A CRITICAL ANALYSIS OF THE IMPACT OF DEVOPS ON INFORMATION TECHNOLOGY ORGANISATIONS: A CASE STUDY

by

Austin Mudadi

submitted in accordance with the requirements

for the degree of

MASTER OF SCIENCE

in the subject

COMPUTING

at the

UNIVERSITY OF SOUTH AFRICA

Supervisor: Prof H.H. Lotriet

September 2022

DECLARATION

Name: Austin Mudadi

Student number: 66652243

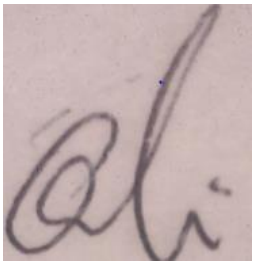
Degree: Master of Science in Computing

A critical analysis of the impact of DevOps on information technology organisations: a case study

I declare that the above dissertation is my own work and that all the sources that I have used or quoted have been indicated and acknowledged by means of complete references.

I further declare that I submitted the dissertation to originality checking software and that it falls within the accepted requirements of originality.

I further declare that I have not previously submitted this work or part of it for examination at Unisa for another qualification or at any other higher education institution.

Student signature:  Date: 02-09-2022

ACKNOWLEDGEMENTS

This research has indeed been a great learning experience for me. I would like to take this opportunity to express my appreciation to all the people who made significant contributions that helped to make the writing of this dissertation a success.

A big thank you to my supervisor, Professor Lotriet, for advice and support. I want to also express my profound gratitude to Professor Venter for her expert views and insights that gave me positive direction. A special thank you to all the interviewees who sacrificed their time to share their experiences with me. Lastly, but not least, I am thankful for my family's support.

ABSTRACT

Volatile market conditions push businesses to develop strategies to adapt. Information technology organisations are no exception, and some of them have adopted DevOps, while others are in the process of implementing it. DevOps is a software development philosophy that involves the working together of various departments, such as development, operations, and other software development departments, with the common aim of accomplishing common business objectives. However, little has been done to establish the footprint of DevOps in information technology organisations, hence, this phenomenon remains inadequately communicated and not fully understood in both the practitioner and academic research communities. The current study therefore used a single case study to build a theory about different scenarios of successful DevOps implementation in a multinational company that is headquartered in South Africa. Various DevOps role players were engaged to obtain their perceptions regarding the DevOps adoption process that the organisation went through. An overview of the existing literature provided information on how organisations have restructured during the DevOps implementation process. The study then compared the existing views on DevOps as found in the literature, with the details of real scenarios that occurred during DevOps adoption. This research study aimed to provide an adequate understanding of an appropriate DevOps restructuring approach that would be suitable for information technology organisations, and this will help other institutions to make informed decisions that positively affect the process of migration towards the adoption of DevOps.

Keywords: DevOps, Development and Operations, Culture, Automation, Deployment pipeline; Testing; Monitoring; Measurement, Tools

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

The following abbreviations are used throughout the study:

CPU	Computer processing unit
FTE	Full time equivalent
IT	Information technology
MTBF	Mean time between failure
RAM	Random access memory
SBG	Standard Bank Group
SDLC	Software Development Life Cycle
UNISA	University of South Africa

CHAPTER 1:
SKETCHING THE BACKGROUND

1.1 INTRODUCTION

The evolution of software development engineering stems from what appears to be a rescue mission, and in response to the widespread software crisis of the past decade (Khan, Khan, Khan, Khan & Whangbo, 2022). The critical moment was characterised by uncountable delays in software delivery and decreasing software quality (Díaz, López-Fernández, Pérez & González-Prieto, 2021). Digital transformation, however, established scientific, quantitative and management perspectives on the evolution of software development engineering through the introduction of Software Development Life Cycle (SDLC) models, such as Agile, Waterfall, Lean and Continuous Improvement (Battina, 2021). The aim of these methodologies is to successfully design and develop cost-effective software that meets customer needs (Gall & Pigni, 2021). Although these software development methodologies have improved the software engineering processes, few efforts have been made to measure the degree of customer satisfaction in most cases, hence, the introduction of DevOps to further improve the processes (Gokarna & Singh, 2021).

The term “DevOps” is a merging of two terms, namely, development and operations. It is a practice during which different software development stakeholders work together, with the main objective of continuously delivering high quality software to the customer using the most efficient systems (Mangot, 2016). Delivering software of uncompromised quality with high and uninterrupted speed facilitates rapid feedback loops that allow continuous product improvement, and enables the organisation to gain a competitive edge in the market (Wurster, Colville, Haight, Tripathi & Rastogi, 2013).

According to Leite, Rocha, Kon, Milojicic and Meirelles (2019), both the practitioner and academic research communities do not fully comprehend the restructuring process that a specific organisation needs to go through to successfully implement DevOps, taking into consideration the existing structure of the organisation, its vision and the market environment in which it operates. In addition, little has been done to assess the impact that the DevOps automation tools have on the overall productivity of organisations (Crowley, McQuillan & O'Brien, 2018).

The current study was motivated by the DevOps shortcomings highlighted above. The current study, therefore, aims to investigate the DevOps movement using both literature and a single case study to consider and evaluate the strategies that can be used to successfully implement DevOps in information technology (IT) firms.

1.2 BACKGROUND

DevOps is a system which aims to create a teamwork environment within companies. The ideal is that the development and the operations teams share the same vision and have common objectives (Díaz *et al.*, 2021). It is regarded as the most effective way to eliminate barriers between the development and operations personnel, and by so doing, promotes the close collaboration that yields agility, skills diversity, and productivity (Leite *et al.*, 2019).

The Agile methodology contains some of the founding principles of DevOps, such as: achieving customer satisfaction through the continuous delivery of high-quality software, and effective communication between the customers and software development stakeholders (Perera, Silva & Perera, 2017).

1.2.1 DevOps definitions

Currently, the concept of DevOps does not have a popular definition that is accepted by most (Gall & Pigni, 2021). This section provides a brief discussion of how DevOps has been defined over the years.

- According to Hüttermann (2012), DevOps is a software development process whereby the development and operations personnel work closely together with other stakeholders to deliver quality software to the customers timeously.
- DevOps is an organisational approach that seeks to promote a work environment that is characterised by interdepartmental collaboration, and employees who are understanding and compassionate (Erich, Amrit & Daneva, 2017).
- Sharma (2013) defines DevOps as a process of continuously delivering high quality software to the customer in small batches, thereby enabling bottlenecks to be detected earlier, fixed quickly and cheaply, and consequently, helping organisations to increase their market share.

- DevOps is an exercise which aims to achieve a reduction of time between the release of software changes, ensuring that the changes are fully operational, and that the production complies with the required expectations (Zhu, Bass & Champlin-Scharff, 2016).
- Kim has another view of DevOps (2014), namely, that DevOps is a practice that values close collaboration between the development and operations teams in order to facilitate quick software deployment, while ensuring that the functional and non-functional requirements of such deployments are of good quality.
- López-Fernández *et al.* (2021) suggested that DevOps is a cultural movement that aims to bring developers and operators into close and effective cooperation, in order to speed up and improve the delivery of software to the customer.

To summarise: The most important common factor within the above DevOps definitions is that there should be a continuous delivery of quality software to the customer. Sharma (2017) emphasised that continuous delivery is the major requirement of DevOps, and without it DevOps ceases to exist. According to Wurster *et al.* (2013), continuous delivery is achievable through continuous integration, automated testing, and improvement of software, as well as the teamwork among software development stakeholders that is facilitated using machines and technology. The points raised in these different definitions, including the need to produce high quality software through the automation of the entire system and having all stakeholders work together, increase the understanding of the implementation of DevOps.

1.2.2 **Early DevOps adoptions**

Big web companies, such as Google, Amazon and Netflix, started to deploy software on a daily basis long before the concept of DevOps was introduced (Díaz, Perez, Yague, Villegas & De Antona, 2019). Their objective was to continuously release software in small batches for use by their customers. This continuous software delivery was triggered by the desire to meet growing customer demands and to satisfy different customer expectations. Although the term DevOps was not used, these companies were applying the same DevOps concept of continuous software delivery under the Agile methodology (Crowley *et al.*, 2018). The challenge during those early years was to get a strategy that would enable them to speed up the delivery of their software in

small batches without compromising the software's operational stability. The lack of close collaboration between development and support staff was one of the causes of delays in software delivery (López-Fernández *et al.*, 202).

The success enjoyed by big web companies, such as International Business Machines Corporation (IBM), who adopted DevOps practices during its early introduction phase caused many other companies to implement DevOps practices in their organisations. This has also led to the establishment of DevOps service providers to help more companies to adopt DevOps, such as, for example, Puppet (Wurster *et al.*, 2013).

The ability to deliver software to customers faster and more continuously has given companies a competitive advantage because they not only meet customer demands but also take advantage of rapid feedback loops to improve their products as per customer requirements (Lwakatare *et al.*, 2016).

1.2.3 Why DevOps improves the functioning of information technology organisations

The principal driving factor for the implementation of DevOps is the need to bridge the gap between development and operations in IT organisations (Shrikanth, 2018). Organisations are often not able to release software to production in good time since their development and operations departments exist as isolated entities (Hemon, Lyonnet, Rowe & Fitzgerald, 2020). DevOps is an effective way to foster collaboration between developers, operators and testers so that they work together, share experiences and make each other's skills more applicable across different departments (Caprarelli, Nitto & Tamburri, 2019). Close collaboration is also extended to customers and other stakeholders to take advantage of fast feedback loops which result in early error detection and customised product improvement (Gall & Pigni, 2021).

According to Sharma (2017), IT firms that aim to maintain market leadership or gain a substantial slice of the market share adopt DevOps. The principal factor behind this is the automation of the entire system using open source or licensed tools. Besides reliability and dependability, automation gives a business the ability to satisfy customer expectations through the production of high-quality software, in high volumes, across different platforms, simultaneously (Shrikanth, 2017).

1.2.4 **Overview of the DevOps and Agile relationship**

DevOps is built on the concepts of the Agile methodology (Kim, 2014). Agile principles promote good working relationships among software development stakeholders, enables effective communication within the team, and result in the production of high-quality software. As such, these principles are all incorporated in DevOps practices (Jabbari, Bin Ali, Petersen & Tanveer, 2018). However, unlike Agile, DevOps emphasises the automation of the entire deployment pipeline (Caprarelli *et al.*, 2019). Automation brings consistency in terms of the quality of software produced and the speed with which it is produced, thus leading to a better level of customer satisfaction (Sánchez-Gordón & Colomo-Palacios, 2018).

DevOps also ensures that there is a collective working relationship between developers and operators to promote teamwork and avoid conflict of interest (Hüttermann, 2012). This close collaboration breaks down the segregation of duties, and yields teamwork that is characterised by mutual trust and commitment, recognising that all team members are tasked with achieving common goals (Lwakatare *et al.*, 2019).

DevOps broadens the scope of Agile principles and has the two main functions of continuous integration and continuous deployment (Vanderjack, 2019). The main objective of these two core aspects of DevOps is to produce quality software continuously in small batches at a high speed. Contrary to Agile, this means that customers get software as soon as it is released, and by so doing, the business quickly enjoys the benefits of the software (Leite *et al.*, 2019). In addition, the speedy software releases also enable the businesses to become more competitive as it allows them to release their products to the market faster than other competitors. This also gives the business enough time to get customer feedback and experiment on their product to improve its non-functional and functional requirements (Riungu-Kalliosaari, Mäkinen, Lwakatare & Männistö, 2016).

Table 1.1 lists some improvements that have been implemented by DevOps to address Agile's shortcomings.

Table 1.1: Improvements to Agile by DevOps facilities

Agile methodology drawbacks	DevOps solutions to solve problems
Delayed delivery of components to the customer.	Testing and releasing the components when they are completed.
Completed software components are not compatible with each other.	Test automation of parts obtained by dividing the project.
Quality of product is not ensured properly prior to release.	Test automation helps the quality assurance.
Developer team and IT operations team are not cooperating.	Developer team and IT operations team agree upon their responsibilities and their goals.

Source: Banica, Radulesca, Rosca & Hagiu, 2017:43

1.3 OVERVIEW OF DEVOPS IMPLEMENTATION PROBLEMS

Organisations with separate departments that have been functioning well, will not necessarily accept the idea of DevOps, which will seek to break down these silos (Kuusinen *et al.*, 2018). This is reiterated by Caprarelli *et al.* (2019) who indicated that if the development and operation departments are separate in terms of business goals and functionality, DevOps implementation will be a problem.

The implementation of DevOps requires the organisation to embrace teamwork and close collaboration (Caprarelli *et al.*, 2019). The close collaboration approach of DevOps might be a problem for some senior executives who might feel their power and jobs are being endangered by it and they might resist its implementation (Sharma, 2017; Cameron & Green, 2019). According to Wurster *et al.* (2013), it is difficult to break down barriers between departments that have been working in a certain way for a long time within the organisation without disrupting productivity. Since current scientific literature lacks a common definition of DevOps and a consensus of what it entails, it is often difficult for DevOps practitioners to implement it (Gall & Pigni, 2021).

According to Ebert, Gallardo, Hernantes and Serrano (2016), software development firms that seek to deliver high-quality software to their customers in a fast and reliable manner normally implement DevOps practices (Ebert *et al.*, 2016). The key being the automation of the entire workflow and close collaboration between the development, quality assurance and operations personnel (Claps, Svensson & Aurum, 2015).

Automation requires that the processes involved should be properly planned to avoid unintended, undesirable and complex consequences (Wurster *et al.*, 2013).

The above section highlighted some of the problems that organisations could face during the adoption of DevOps.

1.3.1 **Problem statement**

There is a limited understanding of the transformational strategy that is appropriate for DevOps implementation to effectively deliver and maintain software applications (Leite *et al.*,2019), and a general lack of knowledge about the degree of usefulness of DevOps tools with respect to accomplishing IT objectives (Crowley et al., 2018). The typical setup of an organisation without DevOps is illustrated in Figure 1.1.

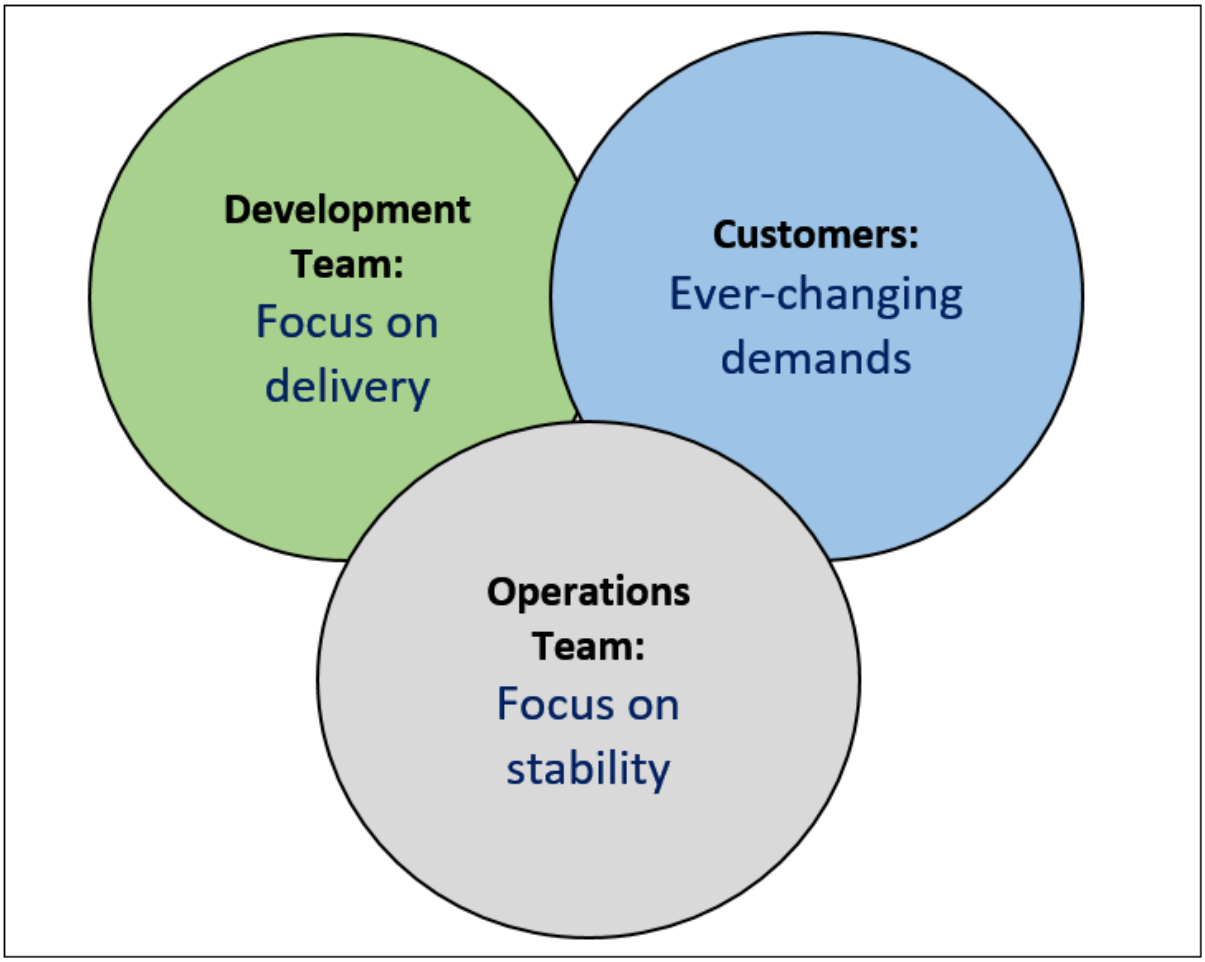


Figure 1.1: Misalignment of goals in a typical product development organisation

Source: Adapted from Pylayeva, 2017:12

The current research focuses on a DevOps implementation strategy with specific reference to the ways an IT organisation is transformed during DevOps adoption. Using a systematic literature study and interviews to engage practitioners who are actively involved with the implementation of DevOps, the researcher explored DevOps thoroughly to highlight its meaning, the processes involved, benefits derived from its adoption, and the unsatisfactory aspects that may be associated with it.

1.4 OBJECTIVE AND PURPOSE OF THE STUDY

The purpose of the current study was to obtain detailed information about how organisations may need to change their culture and technology when implementing DevOps, and the impact the change may have on their businesses processes. The information obtained by this study is expected to be helpful for practitioners, organisations and academic researchers, and enable them to make informed decisions about how to effectively adopt DevOps.

1.5 RESEARCH GAP AND QUESTIONS

DevOps has been in existence for more than a decade (Crowley *et al.*, 2018). However, despite this period there are still many unknown aspects that need to be explored academically and practically. For example, a phenomenon that needs to be investigated, is the way organisations establish new arrangements and relations within their structures when embarking upon the implementation of DevOps (Leite *et al.*, 2019). The degree to which the DevOps tools can deliver the expected outcomes is another area that needs to be researched (Crowley *et al.*, 2018).

The focus of this dissertation is a critical analysis of the impact of DevOps on IT organisations. The primary research question thus is: How should organisations restructure during DevOps implementation to mitigate the related challenges and enhance the delivery of high-quality software to their customers?

This question can be refined into three sub-questions:

- How should organisations restructure when implementing DevOps?
- What challenges are encountered during restructuring and how can these challenges be mitigated?

- To what extent are the DevOps tools used in organisations able to deliver the expected business results?

1.6 SIGNIFICANCE OF THE STUDY

The prime objective of this research is to provide academic researchers and practitioners with a detailed account of DevOps practices, focusing on the way DevOps affects the structure of organisations, and its effectiveness and efficiency regarding the accomplishment of organisational goals. Based on the research outcome, the expectation was that the current study would create new opportunities for future DevOps research.

1.7 RESEARCH METHODOLOGY

The current study employed an interpretive approach and used both a systematic literature review and semi-structured interviews to collect data (Rahman, 2020). An interpretive approach is useful when considering the opinions of people who often perceive things differently (Halkias & Neubert, 2020). The current study followed a case study methodology.

1.8 STRUCTURE OF THE DISSERTATION

Chapter 1 provided a brief background to the aims of the research, the research questions, objectives, methodology and contribution of the study.

Chapter 2 presents a literature review of the related work.

Chapter 3 describes the research approach which is based on a case study and interpretive methods, such as observation and semi-structured interviews, which were used in the research.

Chapter 4 presents the results, as well as providing an evaluation of the design of the study.

Chapter 5 highlights the successes and failures of the study and presents guidelines for future research.

CHAPTER 2:
LITERATURE REVIEW

2.1 OVERVIEW

The previous chapter sketched the background to the research. This chapter (Chapter 2) focuses on recent literature related to the research. The literature review will be done in terms of the key concepts defined, as well as the research questions proposed in Chapter 1. This chapter explores the emergence of DevOps, restructuring required for DevOps implementation as well as challenges involved and mitigation strategies and the effectiveness of DevOps tools.

2.2 UNDERSTANDING THE EMERGENCE OF DEVOPS

DevOps is a movement that emerged from the Lean and Agile business practices (Wurster *et al.*, 2013). In 2009, senior IT consultant, Patrick Debois, organised the first DevOps meeting in Belgium with the aim to formalise this practice, and to create a community of practice which would enable the sharing of information about the use of DevOps (Humble & Molesky, 2011).

This community of practice was not really active until Jez Humble popularised the DevOps movement within the practitioner community in the following few years (Sharma, 2017). This saw many smaller companies adopting DevOps. As the philosophy gained momentum, bigger companies like Facebook and Yahoo implemented it (Wettinger, Breitenbücher & Leymann, 2014). Since then, DevOps has gained more traction (Caprarelli *et al.*, 2019).

As more companies have begun to show interest in DevOps, it is imperative that there is a common understanding of DevOps practices among DevOps practitioners as this can help in its successful adoption (Erich *et al.*, 2017; Crowley *et al.*, 2018). At the present moment, according to Gall and Pigni (2021), there is still a general lack of agreement regarding what exactly DevOps is. Consequently, practitioners are often confused in terms of how DevOps should be implemented (Lwakatare, Kuvaja & Oivo, 2016; Jabbari *et al.*, 2018; Senapathi, Buchan & Osman, 2018; Gall & Pigni, 2021).

The literature, according to several researchers, also does not provide enough information about how the restructuring should be done when adopting DevOps (Rütz, 2019; Smeds, Nybom & Porres, 2015).

Many IT organisations that have adopted Agile practices, now develop software at a faster pace to satisfy the constantly changing functional and non-functional software needs of their customers timeously (Sharma, 2017). However, as the operations teams are often not able to keep up with this pace, they cannot deploy the software to production as soon as it has been developed (Hemon *et al.*, 2020). Consequently, customer expectations are not met (Bucena & Kirikova, 2017).

The major cause of this delay is the lack of alignment between the development and operations teams. Developers traditionally focus on upgrading existing software and adding new application features, while operations personnel specialise in ensuring that the software is deployed and running smoothly in production (Kuusinen *et al.*, 2018; Leite *et al.*, 2019). The developers are assessed on their degree of creativity in terms of software development, while the operators are assessed on their ability to keep servers up and running and the applications responsive in production (Claps *et al.*, 2015).

This silo approach means that the development and the operations departments work as isolated entities with their own set of goals, procedures and skills (Leite *et al.*, 2019; Leite, Pinto, Kon & Meirelles, 2021). This situation is worsened by the fact that in some organisations the development and operations personnel are not collocated, and their interaction is thus extremely limited (Diel, Marczak & Cruzes, 2016; Sharma, 2017). The separation of closely interlinked tasks between the development and operations departments often leads to conflict among software development stakeholders (Ravichandran, Taylor & Waterhouse, 2016) (see Figure 2.1).

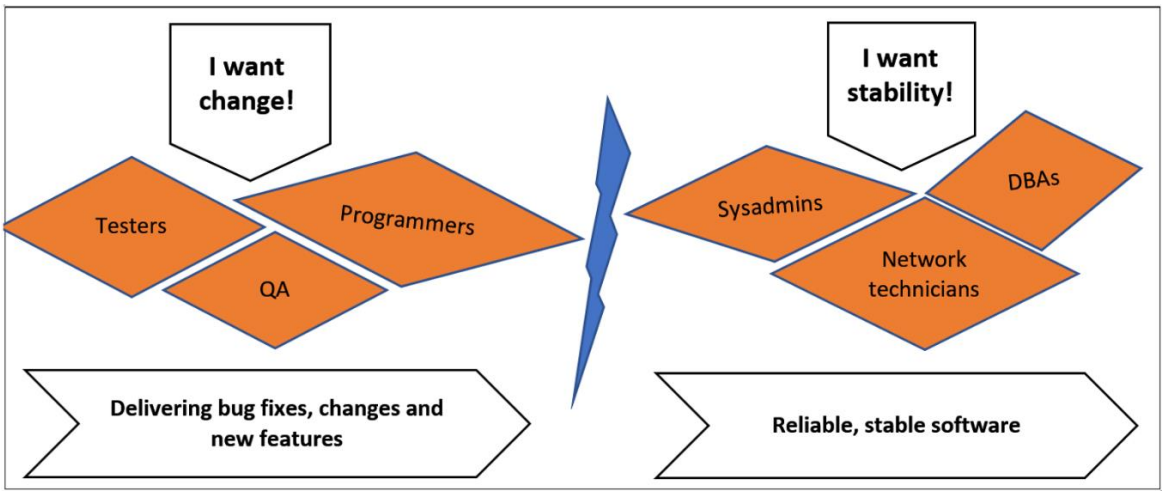


Figure 2.1: Misalignment leading to conflict

Source: Adapted from Hüttermann, 2012: 20

DevOps emerged as an alternative method to solve the friction between development and operations staff by broadening Agile principles and reinventing the entire software development process (Perera *et al.*, 2017; Wurster *et al.*, 2013). The development and operations silos can be eliminated by ensuring that both operations and development personnel become cross-skilled through working together and using the same tools, processes and goals, according to Jabari *et al.* (2018).

To help increase software production and its timeous delivery to customers, DevOps emphasises the need to keep manual activities to a minimum by automating as many software tasks as possible and ensuring that there is effective communication among software development stakeholders (Yarlagadda, 2021). It is important for the structure of the organisation to be transformed to allow for the effective implementation of DevOps, as discussed in the section below.

2.3 RESTRUCTURING REQUIRED FOR DEVOPS IMPLEMENTATION

The restructuring of an organisation for the implementation of DevOps involves several aspects, such as a change in culture, as well as the automation of certain processes to speed up the deployment of software (Teixeira, Pereira, Henriques, Silva & Faustino, 2020). Despite the availability of some information regarding DevOps, it is still a difficult task for many practitioners to grasp DevOps in a way that enables them to effectively restructure their organisations during the implementation of DevOps (Luz, Pinto & Bonifácio, 2019). There is no specific way of adopting DevOps in all

organisations; each organisation can implement DevOps practices differently to address their unique business challenges (Smeds *et al.*, 2015).

2.3.1 **Culture**

Culture as an enabler for change, is a collection of recurring behaviour, dormant emotions, firmly established habits, a commonly accepted frame of mind, and an interpretation of the world (Aguirre, Von Pos & Alpern, 2013). Culture develops in an organisation with or without a controlled pattern, and it augments and enlivens human behaviour (Katzenbach, Oelschlegel & Thomas, 2016). A change in culture often requires the identification of critical behaviours that need to be changed, and the empowerment of lower organisational structures to enable them to participate in the cultural transformation (Casagni *et al.*, 2018)

As mentioned before, when implementing DevOps, development and operations personnel need to work closely together on a daily basis until their roles converge and become similar. This means that the development staff members are ultimately able to perform operations tasks, such as software deployment and system stabilisation in production, while the operations team become skilled in software development and associated activities (Erich *et al.*, 2017). Job rotation among the developers and operators is an effective way of achieving a cross-functional team that can perform tasks across different departments (De França, Jeronimo & Travassos, 2016).

An organisation needs to support this teamwork by recognising both operations and development departments as one department by giving them the same incentives and goals (Hüttermann, 2012). It is important that all members of the team have the freedom to make decisions and are empowered to share their thoughts and ideas without fear of being victimised (Crowley *et al.*, 2018; Van Belzen, DeKruiff & Trienekens, 2019). Figure 2.2 shows the culture characteristics that DevOps teams must adopt to successfully implement DevOps.

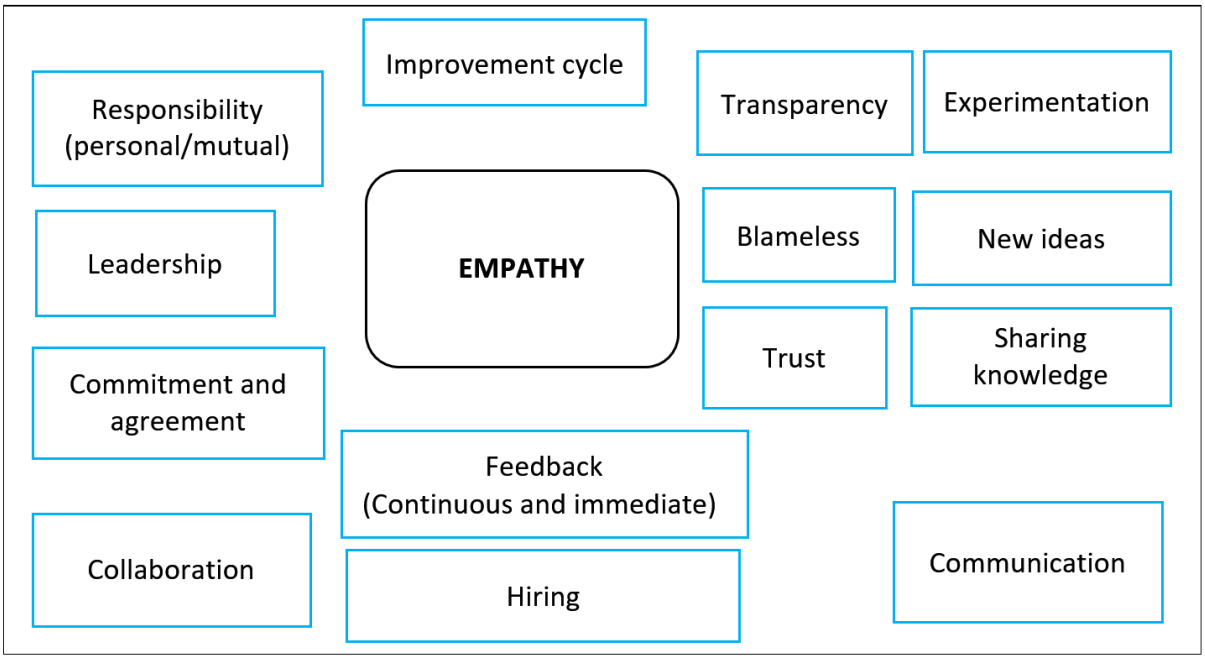


Figure 2.2: Characterising DevOps culture

Source: Adapted from Sánchez-Gordón & Colomo-Palacios, 2018:9

- **Blameless** means that employees are not victimized or punished for making mistakes so that they feel free to give accurate information of how the mistake happened (Davis & Daniels, 2016). By focusing on the situation that causes the error, rather than the person that causes the error, an organisation can improve the quality of its systems. Individuals that make mistakes are equipped with necessary knowledge and support to avoid similar mistakes in the future (Feijter,Vliet,Jagroep,Overbeek & Brinkkemper, 2017).
- **Improvement cycle** is the continuous improvement of the entire application system to ensure it is always at its best level of functionality. It also means that employees should keep on assessing their skills and improving them so that they remain very productive (Leite *et al.*, 2019). However, Fitzgerald and Stol (2017) argued that continuous improvement's ability to add customer value is limited because most of the time it is triggered by a problem that may have happened. Therefore, it is better to mainly focus on a proactive strategy like continuous innovation as it can bring new ideas that pioneer product improvement.

- **Empathy** is the critical requirement for DevOps culture. Team members need to understand each other's point of view and feelings so that they build work relationships that strengthens their sense of belonging. The ability to empathise with others unifies the team (Sánchez-Gordón & Colomo-Palacios, 2018).
- **Collaboration** requires that both the development and operation personnel work together to achieve cross functional teams. This means that operators get software development skills while the developers get experience in operations tasks. This increases the team's productivity (Luz, Pinto & Bonifácio, 2018).
- **Trust** is earned by being honest. For employees to believe in each other they need to be truthful in their communication and have a zero tolerance to lies and deception. Collaboration is achieved when employees trust each other (Khan, Jumani & Farhan, 2020).
- **Experimentation** gives employees an opportunity to try new activities and ideas. The culture of experimentation within DevOps space can result in the improvement of existing products and the introduction of new products. It is the source of new ideas (Sharma, 2017).
- **Sharing knowledge** is achieved through collaboration as employees learn new skills and gain knowledge of each other's tasks. An organisation can offer training programs to share information that improves employees' understanding of tasks (Crowley *et al.*, 2018).
- **Responsibility** is the duty of every team member to ascertain that the quality of the final product is of acceptable standard. DevOps culture encourages shared responsibility to minimise blame game and encourage cross skilling (Rowse & Cohen, 2021).
- **Feedback** that is obtained from customers enables the organisation to understand the quality of their product and apply improvements if need be. Feedback regarding employee performance gives an organisation information about the strength and weaknesses of their employees and this makes it easier to provide them with effective training and mentoring (Muñoz & Rodríguez, 2021).
- **Communication** is the exchange of work-related information that happens during collaboration to ensure that all software development stakeholders are

aware of the state of their applications and organisation in general (Rowse & Cohen, 2021).

- **Hiring** involves identifying the job titles that are required during the period when an organisation restructures for DevOps and recruiting candidates who are qualified and experienced for the roles (Leite *et al.*, 2019).
- **Leadership** is about taking key role to enable DevOps cultural transformation. Leaders should encourage and motivate employees to be creative and to embrace change that stimulates growth rather than maintaining the status quo. Leaders of the organisation are expected to subscribe to ethical values like integrity, fairness and empathy (Maroukian & Gulliver, 2020).
- **Transparency** refers to being clear in work related communication in order to avoid confusion (Yarlagadda, 2021).

The integration of the development and operations departments helps to speed up the development of software and its delivery to the customer, as well as facilitating the fast feedback loops required for continuous product improvement (Gall & Pigni, 2021; López-Fernández, Diaz, Garcia-Martin, Pérez & Gonzalez-Prieto, 2021). The collaboration experienced during the process of combining developer and operator tasks is often a source of employee motivation and improved job satisfaction (Erich *et al.*, 2017).

Table 2.1 presents a summary of some of the initiatives to facilitate developers and operators' cooperation, and the associated disadvantages.

Table 2.1: DevOps adoption approaches

Collaborating departments	Cross-functional team	DevOps team
Development and operations departments collaborate closely. It implies overlapping of developers and operators responsibilities. ***Downside:*** New responsibilities can be unclear for employees.	The product team is responsible for deploying and operating (You built it, you run it). Recommended by Amazon and Facebook. ***Downside*:** Requires more skilled engineers.	Acts as a bridge between developers and operators. It is better accepted when it is a temporary strategy for cultural transformation. ***Downside*:** Risk of creating a third silo.

Source: Leite *et al.*, 2019:27

2.3.2 **Automation**

When implementing DevOps, it is important to automate the entire software system, which includes the deployment pipeline, testing, monitoring and measurement (Erich *et al.*, 2017). Automation aims to minimise human involvement. Manual processing is often slow and error-prone, whereas automated processes are fast, reliable and repeatable (Gill, Loumish, Riyat & Han, 2018). With automation, high-quality software can be delivered to the customers, which demands less effort from the DevOps team. This allows more time to focus on reinventing and improving software systems (Mishra & Otaiwi, 2020). Automated infrastructure improves collaboration, unlike manual systems where each person mostly works individually (Luz et al., 2018). For automation to be a success, it requires good management supported by proper funding (De França *et al.*, 2016), as discussed below.

2.3.2.1 Automated deployment pipeline

An automated deployment pipeline is used for the continuous fast and efficient movement of high-quality software from the developers' computers to cloud-based test or production environments (Lwakatare *et al.*, 2019). Once its deployment pipeline has been implemented, an organisation can deploy software to production many times a day (Gall & Pigni, 2021). Figure 2.3 below shows the structure of a deployment pipeline.

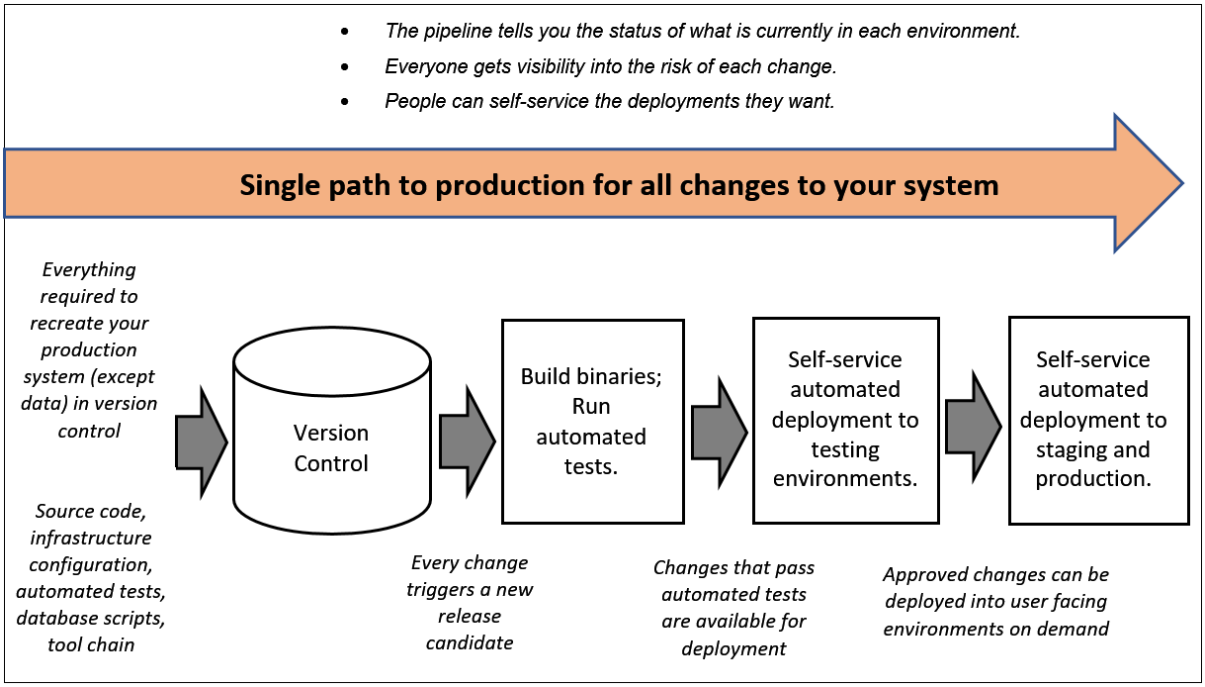


Figure 2.3: The deployment pipeline

Source: Adapted from Humble & Molesky, 2011:8

The deployment pipeline can be achieved using the Microsoft Azure DevOps tool that has a continuous integration functionality to ensure that software code from different developers is built together into an application prior to being tested and deployed (Khan et al., 2020). There are other tools that can be used to construct a deployment pipeline, such as Jenkins, however, the choice depends on the organisation's requirements (Rütz, 2019). Jenkins is a continuous integration tool that is used to automate the building and testing of software applications (Mohammad, 2016).

Before implementing a deployment pipeline, it is important to understand the functionality of an organisation's current system, as this helps to make better decisions regarding which automating tools to use (Crowley *et al.*, 2018). The implementation of a deployment pipeline must be a gradual process that is based on the concept of continuous improvement, where the DevOps teams strategically modify and enhance the system process in accordance with business requirements (Leite *et al.*, 2019).

A deployment pipeline can be customised based on the organisation's requirements, but all software changes made to the system must pass through each stage of the deployment pipeline chronologically (Humble & Farley, 2010). The stages of the deployment pipeline are shown in Figure 2.4, and discussed below.

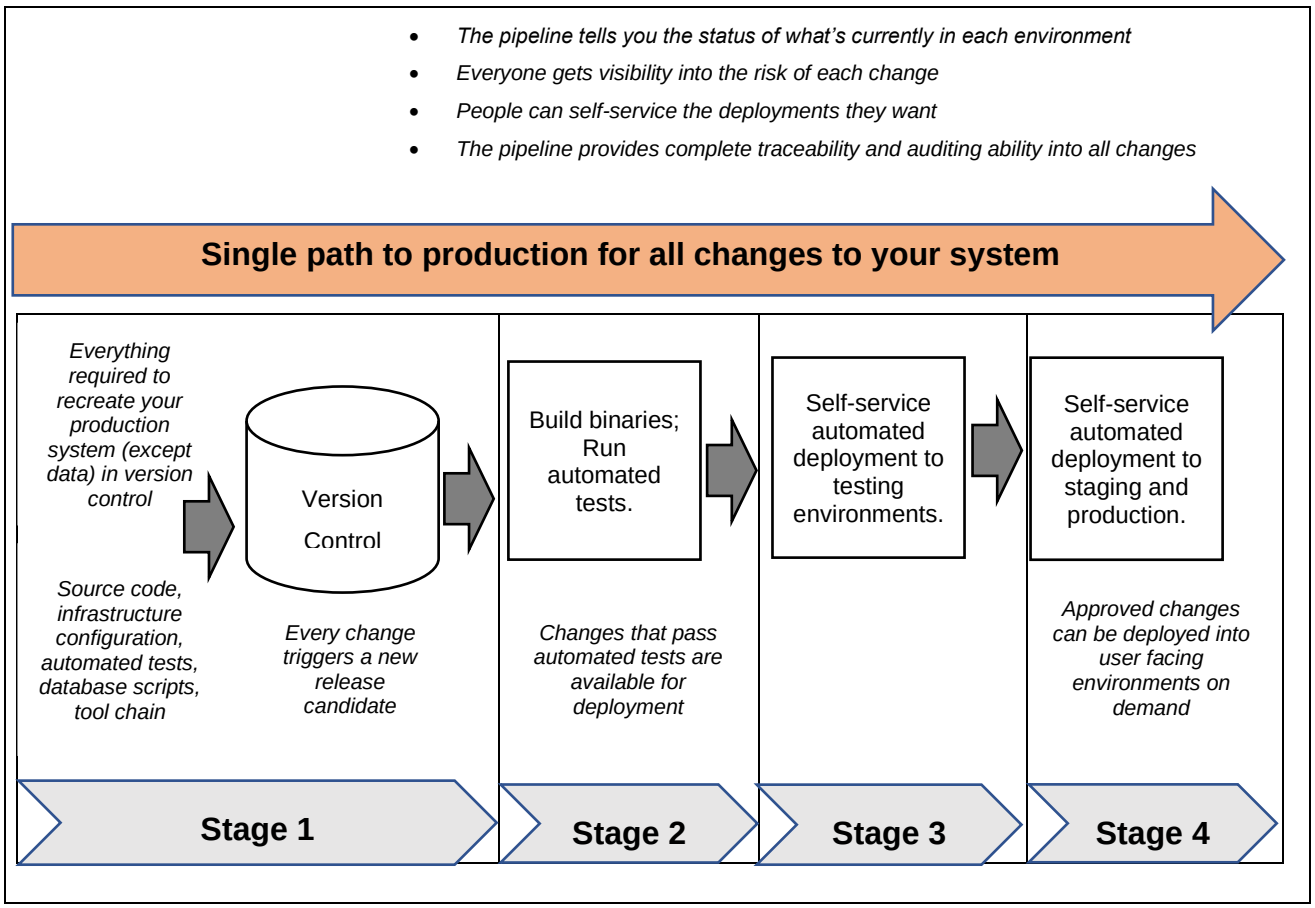


Figure 2.4: Stages of the deployment pipeline

Stages of the deployment pipeline

Stage 1: This is the first stage of the deployment pipeline. Continuous software integration takes place during this stage. It is a process whereby different developers working together on a project continuously combine their software code and save it to the common repository in different versions (Sharma, 2017). Developers normally use Git as a tool to release their source code into a shared repository so that it will undergo unit tests to help identify and eliminate integration-related errors (Mohammad, 2016).

Version control is part of this stage, and it is a process that involves the grouping of software code changes using incremental logic so that it becomes easy to track historical changes to the system (Karamitsos, Albarhami & Apostolopoulos, 2020). It is also important to keep software builds in versions because if a bug finds its way to production, an application can be restored to its earlier version to eliminate the bug (Claps *et al.*, 2015).

During Stage 1, it is advised to develop software components that are loosely coupled with independent functionalities that are linked together, termed as microservices

(Ghantous & Gill, 2017; Chen, 2018). Microservice architectures enable agile and scalable software systems, and this makes it faster to complete software testing and to deploy it to production because these systems are made up of smaller software units that can be developed, tested and deployed on their own (Shahin & Babar, 2020). Therefore, some legacy systems that are monolithic will need to be changed to microservices to enable the use of a deployment pipeline (Smeds *et al.*, 2015). Some organisations prefer to develop new systems using microservices, while the legacy systems remain operational in their original state (Jones, Noppen & Lettice, 2016).

Microservice architectures have varying components of technology that are used to achieve a range of non-functional tasks, and that need to be strictly deployed with same version approved during testing, therefore, it is advised to use the minimum of viable components to reduce complexity (Leite *et al.*, 2019). Changing an organisation's system from monolithic to microservices can take a long time and requires people with technical expertise who are willing to learn and solve challenging tasks (Senapathi *et al.*, 2018).

Stage 2: A deployable component of the software that was produced in Stage 1, is released in Stage 2, as shown in the diagram. However, for the software artefact to be ready for deployment, it must undergo a process of automated testing. Test automation is a critical component of the deployment pipeline (Luz *et al.*, 2019). The DevOps team must work together to come up with test cases that address both functional and non-functional system requirements (Bass, 2017).

Every change to the software must be tested to ensure that the expected quality in terms of security and performance is maintained (Teixeira *et al.*, 2020). Besides enforcing software code best practices, automated tests also provide end-user satisfaction by identifying system defects, so that they can be eliminated before production deployment (Hemon *et al.*, 2020). Selenium is one of the tools that can be used to achieve automated testing (Bucena & Kirikova, 2017). This tool can simulate the activities that end-users perform on the browser when connecting to the application (Hüttermann, 2012). The benefit of automating software testing is that it often becomes simple and less expensive to test, and errors can be detected and resolved quickly (Ravichandran *et al.*, 2016).

Stage 3: Once the software passes the automated tests conducted in Stage 2, it is automatically moved to this stage for test environment deployment. In the test environment, the testers work together with all quality assurance stakeholders to assess the application's ability to address business needs (Hüttermann, 2012). In this case, user acceptance tests are done on the software application to ascertain its ability to respond to end-user requirements, security tests are carried out to ensure that the application is free from intruders, and compliance tests are performed to enforce software coding standards that subscribe to recommended international practices (Sharma, 2017).

An important consideration when building a test server is that it should mirror a production server in terms of configuration and compute nodes to avoid scenarios of having an application that functions well in the test environment and then fails in production (Kuusinen *et al.*, 2018). As Figure 2.4 shows, once all the stakeholders are satisfied with the application's functionality in the test environment, the software code is automatically moved to the next stage.

Stage 4: This is the stage of production deployment using the software that has been approved in Stage 3 to make sure that end-users have access to use the new software's features. Both continuous delivery and continuous deployment take place in this last stage. Continuous deployment differs from continuous delivery in the sense that the process of releasing software to the customers is fully automated without any manual processes, while continuous delivery backs up automatic software delivery to production with human involvement to ascertain the degree of production readiness (Rowse & Cohen, 2021).

Despite the automated deployment pipeline's ability to deliver software to the customer at the expected level of quality, some organisations manually check the products for conformity because they doubt its ability to do the job. Therefore, it is important to have an experienced DevOps team that can manage the automatic deployments properly to minimise manual intervention (Rütz, 2019). It is imperative to have a software functionality that automatically checks the quality of all end-products prior to deployment to ensure that it adheres to expected standards (Luz *et al.*, 2018).

The deployment of software builds is often done in smaller software batches, which reduces the risk of deployment failure, as the deployment challenges that can be

encountered and the solutions required are relatively small (Caprarelli *et al.*, 2019). The advantages of automatic software deployments are that, besides being quick and reliable, they also help with continuous product improvement, since end-users provide valuable feedback regarding new software features (Riungu-Kalliosaari *et al.*, 2016).

There is need for all software development stakeholders to be involved in the deployment pipeline, since decisions made during the implementation phase will ascertain a higher degree of operational effectiveness (Humble & Molesky, 2011). The DevOps team's level of motivation and job security are often increased using the deployment pipeline because they are confident about efficient software releases to production, even if such releases are massive (Riungu-Kalliosaari *et al.*, 2016).

Organisations that integrate their software applications with other software from service providers may have to upgrade their software integrations, as they may become dysfunctional during the building of the deployment pipeline (Claps *et al.*, 2015).

It is imperative to ensure that a DevOps culture of collaboration between the development and operations teams is achieved before implementing a deployment pipeline because organisational silos can prevent the successful operation of the deployment pipeline (Kuusinen *et al.*, 2018).

Some projects produce products that do not need to reach the market with speed, therefore, with such projects it is better to stick to manual deployments because an automated deployment pipeline will unnecessarily increase operational costs (López-Fernández *et al.*, 2021).

2.3.2.2 Automated system monitoring

Automated system monitoring is a post software deployment activity (Rowse & Cohen, 2021). It involves putting tools in environments, such as production, and testing to observe and record the day-to-day functionality of applications (Lwakatare *et al.*, 2016). Thus, to ensure that the software is delivered to the customer within agreed times and with the expected functionality (Gall & Pigni, 2021). If abnormal incidents occur within the system, then automatic notifications are sent to the DevOps teams so that they can take the necessary actions (Yarlagadda, 2021). Nagios is an example of a tool that can be used to automate monitoring (Ghantous & Gill, 2017). This monitoring tool can be used to provide complete monitoring of all application logs in

different operating systems, such as Windows and Linux, and can alert users if any application failure is detected (Nagios, n.d.).

Dashboards are important tools that are used to present monitoring results on users' computers (Humble & Farley, 2010). Monitoring enables the DevOps team to obtain feedback about the software system's performance, and this helps the team to make informed decisions regarding product improvements (Teixeira *et al.*, 2020; Akshaya, Vidya & Veena, 2015).

System monitoring helps an organisation to understand the software application's resource utilisation demands, and therefore, makes it possible to implement the appropriate random access memory (RAM) and computer processing units (CPU) required for servers to maintain application stability (Karamitsos *et al.*, 2020). All DevOps team members are responsible for monitoring; this helps to promote a sense of shared responsibility and teamwork which improves agility (Luz *et al.*, 2018). In addition, historical monitoring logs can be analysed to obtain information that can enable the developers to predict the chances of success of upcoming software releases (Capizzi *et al.*, 2019). Choosing and setting up suitable monitoring tools often require expert knowledge to avoid the risk of missing important functionality (Brunnert *et al.*, 2015).

2.3.2.3 Automated system measurement

System measuring should focus on carefully selected key business areas, and not people, because human beings can change their ways of doing things in relation to measurement objectives (Teixeira *et al.*, 2020).

Measurement is a process of obtaining system data that shows how the system functions in real time using metrics (Lwakatare, Kuvaja & Oivo, 2015). There are three types of metrics: process metrics, technology metrics and service metrics (Cartlidge *et al.*, 2007), as explained below.

- **Process metrics** focuses on system performance, for example, it can tell the time it takes to develop and deploy a software feature, the number of new software products or features that pass through a deployment pipeline, and what is released to production at specific time intervals (Lehtonen, Suonsyrja, Kilamo & Mikkonen, 2015).

- **Technology metrics** measures key system performance areas in terms of productivity and availability, hence, it helps employees to understand their system better and apply appropriate measures to improve performance (Crowley *et al.*, 2018).
- **Service metrics** provides results of product delivery to the end-users, such as customer experience metrics, which is used to obtain the degree of customer satisfaction regarding the organisation's service provision with the objective of addressing customer concerns, if any (DeMartine, Oehrlich & Doerr, 2015). DevOps team members need to have basic mathematics skills to be able to use metrics and interpret the results accurately (De França *et al.*, 2016).

Other measuring tools for DevOps are maturity models. These can be used to find the extent of DevOps transformation within an organisation, and as such, this could help to decide on areas that need to be improved (Leite *et al.*, 2019). Table 2.2 shows how some organisations measure and categorise their software delivery performance using the DevOps maturity model. These include deployment frequency, lead time for changes, and mean time to recovery.

Deployment frequency is the number of times an organisation can deploy their software at a given time (Senapathi *et al.*, 2018). As Table 2.2 shows, the deployment frequency can be regarded as elite, high, medium or low, depending on the number of times that an organisation deploys software per day.

The term 'lead time' for changes refers to the time required between software changes to be approved, and for the changes to be successfully delivered to the customer (Humble & Molesky, 2011).

Mean time to recovery is the time it takes to normalise the system after it has been disrupted (Banica, Radulesca, Rosca & Hagiu, 2017).

Table 2.2: Software delivery performance indicators

Software delivery performance indicators	Elite	High	Medium	Low
Deployment frequency	On demand, multiple deployments per day	One deployment per day	One deployment per week	Between once per week and one per month
Lead time for changes	Less than one hour	Less than one hour	Between one hour and one day	Between one hour and one day
Mean time to recovery	Less than one hour	Less than one hour	Between one hour and one day	Between one day and one week

Source: Díaz *et al.*, 2019:4

DevOps maturity models, furthermore, could assist organisations to identify and adopt appropriate cultural and technological changes to enhance their DevOps practices (Bucena & Kirikova, 2017).

2.3.2.4 Recovery automation

Recovery automation involves equipping the software system with configurations that enable it to restore its normal functioning state after a failure (Luz *et al.*, 2019). Nexus is one of the tools that can be used to recover from software failure, as it stores the deployable components of applications (Leite *et al.*, 2019). In organisations where there are ongoing database changes or a sophisticated deployment solution, system restore can be a challenging task, and, in most cases, requires the DevOps team to rectify the cause of the problem rather than to perform a rollback (Rütz, 2019).

2.4 CHALLENGES ENCOUNTERED DURING RESTRUCTURING

This section outlines the challenges that may be encountered during the process of implementing the restructuring approaches required for DevOps.

2.4.1 Cultural challenges

These are the challenges that are caused by the DevOps team and that make it difficult to implement DevOps.

The alliance between the development and operations teams may be jeopardised by the operations or development staff's unwillingness to do extra work, especially if they perceive DevOps as a way for the organisation to overuse them and undermine their competence (Smeds *et al.*, 2015). In some organisations, the developers or operators may already feel they have too much work, and they cannot afford an extra responsibility (Bucena & Kirikova, 2017). In other situations, the operations personnel could feel that development work is not part of their contract agreement (Jones *et al.*, 2016). Developers may dislike doing an operator's job, since it is not their area of expertise, and therefore, they may not support the decision to work together (Erich *et al.*, 2017). Some senior employees who have been working for a long time may resist any form of change, as they prefer the old and trusted ways of doing things (Khan *et al.*, 2020).

Achieving a DevOps team may not be possible in some organisations, where there are laws and contract obligations that do not allow developers to access production environments, or that do not allow the operations staff to access the development environments. Therefore, it is important to review these impediments first (Riungu-Kalliosaari *et al.*, 2016). In some organisations, where teams are not collocated, communication is mainly electronic and is often delayed due to different standard times. As a result, such teams find it difficult to work at the same time, and this may negatively affect collaboration and productivity (Smeds *et al.*, 2015). To minimise this problem, the teams must have face-to-face meetings as often as possible to enable team members to discuss work-related issues while getting familiar with each other (Diel *et al.*, 2016). It is often a complicated assignment to integrate development and operations teams, and the process can take many years to achieve (Rütz, 2019).

A high level of dedication is required from each member of the team to successfully achieve a DevOps culture (Gall & Pigni, 2021). Having both developers and operators report to the same manager can help to reduce work-related conflicts (Jones *et al.*, 2016). Luz *et al.* (2019) noted that the challenges that result from development and operations teamwork may be mitigated by communicating with employees to encourage a positive mindset, as well as investing in employee training courses to help improve their skills. Rütz (2019) added that a good management team is essential for the successful implementation of DevOps an organisation, especially when a process of cultural transformation is required. Khan *et al.* (2020) concluded that for the

cooperation between development and operations to reach its full potential, it is important that the top management of the organisation recognise it and give it their support.

2.4.2 Automation challenges

Automation is the use of tools or technology to accomplish tasks (Luz *et al.*, 2019), however, this may result in many different challenges which make it difficult to implement DevOps.

2.4.2.1 Automated deployment pipeline challenges

The following are some of the challenges that may result from the automated deployment pipeline:

- **Resistance to change:** Challenges involving stakeholders include managers who resist change, and who want to stick to their known old ways of doing things. Such managers can be big blockers to technological transformation (Leite *et al.*, 2019). Other stakeholders, for example, customers, may not like the idea of having software delivered to them on a daily basis, therefore, an agreement must be reached with them before the deployment pipeline is functional (Shahin, Babar, Zahedi & Zhu, 2017). Some customers may complain that they do not see the new features the organisations claim to deploy every day using the deployment pipeline, therefore, an effective communication strategy should be prioritised to avoid misunderstandings (Claps *et al.*, 2015). In some organisations, production deployment cannot be done until all stakeholders have approved it, and this can have negative effects to the frequency and speed of production deployment (López-Fernández *et al.*, 2021).
- **Choosing the right tools:** Choosing the appropriate tools for the deployment pipeline is not an easy process, and it determines whether DevOps adoption will be successful or not (Bucena & Kirikova, 2017).
- **Continuous integration:** Practising continuous integration within the deployment pipeline can become time consuming and difficult when the code changes from different developers fail to work together, and require team effort to be corrected (Laukkanen, Itkonen & Lassenius, 2017).

- **Resistance to use microservices:** Although organisations are encouraged to change their software architecture from monolithic to microservices to enable the deployment of pipeline functionality, many organisations may resist such a change of architecture if they do not have proof of its benefits (Smeds *et al.*, 2015). Some organisations may prefer to develop new systems using microservices, while ensuring their legacy systems remain operational in their original state. However, the developers often complain that the maintenance of legacy systems hinders them in the effective implementation of new systems (Jones *et al.*, 2016). Employees who are not yet familiar with microservices often find its features difficult to use, hence, there is need to simplify the adoption of microservices by using minimum viable requirements (Leite *et al.*, 2019). As the different components of software in microservices may not behave in the same way, it can be difficult to ensure that the application functions the expected way across the entire system (Chen, 2018).
- **Test environment deployment:** It is important to replicate the production environment in the test environment to ensure smooth deployments. However, the process of ensuring that production and test servers have the same configurations and features can be a big challenge because of the high costs involved, and the strict production access controls and restrictions that some organisations impose (Shahin *et al.*, 2017). Manual quality checks are still being used to determine software's readiness to be deployed to test or production servers. This is because, in most cases, an efficient automatic rollback functionality is not available if software with a bug is deployed (Shahin *et al.*, 2017).
- **Maintenance:** For a deployment pipeline to operate with the required degree of efficiency and effectiveness, it needs to be maintained by employees with a variety of related skills and experience. This is often not the case, as many organisations have a shortage of the required expertise (Senapathi *et al.*, 2018; Lwakatare *et al.*, 2019).
- **Different production environments:** In organisations with many production environments, the implementation of an automated deployment pipeline is often a complicated task because of the different configurations and access requirements of these environments (Smeds *et al.*, 2015).

- **Software bugs:** Having a fully functional automated deployment pipeline does not guarantee the fast deployment of software features to production. In some instances, deployment can be delayed due to software bugs that can be caused by software integration activities (Lehtonen *et al.*, 2015).
- **Compliance requirements:** In some cases, having an operational automated deployment pipeline can be a violation of government software regulation policies, hence, it is necessary to know the compliance requirements before implementing it (Shahin *et al.*, 2017). Lwakatare *et al.* (2019) added that achieving a fully functional deployment pipeline can be a challenging task because it involves a significant change to the way software is managed. For example, if the software improvements include making changes to the database, then implementing a deployment pipeline becomes extremely difficult.
- **Security:** Wilde *et al.* (2016) warned that many organisations may end up implementing a deployment pipeline that is susceptible to security attacks by hackers and malware because of their limited understanding of security configurations. Hence, there is need to consider using virtual machines, where possible, and consulting security experts who can help to deliver a deployment pipeline without security flaws.

2.4.2.2 Automated testing challenges

Automated testing does not always guarantee flawless software, as some new errors that are not catered for by test cases can remain undetected. Therefore, it is necessary to use a variety of testing methods to mitigate these issues (Caprarelli *et al.*, 2019). If an organisation has multiple environments that are not compatible with each other, automated testing often becomes less effective because of the different configurations (Riungu-Kalliosaari *et al.*, 2016).

Some organisations view automated testing as a non-viable option, as it involves high costs and low returns (Shahin *et al.*, 2017). Some organisations end up with deployment pipelines that do not have performance-testing techniques, as it is too expensive and time consuming to establish, while other organisations have inadequate test coverage of their systems, and therefore, software flaws find their way to production (Brunnert *et al.*, 2015).

Testing results are sometimes not clear, which makes it difficult to know the causes of success or failure of the tests, and as a result, the developers are often left confused about the next testing step (Laukkanen *et al.*, 2017).

2.4.2.3 Automated system monitoring challenges

It is not easy to identify the part of the application that needs to be monitored, and consequently, most monitoring activities are installed after software failure within a certain area of an application (Lwakatare *et al.*, 2019). In some scenarios, where organisations experience huge software releases per day, it can be a labour-intensive and time-consuming activity to analyse the monitoring logs to identify problems, and this needs to be done with a sense of urgency (Lwakatare *et al.*, 2015).

The size of monitoring logs can grow rapidly and utilise all the available storage space and memory, resulting in a need to upgrade the existing servers (Shahin & Babar, 2020). Unfortunately, there are still no monitoring tools that can forecast application defects before they cause downtime (Lwakatare *et al.*, 2016).

2.4.2.4 Automated system measurement challenges

There is no specific way to identify the units of the system that need to be measured, and there is no standard way to measure whether an organisation has successfully implemented DevOps or not (Luz *et al.*, 2019). System measurement information that helps organisations to identify the metrics that must be used is still very limited, and as a result, many organisations obtain metrics primarily based on availability rather than effectiveness (Ravichandran *et al.*, 2016).

Most of the tools that are used to measure system performance are not compatible with the tools used for the deployment pipeline, hence, they remain unused (Bezemer *et al.*, 2019). Furthermore, organisations need to be cautious when using metrics, as some of them can negatively affect employee morale (Leite *et al.*, 2019).

Table 2.3 shows the negative effects of metric classes, such as vanity metrics, which measures the size of code and quantity of features developed, traditional metrics that is used to measure the rate of occurrence of system failure (Kaur & Bahl, 2014), and intra-team metrics that is used to measure and compare different teams' effectiveness in terms of productivity failures (Ravichandran *et al.*, 2016).

Table 2.3: Problematic metric classes

Metric classes	Examples	Adverse effects
Vanity metrics	Lines of code produced	May be counterproductive, since it rewards the wrong types of behaviour, especially if incentives are linked to the metric.
	Function points created	Producing more code and features without validation can inhibit other valuable activities, such as refactoring and design simplification.
Intra-team metrics	Agile team leader boards	Beware of metrics that makes teams compete against each other and that uses vanity metrics as scoring mechanisms.
	Deployments/changes prevented	Strike a balance with metrics and rewards that influence positive inter-team behaviours, such as code sharing, peer reviews, and mentoring. Pay particular attention to metrics that promote an anti-DevOps culture, such as rating operational effectiveness on the ability to prevent releases and deployments.
Traditional metrics	Mean time between failure (MTBF)	With faster delivery of services, some failure is to be expected.
	Full time equivalent (FTEs): Servers	Always consider that improving responsiveness can be more important (and less costly) than trying to prevent failures.

Source: Ravichandran, Taylor & Waterhouse, 2016:42

2.5 DEVOPS TOOLS AND THEIR EFFECTIVENESS

It is unlikely that a single DevOps tool can be used to achieve the complete automation of an entire system, hence, there is a need to consider a variety of different tools (Gill *et al.*, 2018). However, integrating different DevOps tools to work together often proves to be extremely challenging (Joby, 2019). Most organisations customise DevOps tools to achieve their objectives, which mainly include minimising manual processes (Shahin *et al.*, 2017).

Some of the tools used to achieve DevOps goals are listed in Table 2.4.

Table 2.4: Tools grouped in DevOps toolchain groups

Toolchain groups	Related tools					
Version and source control	Git/GitHub/GiLab	Mercurial/ bitbucket	Subversion			
Containerisation	Docker		Rocker		Vagrant	
Configuration and management tools		Puppet	Ansible	Cheff		SaltStack
Continuous deployment	Capistrano	Jenkins		Codeship	Travis CI	Circle CI
Continuous integration and orchestration	Atlassian Bamboo		TeamCity			
Build automation	Apache Maven Proj		Apache Ant	Gradle		
Automate. Test and validate.	Cucumber	Selenium	TestComplete	Jmeter		
Monitoring	Zabbix	New Relica	Nagios	Splunk	AppDynamics	
Collaboration tools	Slack	Jira	HipChat	Pager Duty		
Issue tracking	Bugzilla		Track/tesTrack	MantisBT	Assembla	
Planning tools	Clarizen			Asana		
Knowledge sharing tools	Crowdbase	Nuclion	Confluence			
DB handling tools	DBMaestro	LiquiBase	RedGate			

Source: Bucena & Kirikova, 2017:12

It is a difficult process to set up the correct DevOps tools for an organisation, therefore, some organisations consult service providers to help them (Rütz, 2019). DevOps tools, such as Jira, facilitate the interaction between developers and operators, and thus help to achieve a collaborative culture that is a one of the main practices of DevOps (Erich, 2018).

As the ideas surrounding DevOps keep on changing, the tools used are also modified and new tools introduced. This demands more clarity on DevOps practices to avoid adoption-related confusion (Riungu-Kalliosaari *et al.*, 2016). The use of new tools often results in a complicated IT infrastructure, which can negatively affect employee work performance (Khan *et al.*, 2022).

2.6 SUMMARY

The chapter showed that implementing DevOps in an organisation is neither easy nor straightforward, it requires certain job commitments and financial sacrifices. The key practices that must guide organisations during the implementation of DevOps are the need for a collaborative culture and the automation of the entire system. However, culture is the key that enables technological change.

Based on the content of this chapter, it is interesting to note that despite the increasing popularity of DevOps, there is still a high level of uncertainty regarding not only the definition of DevOps but also how organisations should use it to improve their business processes. This means that organisations are likely to implement DevOps based on their understanding but then the way one organisation understands it may differ from the way another organisation understands it. Therefore, it is not surprising to see different DevOps implementations in different organisations. The researcher used probing questions in Chapter 4 to understand how an organisation was restructured to accommodate DevOps methodology. The next chapter discusses the research approach and methodology adopted for the current study.

CHAPTER 3:
RESEARCH APPROACH AND METHODOLOGY

3.1 OVERVIEW

The previous chapter reviewed the related literature. This chapter presents a discussion of the research methodology and research design. More specifically, the research epistemology, theoretical perspective, methodology and methods will be argued. Ethical considerations, the validation of the research, as well as its reliability will also be noted.

3.2 RESEARCH PHILOSOPHY

Crotty (1998) identified four elements that define the research process (see Figure 3.1). The process includes the following elements: epistemology, theoretical perspective, methodology and methods (Crotty, 1998).

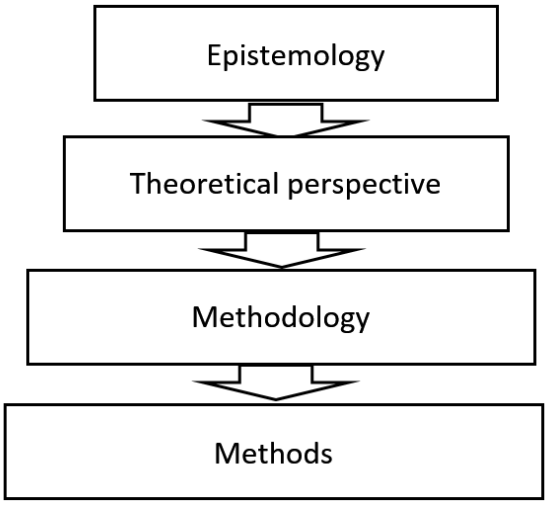


Figure 3.1: Four elements of social research

Source: Adapted from Crotty, 1998:4

3.2.1 Epistemology

Epistemology is the philosophical foundation that determines the type of research that is possible, and how the research can be developed as appropriate and valid (Crotty, 1998). There are three main epistemological paradigms: objectivism, constructionism and subjectivism. Objectivism assumes that research reveals the objective truth and

meaning, while constructionism is the construction of a person's perceived social reality (Crotty, 1998). Subjectivism, on the other hand, is defined as the doctrine that knowledge is merely subjective and that there is no external or objective truth.

The current research study assumed a qualitative epistemological stance, since the outcome of the field research was based on the contribution from the setting in which the study took place. This research study subscribes to both subjectivism and constructionism because the researcher engages the participants to get details of their perceptions regarding the way they changed their organisation for DevOps and to learn how they understand DevOps based on the way they implemented it respectively.

Theoretical perspective

A theoretical perspective is a set of philosophical and conceptual assumptions that the researcher subscribes to during the research, for the purpose of having a strategy that acts as a guideline during methodological execution (Crotty, 1998).

Interpretivism is the belief that the only way to understand the social process of a study is by accessing the area in which it occurs through engaging the people that are involved in it (Nandhakuma & Jones, 1997). The researcher's objective is to understand the social and cultural settings of the area under investigation, without compromising its natural existence (Klein & Myers, 2001).

An interpretive approach seeks to provide a deep insight into a given phenomenon based on the views and understanding of the people who experienced it (Andrade, 2009). The risk of interpretivism is that the researcher may incorrectly understand the respondents' expressions, or the respondents may be deceptive or unwilling to disclose all information, and this can lead to an outcome that does not correctly reflect the phenomenon (Nandhakuma & Jones, 1997). Interpretivism has also been criticised for not considering the historical issues or external factors that may have contributed to certain facts, and also not being critical of the status quo (Orlikowski & Baroudi, 1991). In order to mitigate these risks it is advised that a researcher must rigorously engage the participants over an extended interaction period, or if possible, arrange for a face-to-face conversation (Nandhakuma & Jones, 1997).

The interpretivist approach was deemed appropriate for the current research. This research is interpretive because the researcher engaged people who are actively

involved with DevOps in a specific organisation to obtain their views and understanding of how the implementation of DevOps transformed their organisation. The data that was obtained from the participants' different experiences and opinions was analysed to create the information required to understand the impact that DevOps has on IT organisations.

3.2.2 **Methodology**

A research methodology is the steps to follow during a research project to control the way the data is obtained and processed for the purpose of broadening knowledge (Leedy & Ormrod, 2005). A case study methodology was selected for the purpose of the current study.

A case study is a thorough investigation of a fact using real life scenarios to establish a detailed relationship between the phenomenon and its surroundings (Zainal, 2007). Case study research explores a phenomenon without changing its natural setup to achieve a deeper and more accurate comprehension of the events, based on the interpretations and meanings given by the participants (Crowe *et al.*, 2011).

A case study approach is suitable when the aim of the study is to answer 'why' and 'how' questions, and when there is need for richness of description within research questions (Rowley, 2002). According to Yin (2018), the case study approach can be used when there are no clear boundaries between the phenomenon and the context. A suitable data collection plan must be achieved with the guidance of a case study protocol. A single case study is when a researcher investigates a phenomenon in a single unit of analysis (Yin, 2018). The case study methodology follows a specific set of steps, as illustrated in Figure 3.2.

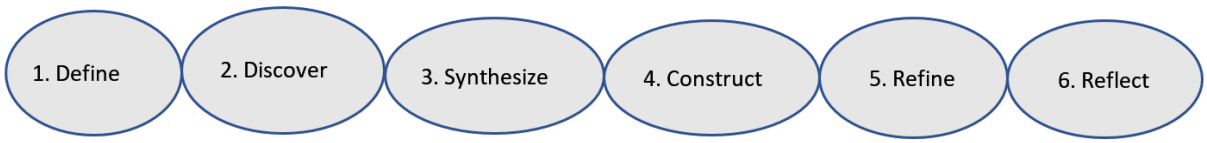


Figure 3.2: Knowledge opportunities in the design process

Source: Adapted from Zimmerman, Evenson & Forlizzi, 2004:9

3.2.3 **Methods**

Research methods refer to the procedures or techniques that are used to collect and analyse data for a specific study (Crotty, 1998). For interpretive research studies,

observation and interviews are usually the main methods for collecting data (Walsham, 1995). For the purpose of the current study, the researcher also used documentation as a secondary source of data.

3.2.3.1 Interviews

An interview is a conversation between a researcher and a participant regarding a point of investigation (Halkias & Neubert, 2020). There are different types of interviews (Graue, 2015), as listed below:

- Structured interviews: These interviews are guided strictly by specific closed-ended questions (Whiting, 2008). Structured interviews can be biased in terms of the responses from the interviewees or as a result of poorly designed questions by the researcher (Yin, 2018). This data collection method is most appropriate for quantitative research (Harrell & Bradley, 2009). Therefore, structured interviews were not part of the current study, since this research is qualitative.
- Semi-structured interviews: This occurs when the research data is collected using guiding questions or probing questions (Graue, 2015). Semi-structured interviews allow for the collection of detailed information about a phenomenon by allowing the interviewees the freedom to sometimes diverge from the question asked. The researcher can access the interviewees' interpretations, views and understanding concerning the area under investigation (Harrell & Bradley, 2009). During a semi-structured interview, a researcher uses predetermined questions but has the flexibility of adjusting a line of thought to get as much relevant information as possible (Whiting, 2008). However, there is a risk of researcher bias caused by subjectivity (Graue, 2015). The process of arranging the interviews and analysing data from the interviews is often time consuming and difficult (Adams, 2015). Semi-structured interviews were the primary data collection method for this study.
- Unstructured interviews: This is when a researcher wants to understand a certain topic by asking open-ended questions, mostly to experienced and knowledgeable interviewees (Whiting, 2008). The researcher can ask questions in any order and the respondent is free to guide the conversation in any direction if the information provided is relevant to the topic (Graue, 2015). The researcher conducted unstructured interviews with field experts to collect more information to guide the selection of the best-case company for the single case study.

3.2.3.2 Observation

When a researcher has the opportunity to observe activities of the organisation in relation to the study it is called observation (Baker, 2006). It needs to be noted that when participants realise that they are being observed, they may change their behaviour (Yin, 2018). The researcher interviewed the respondents online in face-to-face interviews, as visiting the organisation was not possible during the Covid-19 pandemic, thus the respondents could not be observed during the interviews. Observation was thus not a data collection method in the current study.

3.2.3.3 Documentation

Both soft and hard copies relevant to the organisation under investigation can be read to achieve a better understanding of the organisation (Halkias & Neubert, 2020). However, when using Internet searches to find documents, it is important to make sure that these sources are reliable (Yin, 2018).

Online organisational policy documents and reports found on the companies' websites were used by the current study to determine the most suitable organisation for the study. The researcher used the documentation of the organisation targeted for this single case study as a secondary data source.

3.3 DESIGN OF THIS RESEARCH

This section outlines the detailed research design that applies to the current study.

Figure 3.3 shows the steps (phases) of the case study approach, with the duration of each phase and the type of research methods used within each phase.

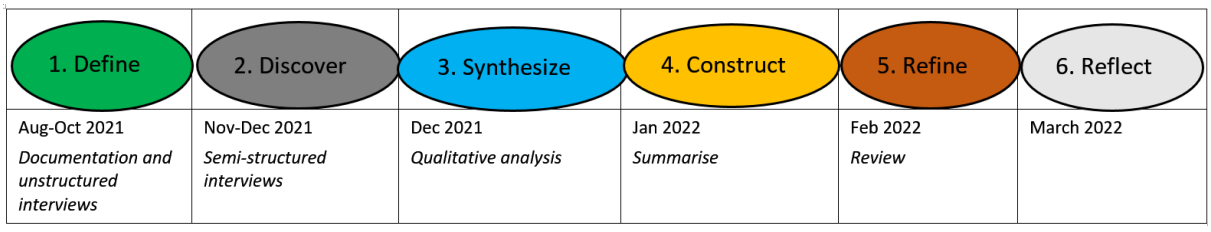


Figure 3.3: The case study design process and dates executed

Each of these phases is briefly discussed below.

3.3.1 **Define**

This is the first stage of this case study methodology. It took three months, from August to October 2021, to collect the data. It involved the identification of an appropriate company for a single case study. Using Google and the query “DevOps companies in South Africa” approximately 20 companies were identified.

To screen the companies, online documents found on the companies’ websites were studied. From these documents, two DevOps specialists were identified, contacted, and interviewed using an unstructured interview method. The specialists were asked their opinion about the maturity of the DevOps implementation in the identified companies, and which of these companies they would suggest would be the best to use as a single case study. It enabled the researcher to identify the most suitable organisation to investigate.

3.3.2 **Discover**

This is the second stage of the case study methodology and refers to the development of a data collection plan. The DevOps specialists assisted the researcher to contact some DevOps role players at the identified organisation. These role players were selected based on their role in the organisation and their years of experience. Eight employees were contacted via e-mail to request an interview. The emails to the respondents included the ethics approval certificate of the current study (Appendix A), a list of interview questions, and a letter requesting permission for the research to be conducted at the company (Appendix B). The interviewees were informed that their participation is voluntarily.

Four senior employees indicated that they would be prepared to participate (see Table 3.1). It was difficult to find an appropriate time for the interviews, and some of the interviewees rescheduled several times because of work commitments whilst others did not respond to the emails for interview request. Three of the interviews were recorded, and one interviewee preferred not to be recorded but allowed the researcher to take notes on what was said during the interview.

The first two interviews were more productive because most of the new ideas were raised. The third interview had fewer new suggestions as most of the points had already been raised during the first two interviews. The outcome of the final and fourth interview was mostly a repetition of the issues that were raised during the first three

interviews. This convinced the researcher that the four interviews were enough to obtain sufficient important information for the research.

The following probing questions guided the semi-structured interviews:

- How did you restructure your organisation during DevOps implementation?
- Are there any other changes related to DevOps transformation that occurred?
- What challenges did you encounter during DevOps transformation?
- What can you say are the possible solutions to the challenges?
- To what extent did the DevOps tools that you used help to implement DevOps?

Table 3.1 lists the roles, gender and experience of the respondents who participated in the interviews.

Table 3.1: Respondents' roles and work experience

Role	Gender	Years of experience in IT	Interview duration	Unique Identifier
Executive head	Male	More than ten years	50 minutes	Respondent 2
Senior DevOps Consultant	Male	More than ten years	1 hour 20 minutes	Respondent 1
Organisational Design Consultant	Female	More than ten years	55 minutes	Respondent 4
Head of Agile transformation	Female	More than twenty years	1 hour	Respondent 3

As shown in Table 3.1, the participants held different positions within the organisation, which included an executive head, a senior DevOps consultant, an organisational design consultant and the head of Agile transformation and included both genders. These participants had extensive experience that helped the researcher to broaden the understanding of the impact DevOps has had on the organisation. These employees are key DevOps decision-makers in their departments, and each of them has unique DevOps experience because of the different organisational assignments they focus on.

An important principle of data collection in single case studies is the collection of evidence from different sources with the objective of verifying the same finding. This is known as triangulation (Baxter & Jack, 2008). This was achieved by getting different perspectives and interpretations from the different respondents who were interviewed. During the interviews, the researcher asked the questions in the order they are presented above and allowed the respondents to express themselves freely as relevant to the questions. The researcher had the freedom to ask more probing questions to get more detail (Adams, 2015), however, the researcher made sure not to influence the opinions of the interviewees with these further probes. A pre-determined idea of what the outcome should be can result in a biased opinion and biased questions (Rowley, 2002).

3.3.3 **Synthesise**

This is the third stage, and it involved the researcher listening to the recordings and summarising the ideas of each interviewee according to the questions asked and noting it down in a notebook. This process entailed considering similar and opposing ideas that were highlighted in the notes. This initial analysis resulted in the identification of specific phrases of text that had a direct bearing on the questions asked.

The manual analysis was followed by an analysis using the NVivo[1] qualitative analysis software. The qualitative data analysed by NVivo can be in the form of text, videos, images or online content (Castleberry & Nolen, 2018).

A thematic approach was used to extract information from the data by creating specific codes. Thematic analysis is a process of scrutinising and identifying data patterns within the phenomenon under study. It helps the researcher to achieve a better understanding of the data in terms of important concepts and their relationships (Lochmiller, 2021).

In this case, the researcher explained the participants' input using the participants' own words, as relevant to the research questions (Castleberry & Nolen, 2018). The

[1] http://qsrinternational.com

researcher was, therefore, able to classify the data based on its similarities and differences (Braun & Clarke, 2012). The researcher extracted the information following the steps as described in Table 3.2.

Table 3.2: Phases and stages of theme development in qualitative content and thematic analysis

Phases	Stages
Initialisation	▪ Reading transcriptions and highlighting meaning units. ▪ Coding and looking for abstractions in participants' accounts. ▪ Writing reflective notes. ▪ Classifying. ▪ Comparing.
Construction	▪ Labelling. ▪ Translating and transliterating. ▪ Defining and describing. ▪ Immersion and distancing.
Rectification	▪ Relating themes to established knowledge. ▪ Stabilising.
Finalisation	▪ Developing the story line.

Source: Vaismoradi, Jones, Turunen & Snelgrove, 2016:103

According to Braun and Clarke (2012), thematic analysis can be done either inductively or deductively. The inductive approach was found to be suitable for the current study because the research involves the participants' points of view, experiences and feelings which are often subjective. The inductive approach is also justified by the fact that the data patterns were extracted directly from the participants' input and are thus closely linked to the data. Figure 3.4 shows the practical steps that the researcher followed during the process of obtaining information from interview data.

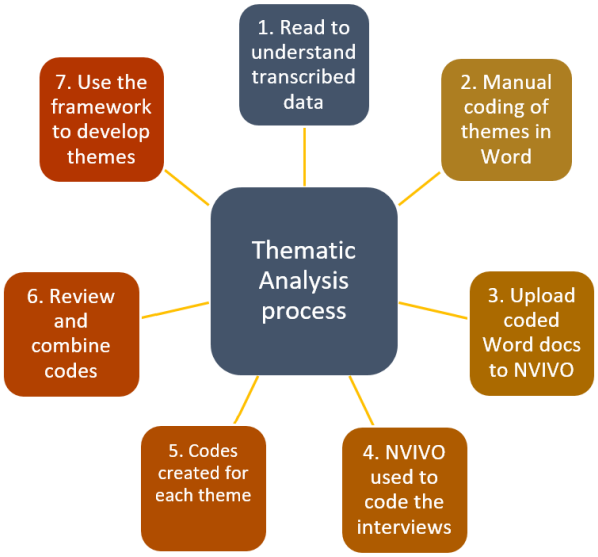


Figure 3.4: Practical steps to get information from collected data

3.3.4 Construct and Refine

These are the fourth and fifth steps of the case study methodology. During these steps the researcher generated a table, which contained a summary of the codes allocated to each respondent.

3.3.5 Reflect

The last step of the case study methodology was to reflect on what results were produced with the various methods and to determine if it addressed the research questions satisfactorily. If not, some more interviews or observations would need to be done.

3.4 VALIDITY AND RELIABILITY OF A SINGLE CASE STUDY

A case study enhances the trustworthiness and credibility of the research if it involves data triangulation, where multiple sources of evidence are used for the same phenomenon in a single unit of analysis (Crowe *et al.*, 2011). Maintaining a chain of evidence helps to increase the research's validity (Baskarada, 2014).

The researcher interviewed different DevOps role players at the identified organisation to obtain different views regarding the transformation required during the adoption of DevOps. To help maintain the chain of evidence, the researcher cited all the literature sources used in the current study and tabulated each question with the answers that the participants gave. Furthermore, the research findings were based on interviewees' inputs.

Reliability is the ability of other researchers to come up with the same result when conducting the same research using the same research method (Baker, 2006). Giving other researchers the opportunity to do the same research again rarely happens in real-life situations but there are ways to ensure the repeatability of the research. The use of a case study protocol that clearly outlines the details of all the procedures that were used to carry out field research, and the development of a case study database to store the case study protocol and the chain of evidence in a manner that is retrievable are important conditions for reliability (Baxter & Jack, 2008).

Finally, the research outcome must also be presented in a professional manner. That is, it must be neat, easy to read and understandable (Yin, 2018).

3.5 ETHICAL CONSIDERATION IN CASE STUDIES

This research project was guided by the values and principles expressed in the policy document of the University of South Africa (UNISA) on research ethics. The ethics review committee reference number for this research is 2021/CSET/SOC/061. Before field research permission was granted by UNISA's research ethics committee, the researcher answered a list of questions related to the conduct of the researcher. The researcher also provided the committee with information about the research methodology and a summary of the research topic for its assessment.

After the research was approved by the ethics review committee, the researcher could start with the field research but had to adhere to the research criteria stipulated by the committee. This study subscribes to the methods and procedures as outlined in the ethics application process; the ethics clearance certificate can be found in Appendix A. The researcher safely stored research recordings, notes and sources on a computer, and it will be destroyed once the study is completed.

3.6 CONCLUDING SUMMARY

In this chapter, the researcher motivated why a single case study is an appropriate choice to use for this type of research. A single case study is reliable when collecting data for a short period of time and when resources are limited.

The next chapter deals with the research findings. The comparison between the research findings and literature will help to clarify the new findings of the research and demonstrate the reliability of the research findings.

CHAPTER 4:
FINDINGS

4.1 OVERVIEW

The previous chapter discussed the research design, as well as how the data was collected. The aim of this chapter (Chapter 4) is to outline the results of the single case study. The results of each step of the case study methodology will be explained.

4.2 RESULTS OF THE RESEARCH DESIGN

Figure 4.1 shows the results of each step of the case study design of the current study.

4.2.1 Define

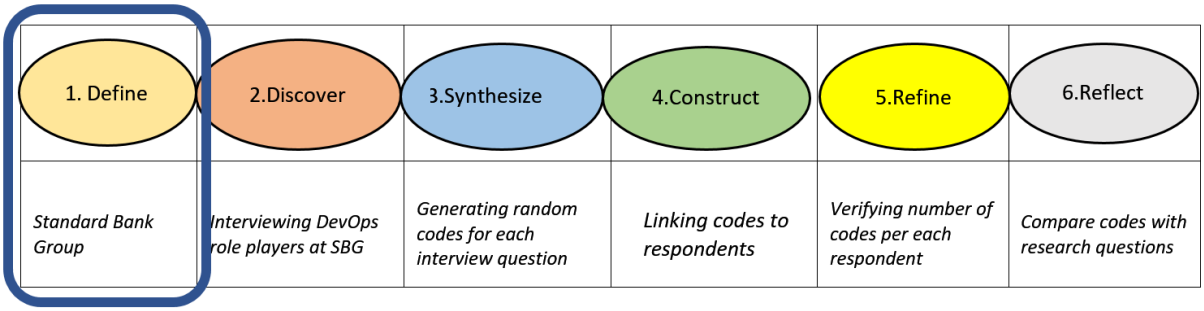


Figure 4.1: Results of the Define step of the case study methodology

The Bank A was selected as the most appropriate company to be used for this study. The bank is one of the biggest multinational banks in South Africa and it adopted DevOps to enhance its IT operations (SBG, 2020).

4.2.1.1 General overview of Bank A

Bank A is one of the leading financial services providers in Africa, and it also offers its financial services abroad. It operates in 20 African countries and is headquartered in Johannesburg South Africa. Some of the countries where it does business abroad include Brazil, the United States of America, the United Kingdom and the United Arab Emirates. Besides being listed on the Johannesburg Stock Exchange, the bank's 158 years of existence has seen it becoming the largest banking group in Africa by assets (SBG, 2020). The purpose and strategy of Bank A is shown in Figure 4.2.

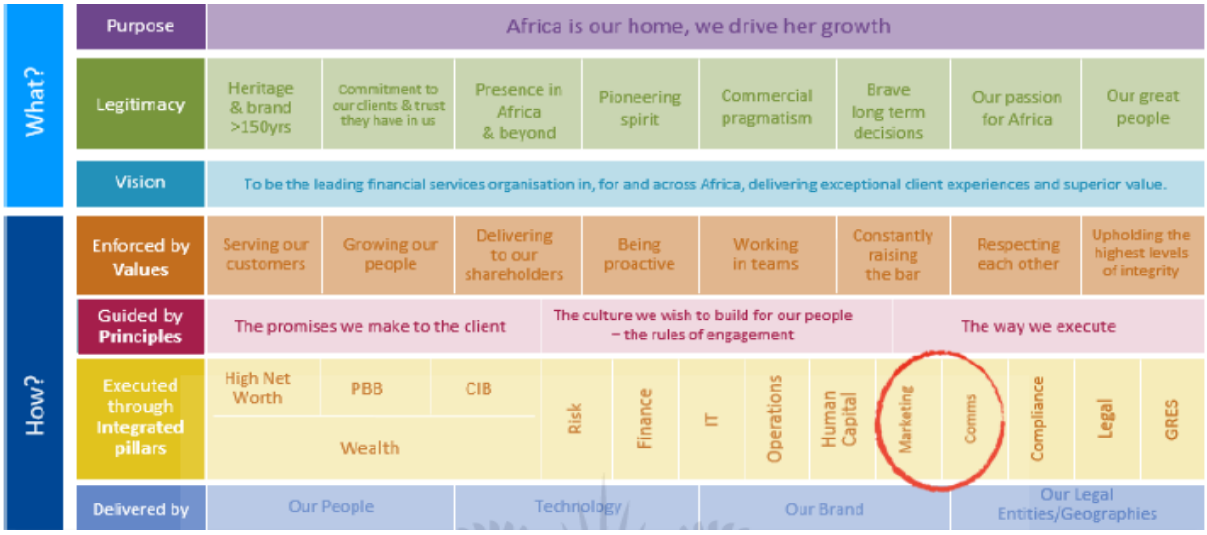


Figure 4.2: Bank A purpose and strategy

Source: SBG, Group marketing and communication strategy, 2016:2

4.2.1.2 Information Technology at Bank A

Bank A understands that IT is the backbone of their business, and therefore, they have a dedicated IT team that ensures that their IT operations subscribe to the best practices and are of the best standards. The IT team's ultimate objective is to maintain an exceptional degree of customer satisfaction using technology. The bank has successfully managed to complete different massive IT projects using leading IT project management methodologies, such as Agile and DevOps (Marnewick & Langerman, 2020).

Over the past years the bank has invested significant resources to enhance their IT operations in terms of infrastructure and systems. They implemented DevOps to modernise and increase the effectiveness and efficiency of their IT delivery to their customers, and to cope with the rapid transformation of the IT industry (SBG, 2020).

Bank A's financial statement ranks IT spending as one of their biggest expenses. This demonstrates the commitment that the bank has to ensure that their digital transformation subscribes to globally accepted standards and uses the best infrastructure. The bank employs an experienced and highly qualified digital transformation team that ensures that their systems provide best quality of services to their customers (SBG, 2021).

4.2.1.3 Justification for using Bank A as a single case study

Although Bank A is not primarily an IT organization, it uses information technology to enable the functioning of its financial services. Therefore, it has an IT organisational structure (Marnewick & Langerman, 2020). According to Blanton, Watson, Milojicic and Moody (1992), an IT organizational structure is an establishment within an organisation that is tasked with achieving IT goals and it normally consists of development, systems and operations department. The researcher justified the use of a multinational organisation such as Bank A because of the need to obtain various individuals' perceptions from the organisation regarding the way they made some changes when implementing DevOps. As outlined earlier, the financial services giant has the largest DevOps deployment in South Africa and operates in many countries. As a result, the bank managed to provide extensive empirical data, obtained from well-experienced DevOps role players who are senior employees of the bank.

Bank A's exposure to both developed and developing markets provided in-depth insight into why the speed of transformation during DevOps implementation in some markets is slower than others. The different markets that the bank operates in presented an opportunity to obtain data from a variety of sources regarding the way an IT organisation restructures when implementing DevOps, the challenges it encounters, and the effectiveness of DevOps tools used.

Due to its deployment of DevOps at a large scale, the researcher found Bank A fit for the purpose of confirming, challenging, and extending the facts regarding the impact of DevOps on IT organisations. The researcher was able to identify the ways that are used to change the organisation during DevOps implementation based on the respondents' input. The expectation is that this will help IT businesses, academic research communities and other business stakeholders to better grasp the effects of DevOps on digital transformation.

4.2.2 **Discover**

This step included identifying the DevOps role players, employees of Bank A, to interview. Each interview occurred as per the respondent's suggestion (see Figure 4.3).

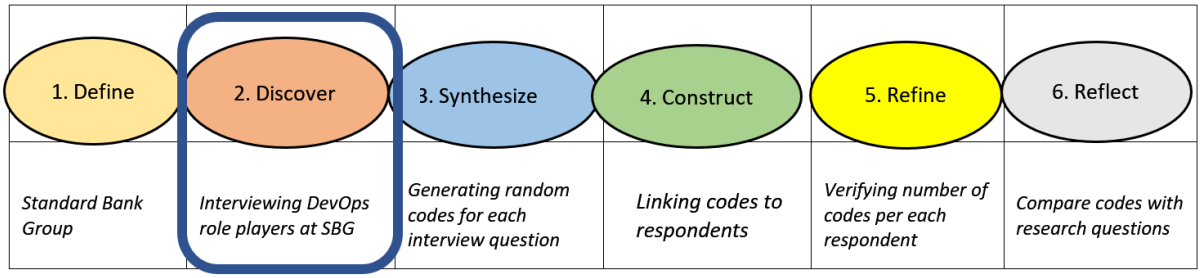


Figure 4.3 Results of the Discover step of the case study methodology

The interviews were guided by the following probing questions:

- How did you restructure your organisation during DevOps implementation?
- Are there any other changes related to DevOps transformation that occurred?
- What challenges did you encounter during DevOps transformation?
- What can you say are the possible solutions to the challenges?
- To what extent did the DevOps tools that you use help to implement DevOps?

Each interview took on average less than an hour. All the interviews were done virtually using Microsoft Teams.

4.2.2.1 Interview probing questions 1 and 2

Table 4.1 shows the different ways that can be used to achieve DevOps adoption, as suggested by the respondents. The aim of this is to minimise negative effects, while maximising the positive outcomes to the business. These suggestions will be discussed below based on the respondents' input.

Table 4.1: Summary of DevOps restructuring approaches

Interview probing questions 1 and 2	Restructuring codes
How did you restructure your organisation during DevOps implementation?	▪ Cross-functional teams. ▪ Understand your current business, its vision and mission. ▪ Assessment of current technology usage and culture setup. ▪ Get information about the skills structure of the organisation. ▪ Do a DevOps impact assessment to find out how it will add value to the organisation. ▪ Be knowledgeable about the state of your business's operating model. ▪ Market the DevOps idea in the organisation to change the employee's mindset. ▪ Design DevOps maturity model based on DevOps best practices.

Are there any other changes related to DevOps transformation that occurred?	▪ Identify pilot project with which to start testing DevOps practices. ▪ Experimentation and learning fast.
	▪ Use DevOps metrics to measure your progress and keep on improving. ▪ Ensure system security is part of every stage of DevOps implementation. ▪ Use of dashboards.

Cross-functional teams

Respondent 4 commented: “You build it, you run it”. The interviewee added that teaming and proper tooling are an important part of DevOps transformation. Both developers and operators need to work together to speed up software development and deployment. Respondent 1 also emphasised the need to have cross-functional teams that can help to improve software delivery. The literature also pointed out the need to have cross-functional teams when implementing DevOps (Erich *et al.*, 2017).

Understanding your current business and its vision and mission

“A good understanding of the goals and aspirations of the future are a critical factor when wanting to implement DevOps” was the point raised by Respondent 2. According to Respondent 1 it is imperative that before any DevOps changes can be done, the organisation understands the business that it is doing and their short- and long-term objectives. Respondent 4 added that adopting DevOps should be justified by the desired future position of the organisation. According to literature, it is important to understand an organisation’s current system (Crowley *et al.*, 2018).

Assessment of current technology usage and culture setup

“Culture of the organisation is important to understand in terms of the shared values and behaviours of the employees” claimed Respondent 3. Respondent 2 commented that an organisation must take note of the tools they are using currently, and their effectiveness and efficiency in terms of delivering outcomes. Respondent 1 added: “DevOps is about sharing, be it tools, responsibility or work, so a good understanding of the current culture will enable the company to focus on areas of improvement during restructuring”. Respondent 4 suggested that working as a team is important to achieve goals.

According to the literature, technology assessment should be a continuous activity to ensure high quality (Wurster *et al.*, 2013). In addition, according to Ebert *et al.* (2016), the individuals’ behaviour must be continuously revised to align to the required norms (Ebert *et al.*, 2016)

Obtain information about the skills structure of the organisation

Respondent 2 commented: “Sometimes you have to work with the human resources department to identify skills gap and find ways to solve the problem”. Respondent 1 indicated that it is important to know the type of skills and experience that an organisation has, so that as you gradually transform the organisation to DevOps, you take advantage of the strength of the current skill set, while reskilling to make some improvements, or you get the skills from outside the organisation. Respondents 1, 2 and 3 emphasised that as an organisation implements DevOps, the need for multi-skilled employees who can work in different departments, be it development, testing or operations, becomes a fundamental requirement.

The literature does not explicitly state the need to involve the human resources department to obtain more skills, but the need to hire employees is pointed out (Sánchez-Gordón & Colomo-Palacios, 2018).

Do a DevOps impact assessment to find out how it will add value to the organisation

“When you implement DevOps, you want to be confident that it is the right thing to do” was a statement made by Respondent 1. Respondents 1 and 2 pointed out that as an organisation you do not want to feel that DevOps is not the right thing to do when it has already been implemented. Therefore, it is important to know the market in which your business operates, especially in terms of customer behaviour and demands. Once you are convinced that the product that you are offering needs to be improved continuously and delivered fast then that can be a good factor for the implementation of DevOps. This point was not stated as clearly in literature as it was by the respondents.

Be knowledgeable about the state of your business’s operating model

Respondent 1 said: “Most businesses have legacy systems that are characterised by monolithic architecture and have on-premise data centres”. He added: “You have to let go a lot of legacy things and embrace experimentation and improvement”. Respondent 1 also explained that the legacy systems are normally known as the traditional operating models. The focus of this model is to maintain the stability and efficiency of the system, while ensuring that each employee is assigned a specific

responsibility. Departments are siloed and there is limited collaboration within the organisation and with customers.

Respondents 1, 2 and 3 added that when implementing DevOps in an organisation that has a traditional operational model, the first step is to shift to a multi-operational model. According to the respondents, this model involves the partial change of legacy systems to digital platforms, where they begin to operate in the cloud. The goal is also to gradually change the monolithic architecture into a microservices architecture, which is made up of small pieces of software that communicate with each other and are easy to troubleshoot and upgrade. Within this model the expectation is also that employees begin to collaborate closely and work as a team, while at the same time, communicating regularly with the customers to understand their needs and implement them. The respondents added that the workflow becomes automated to eliminate manual processes and improve quality of product, speed of processing, reliability and repeatability.

Respondent 1 pointed out that once the multi-model has been achieved and stabilised, the next objective is to achieve what they call a platform model. This model is characterised by a reduced and managed legacy system, more IT infrastructure that is modernised and cloud-based, an optimised workflow that is automated, fast feedback loops, and continuous learning and improvement. More emphasis is placed on customer-focused delivery and increased customer collaboration. The organisation becomes more flexible and can respond to the changes in the market swiftly and efficiently. Operations are set to be continuous to accomplish the continuous delivery of products to the customer and continuous monitoring of system functionality in production.

DevOps metrics are used to measure the level of success and failure, and the workflow is continuously improved. In summary, the transformation from the traditional model to the platform model is shown in Figure 4.4.

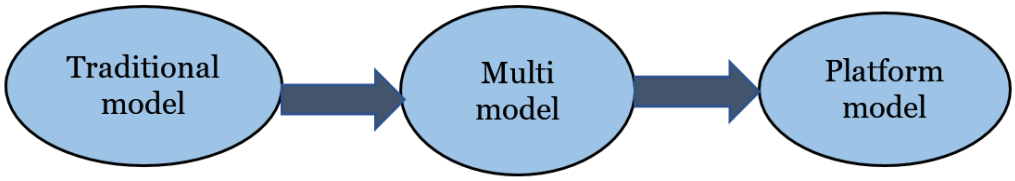


Figure 4.4: Summary of technology operating models' transformation, as suggested by interviewees

According to the literature, the implementation of DevOps necessitates the shift to microservices and the cloud (Chen, 2018). However, the respondents added more detail in relation to the changes occurring in an organisation with the implementation of DevOps.

Market the DevOps idea in the organisation to change the employee's mindset

With regard to this theme, Respondent 2 remarked: "There is need to market the idea of DevOps to the organisation to convince the employees that it is the right thing to do". Respondents 1 and 3 added that it is important that the information about DevOps practices, such as automation, close collaboration and the need to be multi-skilled, are continuously being shared with employees, together with the benefits that are derived from them. If employees are to comply, they need to be convinced about how a change to DevOps will benefit both the company and individual employees.

The literature stated that the critical first step is to change people's mindset to enable them to positively take part in a change (Luz *et al.*, 2019).

Design DevOps maturity model based on DevOps best practices

Respondent 1 stated: "DevOps maturity assessment has got six functions, we look at continuous planning, continuous development, continuous testing, continuous deployment, continuous monitoring and continuous logging". Respondent 1 added that it is important that during or before the implementation of DevOps, an organisation comes up with a DevOps maturity model so that they can measure their progress at each stage and initiate improvements. According to Respondents 1, 2 and 4, the DevOps maturity model is made up of DevOps expectations at each stage of the DevOps implementation.

Since adopting DevOps is a process and not an event, different stages of the process should have a checklist of the activities that must be achieved regarding DevOps practices, such as automation, continuous integration, continuous deployment, continuous delivery, continuous monitoring, best use of toolsets, close collaboration, and continuous learning and improvement. In addition to the checklist, the organisation needs to note down a strategy of how they intend to achieve each step. Setting up different expectations at each stage of the implementation will act as a guideline to motivate and push the team to achieve greater results.

According to the literature, DevOps maturity models are important to measure the progress of DevOps (Díaz *et al.*, 2019).

Identify pilot project with which to start testing DevOps practices

Respondent 1 commented: “It is important that when you implement DevOps for the first time, you do not start it on a big project, rather start it on a small project that is easy to manage”. Respondents 3 and 4 added that it is necessary to have different departments to start working together on that pilot project, sharing ideas on how best to automate and increase the efficiency of the pilot project. This can be used as the first experiment to change technology and culture, therefore, it is imperative that notes be taken of successes and failures to be used as lessons learnt during the next project.

The literature did not discuss the idea of identifying a pilot project.

Use DevOps metrics to measure your progress and keep on improving

Respondent 2 emphasised: “It is important that you measure the improvements of DevOps to understand your successes and failures”. Respondent 1 added that it is important to go beyond the Quality Assurance (QA) department’s scope of work and do extra by measuring the system’s performance in real time. Respondents 2, 3 and 4 commented that the ability to measure the functionality of software in production provides insight about the usefulness of software and enhances the decision-making process about things that need to be fixed and improved.

The literature also stated the need to make use of tools that measure software performance and continuously provide feedback (Wurster *et al.*, 2013).

Ensure system security is part of every stage of DevOps implementation

“Everything that we do must have a security application”, were the words of Respondent 1. Respondent 2 commented that there is need to ensure that the system is highly secured while the DevOps transformation is taking place to protect it against unauthorised users and hackers. Every time the system is improved, the security needs to be revised and adjusted accordingly to ensure it remains relevant and effective.

Wilde *et al.* (2016) noted that security must be part of every DevOps process.

Use of dashboards

"Teams could go to DevOps dashboards and see how they are performing against the targets that they have set themselves", was the point raised by Respondent 4 to explain how dashboards are used. According to the interviewee, the dashboards help to enforce transparency within the organisation, as employees can see the loads that have been processed successfully and those that have failed.

The need for dashboards is also noted in literature (Humble & Farley, 2010).

Experimentation and learning fast

Respondent 2 commented: "People should be willing to learn, experiment and fail fast". Respondent 4 added that experimenting during development is important for creativity.

Experimenting is one of the requirements for DevOps culture (Sánchez-Gordón & Colomo-Palacios, 2018).

4.2.2.2 Interview probing questions 3 and 4

Table 4.2 shows a summary of the challenges that the interviewees encountered during the implementation of DevOps.

Table 4.2: Summary of DevOps restructuring challenges

Interview probing question 3	Responses (DevOps restructuring challenges codes)
What challenges did you encounter during DevOps transformation?	▪ Manual intervention. ▪ Legacy change management processes. ▪ Silo mentality. ▪ Unwillingness to learn new skills and processes. ▪ Lack of executive management support. ▪ Employees are not collocated. ▪ High DevOps implementation success rate concentrated on pilot project and not across all projects. ▪ Indecision when it comes to which part of the business to outsource during restructuring. ▪ Dependence on individual brilliance and heroic efforts. ▪ Automating waste, creating automation before fixing bottlenecks and system failure points. ▪ Changing management employees. ▪ Not taking IT governance seriously. ▪ Lack of shared ownership and accountability due to silos.

Interview probing question 3	Responses (DevOps restructuring challenges codes)
	▪ Lack of operational maturity. ▪ Use of different technologies and different configurations within the same environment. ▪ Achieving a high maturity DevOps structure is an event, and not a process. ▪ Outdated testing practices. ▪ Collective accountability.

The researcher asked the respondents regarding the possible solutions to the challenges outlined in Table 4.2, and these are listed in Table 4.3.

Table 4.3: Summary of possible solutions to the restructuring challenges

Interview probing question 4	Responses (Solutions to DevOps restructuring challenges)
What can you say are the possible solutions to the challenges?	▪ Ensure there is accountability and realistic turnaround time for every approval process. ▪ Automate some of the approval processes. ▪ Set realistic, achievable, and timeous goals, and track them and share findings. ▪ Subscribe to uniform and best standards of development and deployment to ensure stability and repeatability. ▪ Create a DevOps manifesto based on past failures and successes and learn from it to improve. ▪ Achieve a higher degree of automation to minimise manual processes and eliminate any processes that do not add value. ▪ Ensure there are fast feedback loops within the deployment pipeline to fail fast and recover fast. ▪ Work as a team, respect each other. ▪ Effective communication to convince employees about the benefits of the changes to them. ▪ Ask human resources department to help fill skills gap. ▪ Make use of online video conferences. ▪ Organise team building sessions to meet and talk (after Covid-19 is gone). ▪ Communicate and thoroughly consult with all business stakeholders to help make a good decision. ▪ Investigate the root cause of this and fix the problem fast. ▪ The first step is to identify bottlenecks and points of weaknesses, and then to use automation to solve them. ▪ Embrace new ideas from new employees but ensure that the ideas are meant to improve the workflow in line with the vision of the organisation. ▪ Automate the testing.

Interview probing question 4	Responses (Solutions to DevOps restructuring challenges)
	▪ Make testing and quality assurance everybody's responsibility, let developers also test and testers also develop. ▪ Establish governance to set the required standards that control IT processes and make everyone accountable. ▪ Continuously assess your operational processes, such as access management, incident management, request management and problem management, to enable you to improve their agility and transparency. ▪ Standardise infrastructure technology to be identical and to subscribe to the best practices. ▪ Create an environment that promotes continuous learning and improvement. ▪ Be patient, as a high velocity organisation cannot be achieved overnight and stick to basics. ▪ Increase collaboration between different departments. ▪ Promote cross-skilling and market the benefits of that to the employees. ▪ Create cross-functional teams. ▪ Encourage the required behaviour by giving incentives and rewards. ▪ Implement some initiatives from the bottom and ensure that everybody understands DevOps. ▪ Use metrics to measure successes, and then use positive outcomes to justify to the executive the need for DevOps and obtain their support.

The suggestions in Table 4.2 and 4.3 will be discussed in more detail. The objective of the research questions in these tables was to outline the difficulties that an organisation encounters as it embarks upon a process of implementing DevOps, and to come up with possible solutions to the identified problems.

Manual intervention

According to Respondent 1, "Manual intervention leads to human error and non-repeatable processes". Respondents 2 and 3 share the same sentiment that during restructuring, the continued use of manual processes that require more human involvement affect the quality and speed of work because the manual ways of doing things are not repeatable and are unreliable.

Respondents 1, 2 and 3 pointed out that the automation of the entire workflow is a critical requirement to gain confidence in the system's ability to deliver high-quality products.

The literature also indicated that manual processes need to be eliminated in favour of automation (Gill *et al.*, 2018).

Legacy change management processes

“You have to constantly ask for approval to do things”, was a concern raised by Respondent 1. Three respondents believe that long approval processes result in bottlenecks, since it normally takes time for a task to be approved, hence, delaying the entire workflow. Therefore, they think that a more agile approach that involves the automation of some approval processes will be ideal to help speed up task approval.

The literature raised a concern about the negative impact of long approval processes (López-Fernández *et al.*, 2021).

Silo mentality

Respondent 1 stated the following: “Code moves back and forth because we are not aligned”. Respondents 2, 3 and 4 pointed out that some departments are isolated in such a way that they do not communicate or cooperate with other departments. This creates a challenge when sharing skills, responsibilities and working as a team during the implementation of DevOps. The respondents think that there is need to effectively communicate the concept of DevOps and its practices, as well as its benefits to all the departments to encourage close collaboration.

According to the literature, IT organisations that evolve during DevOps do experience a silo mentality (Senapathi *et al.*, 2018).

Unwillingness to learn new skills and processes

According to Respondent 3, “Introducing DevOps means learning new things that are uncomfortable”. Respondents 1 and 3 noted that some employees do not want to change. They want to stick to what they already know. This made it difficult during DevOps changes because the employees must gain extra skills so that they can work in different departments. Respondent 3 added that some employees were not interested in taking on more work and responsibilities. Both respondents (1 and 3) suggested that there is need to convince employees about the positive effects that come along with these changes for them to comply.

The literature also outlined resistance to change as one of the challenges of DevOps adoption (Smeds *et al.*, 2015).

Lack of executive management support

According to Respondents 1, 3 and 4, the moral and financial support from top management was not provided to the expected level. “DevOps is expensive, and you need support from the executive for it to be successful”, was the point raised by Respondent 3. Respondent 3 added that lack of executive support was a demotivation during DevOps restructuring because an initiative that lacks proper top management support has a chance of failing along the way.

To mitigate this, Respondent 4 suggested that if a change starts from the bottom, then there is need to do it correctly so that the positive impact broadcasts itself to the entire organisation and draws the attention of the naysayers. Respondent 1 added that there is need to engage the top management to highlight to them the benefits that the change is bringing and to encourage them to support it.

Khan *et al.* (2020) discussed the need for support from top management.

Employees are not collocated

“If employees are located far away from each other, the culture element becomes difficult”, was the point raised by Respondent 2. Three respondents felt that this issue had a negative effect to the culture during restructuring and stated that it is difficult to build a team relationship with other employees who are in other countries or locations. Although the use of video technology to interact can help, it is not as effective as face-to-face meetings.

According to the literature, organisations that have some entities that are not collocated find it difficult to collaborate (Bucena & Kirikova, 2017).

High DevOps implementation success rate concentrated on pilot project and not across all projects

This was a suggestion from three of the respondents. Respondent 1 felt that it had become a common thing that DevOps implementation normally succeeds on the pilot project but does not do well on big projects. Respondent 2 shared the same sentiment but added that maybe it is time that they take a different approach on bigger projects but stick to DevOps basics. The suggestion was also that they should use the same automated scripts to ensure repeatability and to increase the chances of success.

Respondent 3 pointed out that it is critical to set realistic and achievable objectives for a bigger project and to be a bit more creative.

This contribution raises an issue that has not been dealt with in literature before.

Indecision when it comes to which part of the business to outsource during restructuring

According to Respondent 2, during restructuring for DevOps there are some business sections that may need to be outsourced to enable the organisation to focus on their main areas of interest. Respondent 2 stated: “The decision to know which part of business to outsource is not an easy one”. Respondent 3 commented that making the right decision requires all senior managers to collaborate and to participate in the discussion to come to a final decision.

The researcher did not find information about this point in the literature.

Dependence on individual brilliance and heroic efforts

All four respondents highlighted this problem and stated that most of the times there are individuals who are known to be the best at solving problems. “A guy comes, and he wants to be the hero but he is probably the cause”, was the point raised by Respondent 1 to disapprove individual brilliance. Respondent 2 added that this issue normally causes other team members to depend on the perceived heroes. This may not be a good thing for DevOps, since employees need to share ideas and be cross-skilled. The common suggestion is that this needs thorough investigation to find the root cause and then apply the solution accordingly.

The literature also noted that employees need to work together as a team to achieve success (Mangot, 2016).

Automating waste, creating automation before fixing bottlenecks and system failure points

Two of the respondents felt that the issue of automating waste sometimes creates more problems than solutions. Respondent 1 maintained: “You automate but you automate the wrong things”, and continued to say that this is normally caused by automating part of the system before analysing the entire system to identify points of weaknesses and failure. Consequently, the automation does not work properly. Respondent 4 also made a similar suggestion and pointed out that system thinking is

critical before any automation can be done. In this case, it is important to do an automation impact assessment to establish possible areas of concern, and to fix them before automating.

The literature did not state this issue as clearly as the respondents did.

Changing management employees

According to Respondents 2 and 3, this has caused negative outcomes to the DevOps restructuring approach. “Sometimes an employee who has been a key driver to a certain program is changed and replaced by a new employee”, was said by Respondent 3. Both Respondents 2 and 3 added that the new employee sometimes brings new ideas which may not be the right ideas to move the project forward. Respondent 3 suggested that when this happens, there is need for the new employee to collaborate with other employees to help make effective decisions.

This suggestion was not noted in the literature.

Not taking IT governance seriously

Respondents 1, 2 and 3 pointed out that not having documented procedures to follow, cause an ad hoc and chaotic system upgrading process. Respondent 1 suggested that it is important to have standard and documented procedures to use as a reference point when developing the system.

Shahin *et al.* (2017) noted that compliance requirements must be understood before implementing DevOps.

Lack of shared ownership and accountability due to silos

Respondent 1 felt that when some departments are isolated, it becomes difficult to take collective responsibility or to work as a team, as per the DevOps basic practices. Respondents 2, 3 and 4 commented that culture change needs to be encouraged and incentivised to achieve the expected behaviours.

The literature suggested the need to change culture (Lwakatare *et al.*, 2015).

Lack of operational maturity

According to Respondent 1, this is caused by the traditional setup which involves more processes and activities, such as access management and incident management. All

four respondents said that the legacy system is full of traditional operation models. The transformation, as they suggested, is unfortunately not a quick one, as it will require the employees to analyse the existing model and gradually make changes that modernise it and help increase productivity.

The issue of legacy systems as a challenge to achieving continuous deployment was raised in literature (Smeds *et al.*, 2015).

Use of different technologies and different configurations within the same environment

Respondent 1 stated: “Inconsistent environment, different configurations within each environment”. Respondent 2 emphasised that this becomes a challenge during DevOps restructuring because of the need to make use of a single platform that supports all the various technologies. Respondents 1, 2 and 4 made a similar suggestion that there is need to make a long-term plan to ensure that the technology and the configuration platform are identical. This requires planning related to tooling and to institute principles that govern the toolsets.

Smeds *et al.* (2015) stated that having different environments with different configurations may make it difficult to achieve continuous delivery.

Achieving a high maturity DevOps structure is an event not a process

Respondents 1 and 4 indicated that sometimes when the implementation of DevOps starts, the team has high expectations of getting a return on investment in a short time. However, this often turns out to be a disappointment because changing a massive legacy system to a modern and efficient system takes time, and along the way things may not work as expected. During the transformation, and especially, during the times when things do not seem to go as planned, the most important thing is to remain focused, stick to basics and keep on reinventing.

The literature noted that implementing DevOps is a gradual process (Leite *et al.*, 2019), but not in a negative way.

Outdated testing practices

“Measuring and rewarding using collective accountability is something the organisation struggled with, it is still very much individual based”, was the point raised by Respondent 4 regarding the idea of being rewarded as a team rather than as

individuals. According to the interviewee, recognising and incentivising employees as a team negatively affected the effectiveness and sustainability of the implementation of DevOps.

The literature did not raise this challenge.

Collective accountability

Similar to the findings related to outdated testing practices discussed above, Respondent 4 had this to say about collective accountability and being rewarded as a team, rather than an individual: “Measuring and rewarding using collective accountability is something the organisation struggled with, it is still very much individual-based”. According to the interviewee, recognising and incentivising employees as a team negatively affected the effectiveness and sustainability of the implementation of DevOps.

The literature did not raise this challenge.

4.2.2.3 Interview question 5

Table 4.4 presents the codes related to interviewees’ responses in relation to the effectiveness of the DevOps tools that they used.

Table 4.4: Summary of issues related to DevOps tools

Interview question 5	Responses (Effectiveness of DevOps tools codes)
To what extent are the DevOps tools used in organisations able to deliver the expected business results?	▪ Respondents were satisfied with the DevOps tools, such as code management, deployment, repository and monitoring tools. ▪ Deployments could happen during the day instead of only during nights and weekends. ▪ Lack of required skills to use DevOps tools was a big concern.

The points raised by the respondents regarding DevOps tools in Table 4.4 are discussed as follows.

“Instead of doing deployments over the weekend or at night only, some deployments were done in production during the day when we started all this”, was the reply from Respondent 4 regarding the effects of DevOps tools. All the respondents said that they are satisfied with the DevOps tools that are used, namely, code management tools, deployment tools, repository management and monitoring tools. Respondent 3 stated:

“Even if tools are brought into the organisation there is always room for improvement whether it is for the tool itself or the way we use the tool”. This answer emphasised the fact that, in some instances, employees will need to be retrained to effectively use the tools.

A point regarding the shortage of skills was also raised in literature (Senapathi *et al.*, 2018). Respondent 3 also suggested that some employees did not have the required skills and experience to use some of the DevOps tools, therefore, they needed training. Respondent 2 added that it is important to make strategic decisions about tooling to avoid wasting money buying unnecessary tools, and to ensure that the tools available are being used productively.

4.2.3 **Synthesise**

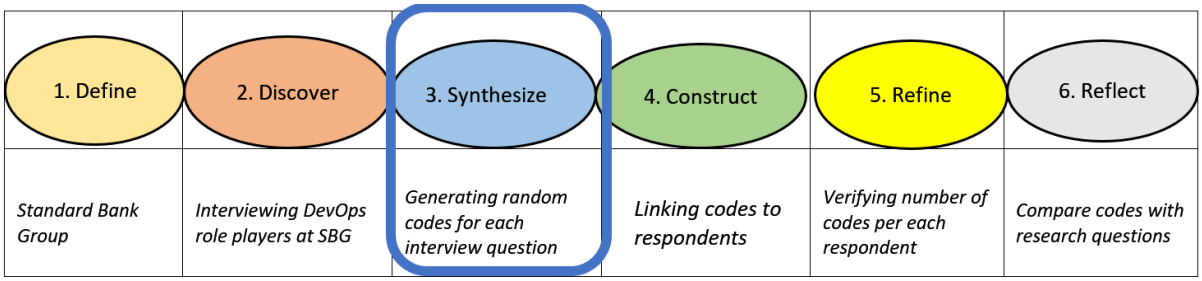


Figure 4.5 Results of the Synthesise step of the case study methodology

Random codes were extracted from interview recordings and notes. These were then grouped per research question.

4.2.3.1 Suggestions for Research sub-question 1

Figure 4.5 shows the respondents’ suggestions in terms of DevOps restructuring that answered the first research sub-question.

Table 4.5: DevOps restructuring strategies

Research sub-question 1	DevOps restructuring ways
How should organisations restructure when implementing DevOps?	▪ Cross-functional teams. ▪ Understand your current business, its vision and mission. ▪ Assessment of current technology usage and culture setup. ▪ Obtain information about the skills structure of the organisation. ▪ Do a DevOps impact assessment to find out how it will add value to the organisation. ▪ Be knowledgeable about the state of your business's operating model. ▪ Market the DevOps idea in the organisation to change the employees' mindsets. ▪ Design DevOps maturity model based on DevOps best practices. ▪ Identify a pilot project with which to start testing DevOps practices. ▪ Use DevOps metrics to measure your progress and keep on improving. ▪ Ensure system security is part of every stage of DevOps implementation. ▪ Use of dashboards. ▪ Experimentation and learning fast.

4.2.3.2 Suggestions for Research sub-question 2

Table 4.6 shows a summary of DevOps restructuring challenges and the proposed solutions, as suggested by respondents.

Table 4.6: Summary of restructuring challenges and proposed solutions

Research sub- question 2	Restructuring challenge	Related solution suggested
What challenges are encountered during restructuring and how can these challenges be mitigated?	Manual intervention.	▪ Automating the entire workflow to minimise human involvement.
	Legacy change management processes.	▪ Ensure there is accountability, and a realistic turnaround time for every approval process. ▪ Automate some of the approval processes.
	High DevOps success rate concentrated on pilot projects, while limited across all projects.	▪ Set realistic, achievable, and timeous goals, and track them and share findings. ▪ Subscribe to uniform and best standards of development and deployment to ensure stability and repeatability. ▪ Create a DevOps manifesto based on past failures and successes and learn from it to improve. ▪ Ensure there are fast feedback loops within the deployment pipeline to fail fast and recover fast.
	Unwillingness to learn new skills and processes.	▪ Effective communication to convince employees about the benefits of the changes to them. ▪ Ask human resources department to help fill skills gap.
	Employees are not collocated.	▪ Make use of online video conferences. ▪ Organise team building sessions to meet and talk.
	Indecision when it comes to which part of the business to outsource during restructuring.	▪ Communicate and thoroughly consult with all business stakeholders to help make a good decision.
	Automating waste, creating automation before fixing bottlenecks and critical infrastructure points.	▪ The first step is to identify bottlenecks and points of weakness and then use automation where possible to solve them.

Research sub- question 2	Restructuring challenge	Related solution suggested
	Changing management employees.	▪ Embrace new ideas from new employees but ensure that the ideas are meant to improve the workflow in line with the vision of the organisation.
	Outdated testing practices.	▪ Make testing and quality assurance everybody's responsibility, let developers also test and testers also develop.
	Not taking IT governance seriously.	▪ Establish governance to set the required standards that control IT processes and make everyone accountable.
	Lack of shared ownership and accountability due to silos.	▪ Continuously assess your operational processes, such as access management, incident management, request management and problem management, so that you can improve their agility and transparency.
	Use of different technologies and different configurations within the same environment.	▪ Standardise infrastructure technology to be identical and to subscribe to the best practices.
	Achieving a high maturity DevOps is a process not an event.	▪ Be patient, as a high velocity organisation cannot be achieved overnight and stick to basics. ▪ Create an environment that promotes continuous learning and improvement.
	Lack of operational maturity.	▪ Increase collaboration between different departments. ▪ Promote cross-skilling and market the benefits of doing that to the employees.
	Silo mentality.	▪ Create cross-functional teams. ▪ Encourage the required behaviour by giving incentives and rewards.
	Lack of executive management support.	▪ Implement initiatives from the bottom and ensure that everybody understands DevOps. ▪ Use metrics to measure successes and then use positive outcomes to justify to the executive the need for DevOps and their support.

4.2.3.3 Suggestions for Research sub-question 3

Table 4.7 displays the interviewees 'responses regarding their assessment of DevOps tools' success in delivering the required outcomes.

Table 4.7: Summary of issues related to DevOps tools

Research sub-question 3	Effectiveness of DevOps tools
To what extent are the DevOps tools used in organisations able to deliver the expected business results?	▪ General satisfaction with the DevOps tools like code management, deployment, repository, and monitoring tools. ▪ Flexibility in deployment times, instead of only deploying during nights and weekends, deployments could happen anytime during the day. ▪ Lack of required skills to use DevOps tools was a big concern.

4.2.4 Construct and Refine

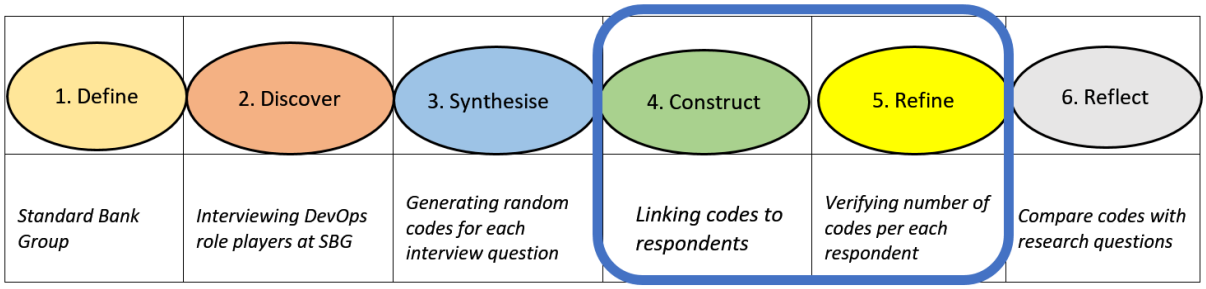


Figure 4.6 Results of the Construct and Refine steps of the case study methodology

The codes listed in the previous step are linked to the respondents and the frequency of each code is noted in the following tables per each sub-question.

4.2.4.1 Research sub-question 1: How should organisations restructure when implementing DevOps?

Table 4.8 shows information from the study data regarding the respondents who contributed to each DevOps restructuring suggestion and shows the total number of respondents per each DevOps restructuring suggestion. Respondent 1 has the highest number of contributions in terms of DevOps restructuring suggestions, whilst the other respondents fall behind with an equal number of contributions.

Table 4.8: Summary of study data for DevOps restructuring suggestions

Restructuring suggestion	Respondent number				Frequency
Cross-functional teams.	1	2		4	3
Understand your current business, its vision and mission.	1	2	3		3
Assessment of current technology usage and culture setup.	1	2	3	4	4
Obtain information about the skills structure of the organisation.	1	2	3		3
Do a DevOps impact assessment to find out how it will add value to the organisation.	1	2			2
Be knowledgeable about the state of your business's operating model.	1		3	4	3
Market the DevOps idea in the organisation to change the employees' mindsets.	1	2	3		3
Design DevOps maturity model based on DevOps best practices.	1	2		4	3
Identify pilot project with which to start testing DevOps practices.	1		3	4	3
Use DevOps metrics to measure your progress and keep on improving.	1	2	3	4	4
Ensure system security is part of every stage of DevOps implementation.	1		3		2
Use of dashboards.				4	1
Experimentation and learning fast.	1	2		4	3

4.2.4.2 Research sub-question 2: What challenges are encountered during restructuring and how can these challenges be mitigated?

Table 4.9 shows the number of each respondent who made a suggestion for each DevOps restructuring challenge. The minimum number of respondents who contributed for each restructuring challenge is two. Five of the restructuring challenges were mentioned by all four respondents, as shown by a frequency of four.

Table 4.9: Summary of study data for DevOps restructuring challenges

Restructuring challenge	Respondent number				Frequency
Manual intervention.	1	2	3		3
Legacy change management processes.	1		3	4	3
Silo mentality.	1	2	3	4	4
Unwillingness to learn new skills and processes.	1		3		2
Lack of executive management support.	1		3	4	3
Employees are not collocated.		2	3	4	3
High DevOps implementation success rate concentrated on pilot project and not across all projects.	1	2	3		3
Indecision when it comes to which part of the business to outsource during restructuring.		2	3		2
Dependence on individual brilliance and heroic efforts.	1	2	3	4	4
Automating waste, creating automation before fixing bottlenecks and system failure points.	1			4	4
Changing management employees.		2	3		2
Not taking IT governance seriously.	1	2	3		3
Lack of shared ownership and accountability due to silos.	1	2	3	4	4
Lack of operational maturity.	1	2	3		3
Use of different technologies and different configurations within the same environment.	1	2		4	3
Achieving a high maturity DevOps structure is an event not a process.	1			4	2
Outdated testing practices.		2	3	4	3
Collective accountability.			3	4	2

Table 4.10 displays the suggested solutions to the restructuring challenges. Respondent 1 suggested more solutions than any other respondent. The minimum number of times a solution was mentioned is by two respondents, while some solutions were suggested by all the respondents, as indicated by a frequency of four.

Table 4.10: Summary of study data for possible solutions to the restructuring challenges

Possible solution	Respondent number				Frequency
Automating the entire workflow to minimise human involvement.	1	2	3		3
Ensure there is accountability, and a realistic turnaround time for every approval process. Automate some of the approval processes.	1		3	4	3
Create cross-functional teams. Encourage required behaviour by giving incentives and rewards.	1	2	3	4	4
Effective communication to convince employees about the benefits of the changes to them. Ask human resources department to help fill skills gap.	1		3		2
Implement initiatives from the bottom and ensure everybody understands DevOps. Use metrics to measure successes and then use positive outcomes to justify to the executive the need for DevOps and their support.	1		3	4	3
Make use of online video conferences. Organise team building sessions to meet and talk (after Covid-19 is gone).		2	3	4	3
Set realistic, achievable, and timeous goals, and track them and share findings. Subscribe to uniform and best standards of development and deployment to ensure stability and repeatability. Create a DevOps manifesto based on past failures and successes and learn from it to improve. Achieve a higher degree of automation to minimise manual processes and eliminate any processes that do not add value. Ensure there are fast feedback loops within the deployment pipeline to fail fast and recover fast. Work as a team, respect each other.	1	2	3		3
Communicate and thoroughly consult with all business stakeholders to help make a good decision.		2	3		2

Possible solution	Respondent number				Frequency
Investigate the root cause of heroic efforts and fix the problem fast.	1	2	3	4	4
The first step is to identify bottlenecks and points of weaknesses and then use automation to solve them.	1			4	4
Embrace new ideas from new employees but ensure that the ideas are meant to improve the workflow in line with the vision of the organisation.		2	3		2
Establish governance to set required standards that control IT processes and make everyone accountable.	1	2	3		3
Continuously assess your operational processes, such as access management, incident management, request management and problem management, so that you can improve their agility and transparency.	1	2	3	4	4
Increase collaboration between different departments. Promote cross-skilling and market the benefits of doing that to the employees.	1	2	3		3
Standardise infrastructure technology to be identical and to subscribe to the best practices.	1	2		4	3
Create an environment that promotes continuous learning and improvement. Be patient, as a high velocity organisation cannot be achieved overnight and stick to basics.	1			4	2
Automate the testing. Make testing and quality assurance everybody's responsibility, let developers also test and testers also develop.		2	3	4	3

4.2.4.3 Research sub-question 3: To what extent are the DevOps tools used in organisations able to deliver the expected business results?

Table 4.11 shows that out of the three responses regarding the effectiveness of DevOps tools, only one response was mentioned by all the respondents. Respondent 3 made two suggestions, while Respondent 2 made only one suggestion.

Table 4.11: Summary of study data related to the effectiveness of DevOps tools

Effectiveness of DevOps tools	Respondent number				Frequency
Respondents were satisfied with the DevOps tools, such as code management, deployment, repository and monitoring tools.	1	2	3	4	4
Deployments could happen during the day instead of only during nights and weekends.	1			4	2
Lack of required skills to use DevOps tools was a big concern.	1		3	4	3

4.2.5 Reflect

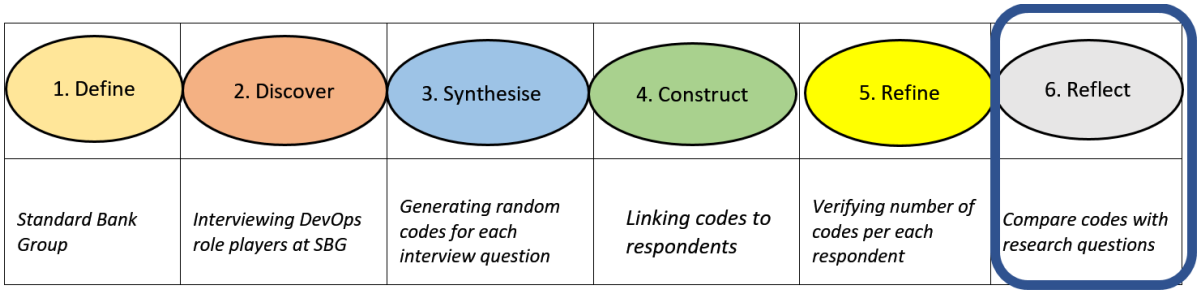


Figure 4.7 Results of the Reflect step of the case study methodology

This last step involved reviewing all the responses that the researcher obtained from the interviewees to ascertain their relevance to the research questions.

4.3 CONCLUDING SUMMARY

The research study identified the factors that must be considered when restructuring an organisation for DevOps by answering the main question of this study. These factors clarified the changes that should be part of an IT organisation's evolution during the process of DevOps implementation. The respondents outlined the challenges that are encountered during restructuring and provided possible solutions. Lastly the respondents expressed their satisfaction with the DevOps tools being used but there was a concern raised in terms of the shortage of skills to effectively use the tools. The consultation with the DevOps role players resulted in both existing and non-existing literature suggestions being raised as indicated for each theme. The next and final chapter provides the overall conclusion of the study.

CHAPTER 5:
CONCLUSION

5.1 INTRODUCTION

In this chapter the research questions are revisited, and the findings are summarised, the contributions and limitations are mentioned, and suggestions for future research is made.

5.2 REVISITING THE RESEARCH QUESTIONS

The main research question was:

How should organisations restructure during DevOps implementation to mitigate the related challenges and enhance the delivery of high-quality software to their customers?

This question was refined into three sub-questions:

1. **How should organisations restructure when implementing DevOps?**

   The findings from the research have shown that the restructuring of an organisation related to DevOps is a process that is done in steps. The steps suggested by the respondents are summarised in Figure 5.1.

   The diagram shows that the first step is to understand the organisation's business and the short- and long-term objectives that it wants to achieve. Once that is clear, then the second step is to determine how DevOps will add value to business processes and to become familiar with the DevOps practices that will need to be implemented. According to Ghantous (2017), an organisation needs to understand what DevOps is and the practices that are associated with it to be able to successfully adopt it. During this step, it is important that all the stakeholders of the organisation are convinced about the positive outcomes that DevOps may bring.

   The third step is to identify a small project within the organisation that can be used as a pilot project for DevOps implementation. Once the pilot project is successful, the next phase is to implement DevOps on a bigger project, and consequently, to all the projects in the organisation.

The final step is to measure the rate of success and to find ways to keep on improving. Measuring and monitoring enable the organisation to interact with the end-users, and to understand their product better, thereby helping them to make the correct decisions that are required to reinvent the system (Luz *et al.*, 2018).

During the implementation of DevOps, it is imperative that employees develop a collaborative culture to help them achieve cross-functional teams. This will ensure that all the employees become skilled in the use of automation technology (Leite *et al.*, 2019).

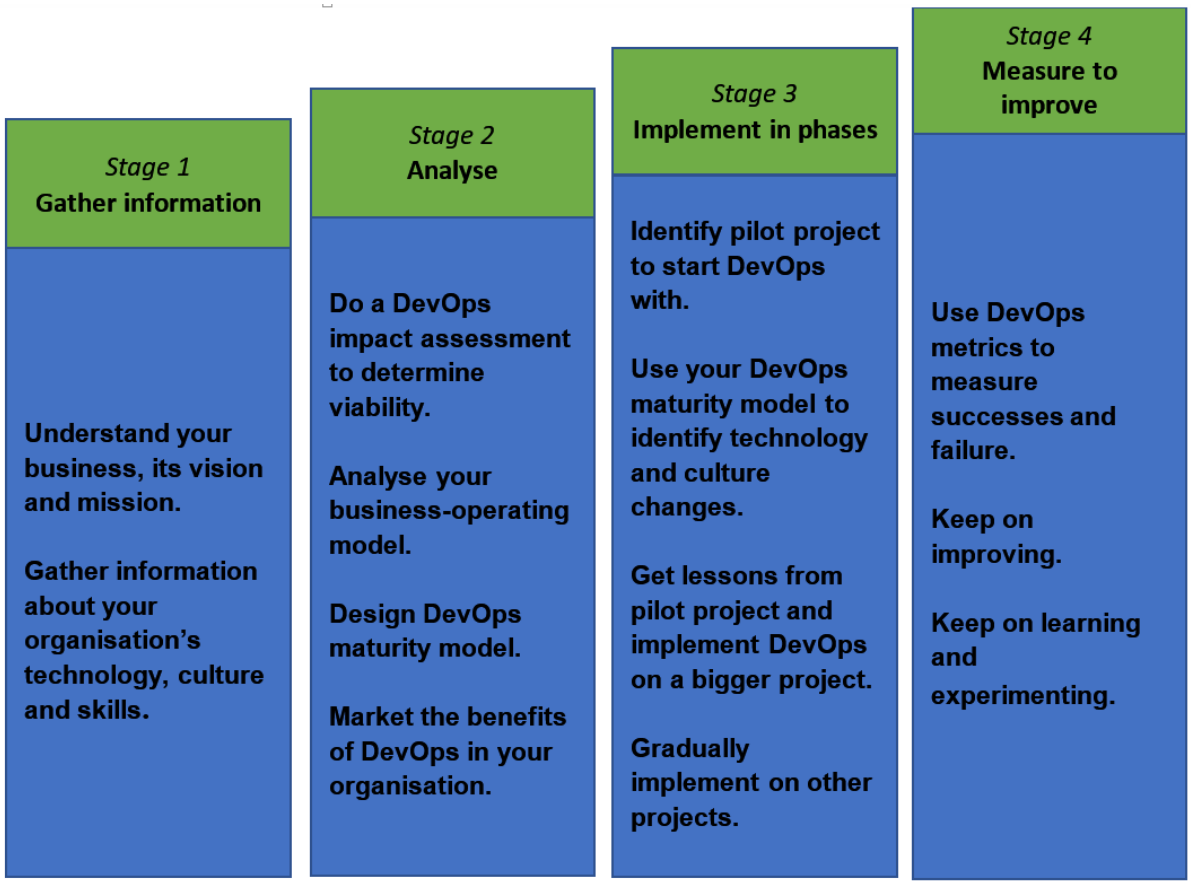


Figure 5.1: DevOps restructuring summary as suggested by respondents

2. What challenges are encountered during restructuring and how can these challenges be mitigated?

Both the literature and the research have shown that implementing DevOps comes with different challenges, mostly cultural and technological challenges. Kuusinen *et al.* (2018) noted that most challenges are caused by an organisation's unwillingness to embrace new changes. It is important that all employees feel

valued and motivated during the implementation of DevOps, as well as having the freedom to make decisions, as these help to improve productivity, and consequently, assist in making DevOps a success (Senapathi *et al.*, 2018).

3. **To what extent are the DevOps tools used in organisations able to deliver the expected business results?**

   The research indicated that DevOps tools help to improve productivity. An important consideration when buying DevOps tools is that they should be the right tools that the organisation needs to help achieve its goals (Riungu-Kalliosaari *et al.*, 2016). According to the respondents, an organisation must invest in employee training to ensure effectiveness and efficiency in using DevOps tools.

5.3 SUMMARY OF RESEARCH FINDINGS

The analysis of the research on the impact of DevOps on IT organisations has made it clear that there are two main changes that occur in organisations during the implementation of DevOps, namely, cultural and technological changes. However, the cultural aspect, which involves changing employees' mindsets is the first critical step that needs to be changed to align with DevOps transformation, otherwise the technology will not bring about the expected success.

Both the literature review and field research showed that the impact of DevOps on IT organisations occurs in three forms. The first is workflow optimisation that involves the changes taking place within the organisation to ensure that DevOps practices help to increase productivity. The second form is fast feedback loops. The entire organisation has to be restructured to enable effective communication to be prioritised in every action and in every process. The main benefits of this are the ability to fail fast and recover fast, and also to do as the customer wants. Lastly, but not least, is a culture of continuous learning and experimentation. The key to becoming leaders in DevOps is to have passionate employees who are willing to keep on learning new technology and to experiment with the existing technology. This results in the reinvention of the wheel and innovations, which are both primary drivers of organisation growth and increased market share.

The expectation for this research is that both the industry and academic research communities will benefit from the findings of this study in terms of the influence

DevOps has on IT organisations. The literature review and field research in their different ways have managed to contribute to the key factors that are part of organisational evolution during the implementation of DevOps.

5.4 CONTRIBUTION TO KNOWLEDGE

The main contribution of this research has been to provide an analysis of the transformation that IT organisations go through when implementing DevOps. The literature addressed the need to regard DevOps practices as part of restructuring during the adoption of DevOps. It also highlighted the importance of changing culture when implementing DevOps and discussed the challenges that are encountered during the process of making changes.

Although the field research raised some issues similar to those mentioned in the literature, it did come up with some new points regarding organisational restructuring during the implementation of DevOps, and the challenges encountered. Chapter 4 discussed the insights that emerged after consultation with DevOps role players. The research findings that were presented in Chapter 4 are important to help IT organisations understand the changes that they are likely to experience when adopting DevOps.

Based on the facts raised from literature and the field research, the researcher believes that this study provides both the practitioner and academic research communities with a novel way of grasping the impact that DevOps has on IT organisations, and consequently, creates new opportunities for future research.

5.5 LIMITATION AND FUTURE RESEARCH SUGGESTION

The research was conducted using a single case study. Although the unit of analysis operates in many countries, the findings from the South African point of view may not be a representation of other countries or other continents.

Another limitation is that the research was focused only on determining the impact that DevOps has on IT organisations. After this research experience, the researcher has realised that DevOps may affect customers in more ways that may need to be explored and shared.

Based on these limitations, the suggestion for future research is that it should have a much larger sample spread over different geographical locations. The scope of the research should be expanded to determine the impact that DevOps has on the customers who directly benefit from companies that have implemented DevOps. In addition, future research should consider the use of quantitative methods of research to improve the credibility and reliability of the findings.

5.6 OVERALL CONCLUSION

DevOps provides important technological and cultural changes that can help IT organisations to become innovative and competitive. The current study found that the most successful IT organisations subscribe to DevOps practices. The future of digital transformation is indeed DevOps, and the more IT organisations embrace this change, the more the future of technology will be life-changing and exciting.

LIST OF REFERENCES

Adams, W. 2015. Conducting semi-structured interviews. In K.E. Newcomer, H.P. Hatry & J.S. Wholey (Eds.), *Handbook of practical program evaluation,* pp. 492-505*.* 4th ed. Hoboken, NJ: Wiley.

Aguirre, D., von Pos, R. & Alpern, M. 2013. *Culture's Role in Enabling Organisational Change. Survey Ties Transformation Success to Deft Handling of Cultural Issues.* Booz & Co.

Akshaya, H., Vidya, J. & Veena, K. 2015. A basic introduction to DevOps tools. *International Journal of Computer Science & Information Technologies,* 6(3): 2349-2353.

Andrade, A. 2009. Interpretive research aiming at theory building: Adopting and adapting the case study design. *The Qualitative Report,* 14(1): 42-60.

Baker, L. 2006. Observation: A complex research method. *Library Trends,* 55(1): 171-189.

Blanton, J.E., Watson, H.J. and Moody, J., 1992. Toward a better understanding of information technology organization: a comparative case study. MIS quarterly, pp.531-555.

Banica, L., Radulescu, M., Rosca, D. & Hagiu, A. 2017. Is DevOps another Project Management Methodology? *Informatica Economica,* 21(3): 39-50.

Baskarada, S. 2014. Qualitative case studies guidelines. *The Qualitative Report,* 19(40): 1-25.

Bass, L. 2017. The software architect and DevOps. *IEEE Software*, 8-10.

Battina, D. 2021. The Challenges and Mitigation Strategies of Using DevOps during Software Development. *International Journal of Creative Research Thoughts,* 9(1): 2320-2882.

Baxter, P. & Jack, S. 2008. Qualitative case study methodology: Study design and implementation for novice researchers. *The Qualitative Report,* 13(4): 544-559.

Bezemer, C., Eismann, C., Ferme, V., Grohmann, J., Heinrich, R., Jamshidi, P., ... Willnecker, F. 2019. How is performance addressed in DevOps? *In Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering*, pp. 45-50.

Braun, V., & Clarke, V. (2012). Thematic analysis. In H. Cooper, P.M. Camic, D.L. Long, A.T. Panter, D. Rindskopf, & K.J. Sher (Eds.), *APA handbook of research methods in psychology, Vol. 2. Research designs: Quantitative, qualitative, neuropsychological, and biological,* pp. 57–71. American Psychological Association. Doi 10.1037/13620-004

Brunnert, A., Van Hoorn, A., Willnecker, F., Danciu, A., Hasselbring, W., Heger, C., ... Koziolek, A. 2015. Performance-oriented DevOps: A research agenda. *arXiv.* SPEC RG DevOps Performance Working Group.

Bucena, I. & Kirikova, M. 2017. *Simplifying the DevOps Adoption Process*. BIR Workshops.

Cameron, E. & Green, M. 2019. *Making sense of change management: A complete guide to the models, tools and techniques of organisational change.* 2nd ed. Kogan Page.

Capizzi, A., Distefano, S., Araújo, L., Mazzara, M., Ahmad, M. & Bobrov, E. 2019. Anomaly detection in DevOps toolchain. *In International Workshop on Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment*, pp. 37-51.

Caprarelli, A., Nitto, E. & Tamburri, D. 2019. Fallacies and pitfalls on the road to devops: A longitudinal industrial study. *In International Workshop on Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment*, pp. 200-210.

Cartlidge, A., Hanna, A., Rudd, C., Macfarlane, I., Windebank, J. & Rance, S. 2007. *An introductory overview of ITIL V3.* UK: The UK Chapter of the itSMF.

Casagni, M., Lead, M., Hanf, D., Area, C., Lead, I., Malhotra, P., … Karmel, A. 2018. Federal DevOps Summit, Pp. 1-29.

Castleberry, A. & Nolen, A. 2018. Thematic analysis of qualitative research data: is it as easy as it sounds? *Currents in pharmacy teaching and learning,* 10(6): 807-815.

Chen, L. 2018. Microservices: architecting for continuous delivery and DevOps. *In 2018 IEEE International conference on software architecture (ICSA)*, pp. 39-397.

Claps, G., Svensson, R. & Aurum, A. 2015. On the journey to continuous deployment: Technical and social challenges along the way. *Information and Software Technology,* 57: 21-31.

Crotty, M. 1998. *Foundations of Social Research: Meaning and Perspective in the Research Process.* London: SAGE.

Crowe, S., Cresswell, K., Robertson, A., Huby, G., Avery, A. & Sheikh, A. 2011. The case study approach. *BMC medical research methodology,* 11(1): 1-9.

Crowley, C., McQuillan, L. & O'Brien, C. 2018. Understanding DevOps: Exploring the origins, composition, merits, and perils of a DevOps Capability. Exploring the origins, composition, merits, and perils of a DevOps Capability. In *Proceedings of the 4th International Conference on Production Economics and Project Evaluation*, pp. 29-37.

Darke, P., Shanks, G. & Broadbent, M. 1998. Successfully completing case study research: combining rigour, relevance and pragmatism. *Information Systems Journal,* 8(4): 273-289.

Davis, J. and Daniels, R. 2016. *Effective DevOps: building a culture of collaboration, affinity, and tooling at scale*. " O'Reilly Media, Inc.".

De França, B., Jeronimo, H. & Travassos, G. 2016. Characterizing DevOps by hearing multiple voices. In *Proceedings of the 30th Brazilian symposium on software engineering*, pp. 53-62.

DeMartine, A., Oehrlich, E. & Doerr, M. 2015. *CALMSS: a model for assessing modern service delivery.* Research report, Forrester Research, pp. 1-13.

Díaz, J., López-Fernández, D., Pérez, J. & González-Prieto, A. 2021. Why are many businesses instilling a DevOps culture into their organisation? *Empirical Software Engineering,* 26(2).

Díaz, J., Perez, J., Yague, A., Villegas, A. & De Antona, A. 2019. Devops in practice–a preliminary analysis of two multinational companies. *International Conference on Product-Focused Software Process Improvement*, pp. 1-8.

Diel, E., Marczak, S. & Cruzes, D. 2016. Communication challenges and strategies in distributed DevOps. *IEEE 11th International Conference on Global Software Engineering*, pp. 24-28.

Ebert, C., Gallardo, G., Hernantes, J. & Serrano, N. 2016. DevOps. *Ieee Software,* 33(3): 94-100.

Erich, F. 2018. DevOps is simply interaction between development and operations. *International Workshop on Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment*, pp. 89-99.

Erich, F., Amrit, C. & Daneva, M. 2017. A qualitative study of DevOps usage in practice. Journal of software. *Journal of Software: Evolution and Process,* 29(6): 1-47.

Feijter, R., Vliet, R., Jagroep, E., Overbeek, S. & Brinkkemper, S. 2017. Towards the adoption of DevOps in software product organizations: A Maturity Model Approach. *Technical Report Series*, (UU-CS-2017-009).

Fitzgerald, B. & Stol, K.J. 2017. Continuous software engineering: A roadmap and agenda. *Journal of Systems and Software*, *123*, pp.176-189.

Gall, M. & Pigni, F. 2021. Taking DevOps mainstream: a critical review and conceptual framework. *European Journal of Information Systems:* 1-20.

Ghantous, B. & Gill, A. 2017. DevOps: Concepts, practices, tools, benefits and challenges. *Pacific Asia Conference on Information Systems*, pp. 1-12.

Gill, A., Loumish, A., Riyat, I. & Han, S. 2018. DevOps for information management systems. *VINE Journal of Information and Knowledge Management Systems,* 48(1): 122-139.

Gokarna, M. & Singh, R. 2021. DevOps: a historical review and future works. *International Conference on Computing, Communication, and Intelligent Systems (ICCCIS).* Pp. 366-371.

Graue, C. 2015. Qualitative data analysis. *International Journal of Sales, Retailing & Marketing,* 4(9): 5-15.

Halkias, D. & Neubert, M. 2020. Extension of Theory in Leadership and Management Studies Using the Multiple Case Study Design. *International Leadership Journal,* 12(2): 48-73.

Harrell, M. & Bradley, M. 2009. *Data collection methods. Semi-structured interviews and focus groups*. Rand Corporation, Santa Monica. CA.

Hemon, A., Lyonnet, B., Rowe, F. & Fitzgerald, B. 2020. From agile to DevOps: Smart skills and collaborations. *Information Systems Frontiers,* 22(4): 927-945.

Humble, J. & Farley, D. 2010. *Continuous Delivery: Reliable Software Releases through Build, Test and Deployment Automation.* Boston: Pearson Education.

Humble, J. & Molesky, J. 2011. Devops: A software revolution in the making. *Cutter IT Journa,* 24(8): 6-12.

Huttermann. 2012. *DevOps for developers.* New York: Apress.

Jabbari, R., Bin Ali, N., Petersen, K. & Tanveer, B. 2018. Towards a benefits dependency network for DevOps based on a systematic literature review. *Journal of Software: Evolution and Process,* 30(11): e1957.

Joby, P. 2019. Exploring devops: challenges and benefits. *Journal of Information Technology,* 1(1): 27-37.

Jones, S., Noppen, J. & Lettice, F. 2016. Management challenges for DevOps adoption within UK SMEs. *Proceedings of the 2nd International Workshop on quality-aware devops*, pp. 7-11.

Karamitsos, I., Albarhami, S. & Apostolopoulos, C. 2020. Applying DevOps practices of continuous automation for machine learning. *Information*, 11: 1-15. DOI: 10.3390/info11070363

Katzenbach, J., Oelschlegel, C. & Thomas, J. 2016. *10 principles of organisational culture*. strategy+business [Online} Available fom: https://www.strategy-business.com/issue82-spring2016 [Accessed 16 October 16, 2021].

Kaur, G. & Bahl, K. 2014. Software reliability, metrics, reliability improvement using agile process. *International Journal of Innovative Science, Engineering & Technology,* 1(3): 143-147.

Khan, M., Khan, A., Khan, F., Khan, M. & Whangbo, T. 2022. Critical Challenges to Adopt DevOps Culture in Software Organisations: A Systematic Review. *IEEE Access,* 10: 14339-14349.

Khan, M., Shaikh, A. & Farhan, W. 2020. Fast delivery, continuously build, testing and deployment with DevOps pipeline techniques on cloud. *Indian Journal of Science and Technology,* 13(5): 552-575.

Kim, G. 2014. *Top 11 Things You Need to Know About DevOps*. [Online Available from: https://www.thinkhdi.com/~/media/HDICorp/Files/White-Papers/whtppr-1112-devops-kim.pdf [Accessed 8 June 2020].

Klein, H. & Myers, M. 2001. A classification scheme for interpretive research in information systems. In E.M. Trauth (Ed.), *Qualitative research in IS: issues and trends*, pp. 218-239. Oakland University: Idea Group Publishing.

Kuusinen, K., Balakumar, V., Jepsen, S., Larsen, S., Lemqvist, T., Muric, A., … Vestergaard, O. 2018. A large agile organisation on its journey towards DevOps. *44th Euromicro Conference on Software Engineering and Advanced Applications*, pp. 60-63.

Laukkanen, E., Itkonen, J. & Lassenius, C. 2017. Problems, causes and solutions when adopting continuous delivery—A systematic literature review. *Information and Software Technology,* 82: 55-79.

Leedy, P. & Ormrod, J. 2005. *Practical research*. Saddle River: Prentice Hall.

Lehtonen, T., Suonsyrja, S., Kilamo, T. & Mikkonen, T. 2015. Defining metrics for continuous delivery and deployment pipeline. *SPLST:* 16-30.

Leite, L., Pinto, G., Kon, F. & Meirelles, P. 2021. The organisation of software teams in the quest for continuous delivery: A grounded theory approach. *Information and Software Technology,* 139: 1-19.

Leite, L., Rocha, C., Kon, F., Milojicic, D. & Meirelles, P. 2019. A Survey of DevOps Concepts and Challenges. *ACM Computing Surveys,* 52(6): 1-35.

Lochmiller, C. 2021. Conducting Thematic Analysis with Qualitative Data. *The Qualitative Report,* 26(6): 2028-2043.

López-Fernández, D., Diaz, J., Garcia-Martin, J., Pérez, J. & Gonzalez-Prieto, A. 2021. DevOps Team Structures: Characterization and Implications. *IEEE Transactions on Software Engineering,* 14(8): 1-18.

Luz, W., Pinto, G. & Bonifácio, R. 2018. Building a collaborative culture: a grounded theory of well succeeded DevOps adoption in practice. *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, pp. 1-10.

Luz, W., Pinto, G. & Bonifácio, R. 2019. Adopting DevOps in the real world: A theory, a model, and a case study. *Journal of Systems and Software,* 157: 1-35.

Lwakatare, L., Karvonen, T., Sauvola, T., Kuvaja, P., Olsson, H., Bosch, J. & Oivo, M. 2016. Towards DevOps in the embedded systems domain: Why is it so hard? *49th Hawaii International Conference on System Sciences*, pp. 5437-5446.

Lwakatare, L., Kilamo, T., Karvonen, T., Sauvola, T., Heikkilä, V., Itkonen, J., …Lassenius, C. 2019. DevOps in practice: A multiple case study of five companies. *Information and Software Technology,* 114: 1-18.

Lwakatare, L., Kuvaja, P. & Oivo, M. 2015. Dimensions of devops. *International conference on agile software development*, pp. 212-217. Springer.

Lwakatare, L., Kuvaja, P. & Oivo, M. 2016. An exploratory study of devops extending the dimensions of devops with practices. *ICSEA*, pp. 91-341.

Mangot, D. 2016. *Mastering DevOps*. [Online] Available from: https://learning.oreilly.com/videos/masteringdevops/9781786468048/9781786468048-video1_3 [Accessed 4 June 2020].

Marnewick, C. and Langerman, J. 2020. DevOps and organizational performance: The fallacy of chasing maturity. IEEE Software, 38(5), pp.48-55.

Maroukian, K. and Gulliver, S.R. 2020. The link between transformational and servant leadership in DevOps-oriented organizations. In *Proceedings of the 2020 European Symposium on Software Engineering* (pp. 21-29).

Mishra, A. & Otaiwi, Z. 2020. DevOps and software quality: A systematic mapping. *Computer Science Review,* 38: 1-13.

Mohammad, S. 2016. Continuous Integration and Automation. *International Journal of Creative Research Thoughts,* 4(3): 938-944.

Muñoz, M. & Rodríguez, M.N. 2021. A guidance to implement or reinforce a DevOps approach in organizations: A case study. *Journal of Software: Evolution and Process*, p.e2342.

Nagios. n.d. *What can Nagios help you do?* Nagios [Online] Available from: https://www.nagios.org/ [Accessed 18 April 2022].

Nandhakuma, J. & Jones, M. 1997. Too close for comfort? Distance and engagement in interpretive information systems research. *Information Systems Journal, 7*(2): 109-131.

Noor, K. 2008. Case study: A strategic research methodology. *American Journal of Applied Sciences,* 5(11): 1602-1604.

Orlikowski, W. & Baroudi, J. 1991. Studying information technology in organisations: Research approaches and assumptions. *Information Systems Research,* 2(1): 1-28.

Perera, P., Silva, R. & Perera, I. 2017. Improve software quality through practicing DevOps. *Seventeenth International Conference on Advances in ICT for Emerging Regions*, pp. 14-18.

Pylayeva, D. 2017. *Introduction to DevOps with Chocolate, LEGO and Scrum Game.* New York: Apress.

Rahman, M. 2020. The advantages and disadvantages of using qualitative and quantitative approaches and methods in language “testing and assessment” research: A literature review. *Journal of Education and Learning,* 6(1): 102-112.

Ravichandran, A., Taylor, k. & Waterhouse, P. 2016. *Devops for digital leaders: Reignite business with a modern devops-enabled software factory*. Springer Nature.

Riungu-Kalliosaari, L., Mäkinen, S., Lwakatare, L. & Männistö, T. 2016. DevOps adoption benefits and challenges in practice: a case study. *International conference on product-focused software process improvement*, pp. 590-597.

Rowley, J. 2002. Using case studies in research. *Management Research News,* 25(1): 16-27.

Rowse, M. & Cohen, J. 2021. A Survey of DevOps in the South African Software Context. *Proceedings of the 54th Hawaii International Conference on System Sciences*, pp. 6785-6794.

Rütz, M. 2019. *Devops: A systematic literature review*. IT Management Seminar Paper Summer Term, pp. 1-11.

Sánchez-Gordón, M. & Colomo-Palacios, R. 2018. Characterizing DevOps culture: a systematic literature review. *International Conference on Software Process Improvement and Capability Determination*, pp. 1-13.

SBG (Standard Bank Group). 2016. *Group marketing and communication strategy.* Standard Bank Group, Johannesburg.

SBG. 2020. *Annual integrated report*. [Online] Available from: https://thevault.exchange/?get_group_doc=18/1615894015-SBG2020AnnualFinancialStatements.pdf [Accessed 14 October 2021].

SBG. 2021. *Standard Bank Group Annual Financial Statement*. [Online] Available from: https://thevault.exchange/?get_group_doc=18/1617173063-SBG2020AnnualIntegrated Report.pdf [Accessed 14 October 2021].

Senapathi, M., Buchan, J. & Osman, H. 2018. DevOps capabilities, practices, and challenges: Insights from a case study. *Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018*, pp. 1-11.

Shahin, M. & Babar, M. 2020. On the role of software architecture in DevOps transformation: An industrial case study. *Proceedings of the International Conference on Software and System Processes*, pp. 175-184.

Shahin, M., Babar, M., Zahedi, M. & Zhu, L. 2017. Beyond continuous delivery: an empirical investigation of continuous deployment challenges. *2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, pp. 111-120.

Sharma, S. 2013. *What exactly is DevOps?* [Online] Available from: http://www.drdobbs.com/architecture-and-design/what-exactly-is-devops/240009147 [Accessed 10 June 2020].

Sharma, S. 2017. *The DevOps Adoption Playbook: A Guide to Adopting DevOps in a Multi-Speed IT Enterprise.* Hoboken, NJ: Wiley.

Shrikanth, G. 2017. Decoding DevOps: The escalation of DevOps as a software development hygiene practice is all set to alter the way IT is being developed and delivered. DATAQUEST, A CyberMedia Publication, India. Pp. 15-19.

Shrikanth, G. 2018. A Panacea called DevOps: A radical intervention is needed to uplift the ailing current software development practices and why is DevOps being touted as the key to better software and touted as the panacea. *Intelligent Computing,* 26(1): 26-29.

Smeds, J., Nybom, K. & Porres, I. 2015. DevOps: a definition and perceived adoption impediments. *International Conference on Agile Software Development*, pp. 166-177.

Teixeira, D., Pereira, R., Henriques, T., Silva, M. & Faustino, J. 2020. A systematic literature review on DevOps capabilities and areas. *International Journal of Human Capital and Information Technology Professionals,* 11(3): 1-22.

Vaismoradi, M., Jones, J., Turunen, H. & Snelgrove, S. 2016. Theme development in qualitative content analysis and thematic analysis. *Journal of Nursing Education and Practice,* 6(5): 100-110.

Van Belzen, M., DeKruiff, D. & Trienekens, J. 2019. Success factors of collaboration in the context of DevOps. *International Conference Information System*, pp. 26-30.

Vanderjack, B. 2019. What is the difference between DevOps and scrum? *PM World Journal,* 8(6): 1-7.

Walsham, G. 1995. Interpretive case studies in IS research: nature and method. *European Journal of Information Systems,* 4(2): 74-81.

Wettinger, J., Breitenbücher, U. & Leymann, F. 2014. Devopslang–bridging the gap between development and operations. *European Conference on Service-Oriented and Cloud Computing*, pp. 108-122.

Whiting, L. 2008. Semi-structured interviews: guidance for novice researchers. *Nursing Standard,* 22(23): 35-40.

Wilde, N., Eddy, B., Patel, K., Cooper, N., Gamboa, V., Mishra, B. & Shah, K. 2016. Security for Devops Deployment Processes: Defenses Risks Research Directions. *International Journal of Software Engineering & Applications,* 7(6): 1-16.

Williams, C. 2007. Research methods. *Journal of Business & Economics Research,* 5(3): 65-72.

Wurster, L., Colville, R., Haight, C., Tripathi, S. & Rastogi, A. 2013. *Emerging technology analysis: DevOps a culture shift not a technology*. Pp. 1-12.

Yarlagadda, R. 2021. DevOps and Its Practices. *International Journal of Creative Research Thoughts,* 9(3): 2320-2882.

Yin, R.K. 2018. *Case study research and applications.* 6th ed. Thousand Oaks, CA: SAGE Publications.

Zainal, Z. 2007. Case study as a research method. *Jurnal Kemanusiaan,* 5(1): 2-6.

Zhu, L., Bass, L. & Champlin-Scharff, G. 2016. DevOps and its practices. *IEEE Software,* 33(3): 32-34.

Zimmerman, J., Evenson, S. & Forlizzi, J. 2004. Discovering and extracting knowledge in the design project. *Conference of the Design Research Society Proceedings, Melbourne, Australia, November 2004*, pp. 1-13.

APPENDIX A:
ETHICAL CLEARANCE CERTIFICATE

Ethics clearance

Ethics Clearance.pdf

APPENDIX B:
COMMUNICATION WITH RESPONDENTS

Respondents related emails


Accepted_ DevOps Discussion_Thabiso


Accepted_ DevOps Research Discussion


Accepted_ DevOps Research Discussion


FW_ DevOps Research Discussion


DevOps Research Discussion_Renita.m


DevOps Research Discussion(Standarc


Accepted_ DevOps Research Discussion


Accepted_ DevOps Research Discussion


RE_ DevOps Research Discussion


RE_ DevOps Research Discussion


Standard Bank DevOps Research Di


RE_ DevOps Research Discussion

APPENDIX C:
DECLARATION OF PROFESSIONAL EDIT

Retha Burger
B.A.(H.E.D.)
tel : 012 807 3864
cell : 083 653 5255
fax : 012 807 3864
e-mail : retha@skillnet.co.za
Independent Skills Development Facilitator

Dear Mr Mudadi

This letter is to record that I have completed a language edit of your dissertation entitled, "A critical analysis of the impact of DevOps on information technology organisations: a case study".

The edit that I carried out included the following:

-Spelling
-Grammar
-Vocabulary
-Punctuation
-Pronoun matches
-Word usage
-Sentence structure
-Correct acronyms (matching your supplied list)
-Captions and labels for figures and tables
-Spot checking of 10 references

The edit that I carried out excluded the following:

-Content
-Correctness or truth of information (unless obvious)
-Correctness/spelling of specific technical terms and words (unless obvious)
-Correctness/spelling of unfamiliar names and proper nouns (unless obvious)
-Correctness of specific formulae or symbols, or illustrations

Yours sincerely

Retha Burger

14 September 2022