table 3: Average Days to Resolve Defects/Features Over Time**

ANOVA: Two-Factor Without Replication

<i>SUMMARY</i>	<i>Count</i>	<i>Sum</i>	<i>Average</i>	<i>Variance</i>
High	7	539.4185095	77.05978708	7032.007775
Low	7	594.3101224	84.90144606	12895.17669
Medium	7	623.865327	89.12361814	15042.3027
2013	3	963.4865119	321.1621706	3068.060562
2014	3	252.9551913	84.31839711	307.2831082
2015	3	112.872869	37.62428967	58.09111892
2016	3	102.7515812	34.25052705	63.25465295
2017	3	115.1441058	38.38136859	18.65642669
2018	3	111.2043219	37.0681073	8.238162334
2019	3	99.1793779	33.05979263	22.76473399

ANOVA

<i>Source of Variation</i>	<i>SS</i>	<i>df</i>	<i>MS</i>	<i>F</i>	<i>P-value</i>	<i>F crit</i>
Rows	524.660203	2	262.3301015	0.479284916	<b>0.630587033</b>	3.885293835
Columns	203248.8857	6	33874.81428	61.89029554	2.49523E-08	2.996120378
Error	6568.037328	12	547.336444			
Total	210341.5832	20				

**Table 4: Two-way ANOVA Test for Feature Task Priority vs Average Resolution Time (Days) over Time Period**

- Ho: There is no significant difference between the level of feature priority and the average resolution time.
- H1: There is a difference between the level of feature priority and the average resolution time.

From the above output we can see that the rows have a **P-value>0.05**, therefore we fail to reject the null hypothesis and conclude that there is no significant difference between level of feature task priority and the average resolution time for a corresponding time period.

ANOVA: Two-Factor Without Replication

<i>SUMMARY</i>	<i>Count</i>	<i>Sum</i>	<i>Average</i>	<i>Variance</i>
High	7	437.3115728	62.47308182	4702.421156
Low	7	465.8066126	66.5438018	7482.588881
Medium	7	428.1566734	61.16523906	6524.41662
2013	3	711.1395202	237.0465067	370.3647588
2014	3	106.3717864	35.45726213	1.803191423
2015	3	76.81050391	25.6035013	65.4500428
2016	3	69.35173351	23.1172445	10.00559516
2017	3	63.54677539	21.18225846	158.2880192
2018	3	105.4667631	35.15558768	158.6307517
2019	3	198.5877763	66.19592545	559.1915001

<i>ANOVA</i>						
<i>Source of Variation</i>	<i>SS</i>	<i>df</i>	<i>MS</i>	<i>F</i>	<i>P-value</i>	<i>F crit</i>
	110.15701		55.0785095	0.26048923	<b>0.77492177</b>	3.8852938
Rows	91	2	7	2	<b>6</b>	35
	109719.24		18286.5415	86.4846778		2.9961203
Columns	92	6	4	8	3.59038E-09	78
	2537.3106		211.442558			
Error	99	12	3			
	112366.71					
Total	7	20				

**Table 5: Two-way ANOVA Test for Defect Task Priority vs Average Resolution Time (Days) Over Time Period**

- Ho: There is no significant difference between the level of defect priority and the average resolution time.

- H1: There is a difference between the level of defect priority and the average resolution time.

From the above output we can see that the rows have a **P-value>0.05**, therefore we fail to reject the null hypothesis and conclude that there is no significant difference between level of defect task priority and the average resolution time for a corresponding time period.

Year	Feature Priority vs Average Resolution Time P-value Across Teams	Defect Priority vs Average Resolution Time P-value Across Teams
2013	0.193874627	N/A - Limited Data
2014	0.514157896	0.762193937
2015	0.238939158	0.682222499
2016	0.192385854	0.307389081
2017	0.493831157	0.046503822
2018	0.252268634	0.932533378
2019	0.860310696	N/A - Limited Data

**Table 6: Two-way ANOVA Tests P-values for Feature and Defect Priority vs Average Resolution Time (Days) Across Teams Per Year**

- Ho: There is no significant difference between the level of task priority and the average resolution time across teams.
- H1: There is a difference between the level of task priority and the average resolution time across teams.

From the above output we can see that the *majority* of the P-values have a **P-value > 0.05**, therefore we fail to reject the null hypothesis for the *majority* of the values and can conclude that overall, for the majority of the years, there is no significant difference between levels of feature and defect task priority and the average resolution time across teams for the corresponding time period.

**NOTE:** To derive the above P-values, only teams that's categorized tasks in all the high, low and medium priority levels could be used. Therefore, these values are determined from a subset of teams – something that could not be controlled in data collection given the use of secondary data. This is why no values could be derived for defect priority P-values between Teams for 2013 and 2019 due to limited available data.