|NST|TUTO
UN|VERS|TÁR|O
DE L|SBOA

Team Management Strategies for DevOps

Rui Maurício da Cunha Ferreira

Master's Degree in Computer Science and Business Management

Supervisor:
PhD. Rúben Filipe de Sousa Pereira, Auxiliar Professor,
ISCTE-IUL

Co-supervisor:
PhD. Ana Margarida Soares Lopes Passos, Associated Professor,
Universidade Europeia

October, 2021

TECNOLOGIAS
E ARQUITETURA

Department of Information Science and Technology

Team Management Strategies for DevOps

Rui Maurício da Cunha Ferreira

Master's Degree in Computer Science and Business Management

Supervisor:
PhD. Rúben Filipe de Sousa Pereira, Auxiliar Professor,
ISCTE-IUL

Co-supervisor:
PhD. Ana Margarida Soares Lopes Passos, Associated Professor,
Universidade Europeia

October, 2021

Acknowledgment

This study and all the process behind it would not be possible without the great support of my supervisor, Ruben Pereira. His agile process consistently guided this study with the real notion of working towards the research goal's deadline. His knowledge and scientific discipline have been an inspiration. His availability to help with meetings and contacts for interviews were facilitators of this study. Ana Passos, my co-supervisor, has been another important guide to better understand teams, conflicts and managing, always with new perspectives. BNP Personal Finance Sucursal em Portugal, for investing in my career and financially supporting this study in an indirect way. ISCTE-IUL, for providing resources and frameworks to facilitate the research with access to online libraries. All the anonymous interviewees for sharing their experiences and information and for giving the most valuable resource: time. Grateful to Emma for the document review and improvements suggested. My love, Amelia, has been a companion who encouraged and took care of me during all the process, focusing me on the essentials. And finally, to my family, for making me resilient.

Resumo

Num mercado cada vez mais digital e onde o tempo de mercado é cada vez mais curto, a qualidade e fiabilidade mais relevantes, é imperativo que as equipas de desenvolvimento de software consigam organizar-se de modo a proporcionar uma resposta rápida no mercado e cada vez mais fiável.

A filosofia DevOps pretende terminar com a existência de silos (Desenvolvimento e Operações) e agilizar a produção de software, diminuindo desperdício e dificuldades na sua construção, aumentando a produtividade e desenvolver produtos melhores com foco na satisfação do cliente.

Contudo, a junção de equipas em torno de um mesmo objetivo acarreta desafios cruciais para a gestão, nomeadamente a gestão de conflitos e da informação entre as equipas. A forma como estes desafios são geridos poderá interferir no sucesso da implementação de uma filosofia DevOps.

Através de um Caso de Estudo, o objetivo desta pesquisa é o levantamento das melhores estratégias de gestão de equipas que ajudem a reduzir o surgimento de conflitos e potenciar a partilha de informação em contexto de implementação da filosofia DevOps, aumentando a eficácia destas equipas.

Como resultado, esta pesquisa traz algumas estratégias que podem facilitar a gestão de equipas DevOps e reforça a importância de fazer uma boa gestão dos conflitos, tarefas, processos e da informação.

Palavras-Chave: DevOps, Implementação DevOps, Gestão, Estratégias de Gestão de Equipas, Eficácia

Abstract

In an increasingly digital market, and where the time to market is shorter and the quality and reliability more relevant, it is imperative that software development teams can organize themselves in order to provide a faster reaction to the market with more reliability.

DevOps intends to eliminate the existence of silos (Development and Operations) and streamline the software production, declining waste and difficulties in its construction, increasing productivity and developing better products with a focus on client satisfaction.

Nevertheless, the joining of teams around the same goal causes key managing challenges, namely the management of conflicts and information sharing between teams. The way that these challenges are managed can interfere with the successful implementation of DevOps philosophy.

Though a Case Study, the research goal is to study the best team management strategies that help to reduce the appearance of conflicts and enhance information sharing in the context of DevOps implementation, increasing effectiveness in those teams.

As a result, this research brings some strategies to facilitate the DevOps team management and reinforces the importance of managing conflicts, processes, tasks and information well.

Keywords: DevOps, DevOps Implementation, Management, Team Management Strategies, Effectiveness

Index

Index of Tables

Index of Figures

Glossary of Abbreviations and Acronyms

CD	-	Continuous Delivering
CI	-	Continuous Integration
CM	-	Continuous Monitoring
DevOps	-	Development and Operations teams
IT	-	Information Technology
QA	-	Quality Team
SLR	-	Systematic Literature Review

Chapter 1 – Introduction

Information technology (IT) is present in our lives more and more, with impacts on economy and quality of life across society (Betz et al., 2016) and many organizations have benefits such as: coordination, reactivity, better efficiency and more competitivity (Gu et al., 2020). Recent methods, like Agile, are changing the IT and delivery approach. As a movement, Agile expanded to different related fields (Betz et al., 2016). Based on this Agile movement, a new way of working arose, known as DevOps (combination of Development and Operations teams) (Balalaie et al., 2016) where Agile methods form the basis (Hemon et al., 2020).

DevOps intends to decrease the time to make a change in a system and the time to promote that change to the production environment (Balalaie et al., 2016; Betz et al., 2016) and set up an environment and culture where developing, testing and releasing software can be rapid, frequent and more reliable (Lu et al., 2019). DevOps appeared as a solution to solve the old known bottleneck between Development and Operations teams, using a faster and automated manner. Automation has become the essential and tight way to an efficient collaboration between these teams (Hemon et al., 2020).

Nowadays, in scenarios with separated teams of Development and Operations, there is time pressure and this often causes conflicts between these teams. The reasons provided are quick deployments, lack of communication, and sometimes, different reaction times among the teams when the system crashes. Also, there is an isolation problem that originates from a lack of interdependence among both teams, usually because each team works as an independent silo (Wahaballa et al., 2015) and each team should produce their documentation which is understandable by both sides (Erich et al., 2014). Plus, sometimes Operation teams have to take a lot of *ad hoc* decisions, instead of Development teams, that are not confronted with this fast decision-making process (Wiedemann & Wiesche, 2018). All these issues can compromise team effectiveness.

Mathieu et al. (2019) considered team effectiveness in terms of: a) tangible outputs (productivity, efficiency and quality), and b) influences on team members (including shared experiences, such as cohesion or psychological safety, etc.). While IT teams and processes must be designed to maximize the tangible outputs, both productivity and quality can be decreased as communication is reduced (Cois et al., 2015). That way, DevOps philosophy, with its knowledge and communication sharing and automation

process, is recommended to increase overall software project productivity (Cois et al., 2015).

In many organizations, the use of IT creates a typical structural division in their software departments. The division between Development and Operations is often repeated as a pattern. Lately, many discussions have appeared to debate if this division is justified. That discussion is centred around the concept DevOps, which needs further research in many areas (Mishra & Otaiwi, 2020).

In modern teams, there are two critical aspects that require more attention: conflicts "us versus them" and incomplete information (Haas & Mortensen, 2016). Also, there is a concept named "teaming" when two or more teams are joined with the same goals, where team members need to work together, with different skills and disciplines and with their own language, norms, rules and knowledge. These characteristics can create disagreements, conflicts and make members group themselves with their already known group members, without sharing information (Amy C. Edmonson, 2012). In DevOps, Development and Operations teams work together to achieve the same goals.

However, there is a lack of research in literature on how DevOps teams are managed and this lack arouses some doubts, for example: Will the bias of "us versus them" continue to exist? And regarding the problem of information sharing presented by Cois et al. (2015), will it continue to exist when the teams are merged in a DevOps application?

The motivation for this research originated from these questions. It is necessary to understand the obstacles and facilitators of the function of DevOps teams. For this, the study will be based on important mediators from literature that are known as the most impactful on team effectiveness: conflicts (Mathieu et al., 2019), information sharing (Mathieu et al., 2019), (Amy C. Edmonson, 2012) and "us versus them" conflicts (Haas & Mortensen, 2016). Therefore, this research aims to explore and identify how these important mediators must be managed on DevOps teams to promote team effectiveness.

The remaining parts of the document are organized with a *Theoretical Background* section, where the terms DevOps and Team Management are bounded. Through a Systematic Literature Review (SLR), the current state of the art is presented in the *Related Work* section. Next, the *Methodology* section explains all the steps used to support the research. The findings and their analysis are demonstrated in the *Reporting the Findings*

section. Finally, the *Conclusion* has the important notes of this study and the recommendations.

Chapter 2 – Theoretical Background

2.1 DevOps

Usually in IT organizations, there are at least two teams: Development and Operations teams. These teams are usually structured in different positions in the organizations and are, often, geographically distant (Hussaini, 2014).

During the last years, literature has identified some problems and conflicts between Development and Operations teams: poor collaboration, Operations team involvement too late in project cycle, and differences in culture, co-operation or collaboration (Hussaini, 2014; Lwakatare et al., 2019). Often, Development and Operations teams have communication problems (Hussaini, 2014), generally do not help each other and have different time reactions to crashes (Wahaballa et al., 2015).

As a response to this, a new paradigm emerged called DevOps that means Development (Dev) and Operations (Ops), whose approach intends to reduce those problems (Hussaini, 2014; Lwakatare et al., 2019). This approach is not limited to Development and Operations teams, but is a simple way to name it without extending to all the groups involved (Dyck et al., 2015).

The first usage of the term DevOps was presented by Patrick Debois, at the Agile 2008 Conference, where the need for an agile organization between Development and Operations teams was exposed (Mishra & Otaiwi, 2020). As an agile-based philosophy, DevOps is indicated in literature as having a positive effect on IT (Erich et al., 2014).

In order to clarify the DevOps paradigm, Dyck et al. (2015) proposed a scientific definition: “DevOps is an organizational approach that stresses empathy and cross-functional collaboration within and between teams – especially development and IT operations – in software development organizations, in order to operate resilient systems and accelerate delivery of changes.”

Otherwise, other authors describe DevOps as a set of practices whose objective is to reduce the timing between a change of a system and its promotion to the production environment. Though, the quality of the deliveries is also important. So, any practice that promotes these objectives is considered as DevOps (A. Brunnert et al., 2015; L. Bass, I. Weber, 2015).

Some benefits are highlighted in DevOps, such as increase of performance and productivity, better quality, costs reduction, operational efficacy and efficiency, and business alignment among Development and Operations teams (Luz et al., 2019).

In practice, a team with a DevOps implementation is formed by different players, which include: 1) for Development: business analysts, developers and testers of software and quality assurance members; 2) for Operations: database/systems/network administrators, web masters and security officers (Bang et al., 2013). Each of these players should anticipate the job to be done by the other and build a strong and continuous connection among all the players (Hemon et al., 2020).

DevOps is now a fundamental philosophy in organizations, working well for better communication between Development and Operations teams and as a part to deliver high quality products and services (Mishra & Otaiwi, 2020).

2.2 Team Management

Recently, organizations have become extremely competitive and based in moving environments. To face it, organizations have developed team-based design to maximize their human capital. So, teams have become the essential part of an organization (Mathieu et al., 2019).

However, teams of today are different from the past. Some characteristics are often present: dispersed, digital, more diverse, dynamic and with membership rotation (Hassan & Saeed, 2003).

A team can be defined as (Kozlowski SW, 2006): "(a) two or more individuals who (b) socially interact (face-to-face or, increasingly, virtually); (c) possess one or more common goals; (d) are brought together to perform organizationally relevant tasks; (e) exhibit interdependencies with respect to workflow, goals, and outcomes; (f) have different roles and responsibilities; and (g) are together embedded in an encompassing organizational system, with boundaries and linkages to the broader system context and task environment."

As a way to understand how a team works, evolves and which factors can contribute to its effectiveness, it is possible to analyse a team based in the IMO framework (Mathieu et al., 2019), where:

- (I) represents the Inputs – antecedent factors as member characteristics (e.g., competencies, personalities), team-level factors (e.g., task structure, external leader influences), and organizational and contextual factors (e.g., environment complexity)
- (M) represents the Mediating Mechanisms – describing the members' interactions resulting from holding tasks, as well as how the team inputs are transformed to outcomes, and collective effect and cognitions
- (O) represents the Outcomes – it means the outputs from team activity as results or products, these include performance (e.g., quality and quantity) and members' affective reactions (e.g., satisfaction, commitment).

However, recently new research appointed teams as more complex, multilevel and dynamic structures. So, as a way to include a dynamic view over the team effectiveness, Mathieu et al. (2019) proposed a new perspective which preserves the IMO framework, as represented in Figure 1. Thus, Inputs (Region A and B) and Mediating Mechanisms (Region C). Nevertheless, there are small regions (D, E and F) that recognize the nature of relationships between members and factors that result from intersection among the big regions.

Another perspective regarding team effectiveness, proposed by Haas & Mortensen (2016), enhances two vulnerabilities that, especially, modern teams are exposed to and managers should understand how to achieve big returns from their teams: (a) "us versus them" thinking and (b) incomplete information.

These two vulnerabilities can be reflected in Region C of Figure 1, where "us versus them" matches with "conflicts" and "incomplete information" with the problem of "Information Sharing" (Haas & Mortensen, 2016; Mathieu et al., 2019).

"Conflicts" are explained by Mathieu et al. (2019), as a process of understanding the incompatibilities or differences among group members. And three types of conflicts can emerge: (a) tasks oriented, (b) member relationships and (c) regarding processes.

Regarding “Information Sharing”, the same authors summarize this as the way the team makes their training resources available.

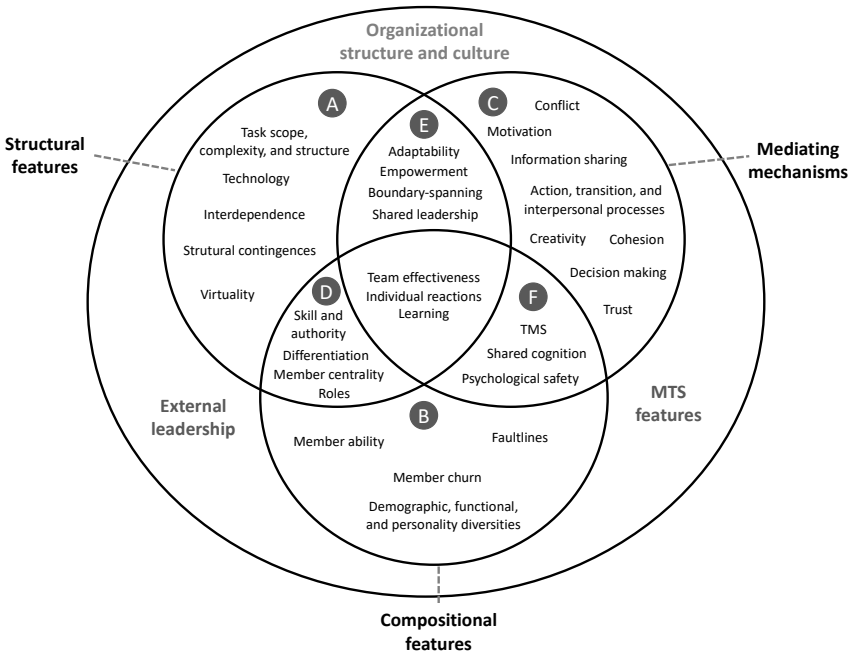

Figure 1 – Co-evolving team compositional and structural features, mediating mechanisms, external influences, and outcomes (Mathieu et al., 2019)

Otherwise, sometimes organizations must face some new problems that they have never faced. To tackle that, traditional teams’ structure is not practical, so, organizations build teams with different competencies and with colleagues from diverse fields and even cultures. This is a way to gather experts in groups to solve a problem and it is named: Teaming. This evolution of teamwork, with multiple teams together, can lead to chaos. It is necessary to embrace good team management principles. With that, important benefits can emerge and help the sharing of information, skills and networks; as well as, accelerate the delivery of products and services (Amy C. Edmonson, 2012).

Team management is not easy and, in last years, it has become more complex due to the increase of new team trends, such as virtuality, globalization and project orientation. So, the right approach to set the way to team success can make all the difference (Haas & Mortensen, 2016).

Chapter 3 – Related Work

Regarding the related work, a SLR was performed to summarize the existing work and provide a position for further studies. A SLR requires an exhaustive effort when compared with the traditional literature review (Kitchenham, 2007).

This SLR is based on the methodology proposed by Kitchenham (2007) with the review process presented in Figure 2.

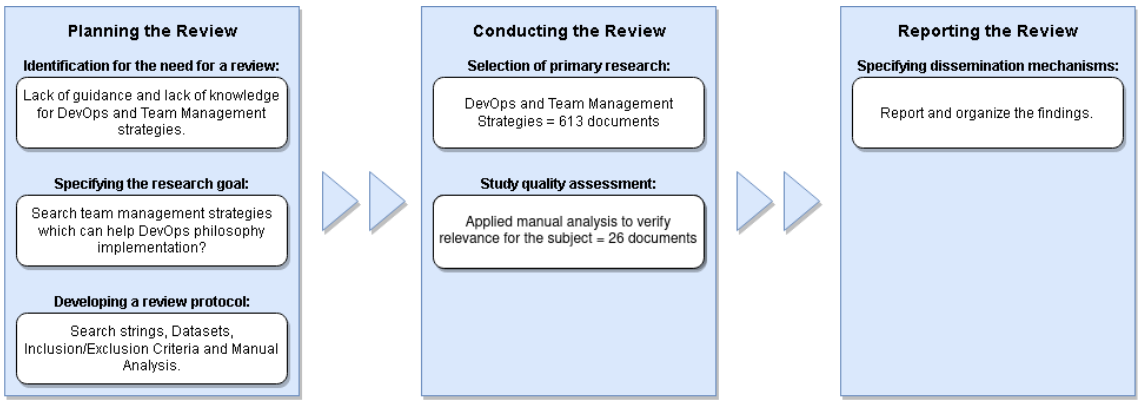

Figure 2 – Review Process (Kitchenham, 2007)

Using a strategy based on the model presented above, it is intended to summarize all the related work with correlation among DevOps and Team Management strategies.

3.1 Planning the Review

Before undertaking a SLR it is necessary to evaluate the need for such a review. In this section, the main need for this research is presented, defining the goal and developing a systematic review protocol (Kitchenham, 2007).

Usually, in many organizations that use IT, there is a clear division between Development and Operations teams. This idea of structural separation between these teams has started from the early days of computing and has been repeated often since then. Recently, with the evolution of IT toward making their organizations more competitive, faster and building more integrated products, interdependence was created between the Development and Operations teams where, sometimes, there is no clear separation of each team's work (Erich et al., 2014).

Facing this reality, a concept started to emerge called DevOps. Patrick Debois, at the Agile 2008 Conference, where the need for an agile organization between Development and Operations teams was exposed, presented the term DevOps for the first time (Mishra & Otaiwi, 2020).

Based on that, one of the most important values of Agile is "Individuals and interactions over processes and tools" (Beck et al., 2001). In a DevOps perspective, this core value should be highlighted, however, there is still much discussion around tools and processes (Erich et al., 2014) because some areas of DevOps continue to need further research (Mishra & Otaiwi, 2020) and there is lack of management structure between Development and Operations teams (Farroha & Farroha, 2014).

Regarding team management, the interactions mentioned above can be analysed as Mediating Mechanisms, as proposed by Mathieu et al. (2019), which includes key processes and emergent states: the members' interactions resulting from holding tasks, as well as how the team inputs are transformed to outcomes, and team affective and cognitive properties.

In order to deepen the lack of work related with DevOps and individuals and interactions, and using the point of Mathieu et al. (2019) regarding team management, this SLR aims to research team management strategies applied in DevOps context.

For that, this research considers the features of the IMO framework (Mathieu et al., 2019), with its aspects and processes, as a good way to include the major aspects that team management work can deal with. Thus, each feature of the IMO (referred in Figure 1) would be researched in DevOps context as a way to relate "Team Management Strategies" to "DevOps".

In this research, it was decided to cover only the Region C (referred in Figure 1), because there is proximity with the concept of Teaming (Amy C. Edmonson, 2012) and the conflicts of "us versus them" and "incomplete information" (Haas & Mortensen, 2016) as explained in section 2.2.

This SLR is focusing on getting the work that has been made related with the philosophy of DevOps and Mediating Mechanisms presented by Mathieu et al. (2019) in order to understand which team management strategies are being applied.

This phase begins with a literature review based on a search string in a set of chosen libraries. That way, it is supposed to get a vast work that could address the research goal defined in the last step. Below are the search string and the list of libraries.

String: DevOps AND (Conflict OR Motivation OR "Information Sharing" OR Creativity OR Cohesion OR "Decision Making" OR "Trust" OR "Action Process" OR "Transition Process" OR "Interpersonal Process")

Libraries: IEEE Xplore; Elsevier; Scopus; Web of Science

After that, the following exclusion criteria must be applied as a filter over the obtained documents, see Table 1.

Table 1 – Exclusion Criteria

Annotation	Description
F1	Filter documents since 2005
F2	Filter documents by type: Articles, Reviews and Conferences
F3	Filter documents only written in English language.
F4	Filter documents by free access or access given by the ISCTE protocols.

3.2 Conducting the review

This section is the second step of the SLR methodology undertaken during the related work. The review protocol described in the section "Planning the Review" has been applied and an analysis of the extracted documents performed.

After performing the search in the libraries with the search string mentioned in the section "Planning the Review" and applying the exclusion criteria presented in Table 1, it resulted in 613 documents.

In this step, based on the 613 documents reported in the last step, a manual analysis to exclude documents not related with DevOps implementation was performed and the meaning of each concept described as Mediating Mechanisms by Mathieu et al. (2019), in Region C of Figure 1.

In order to understand all the steps carried out, all the process is reported in Table 2 with the application of the filters, as well as the manual analysis referred above which is represented in the column “Result”.

Afterwards, the final result is 26 documents, excluding duplications. This final set of documents is, in a first analysis, related with the goal of this SLR. Some of the excluded documents, although the search string was present, were not related with DevOps implementation and others were not using the rest of the search string in the context of team management aspects as established by Mathieu et al. (2019).

Distribution of the Results

To better understand the results produced from this SLR, it is possible to see in Figure 3, the distribution of the documents found by Mediating Mechanism. Here, duplicated documents were discounted only when in the same Mediating Mechanism. There are documents repeated for different Mediating Mechanisms.

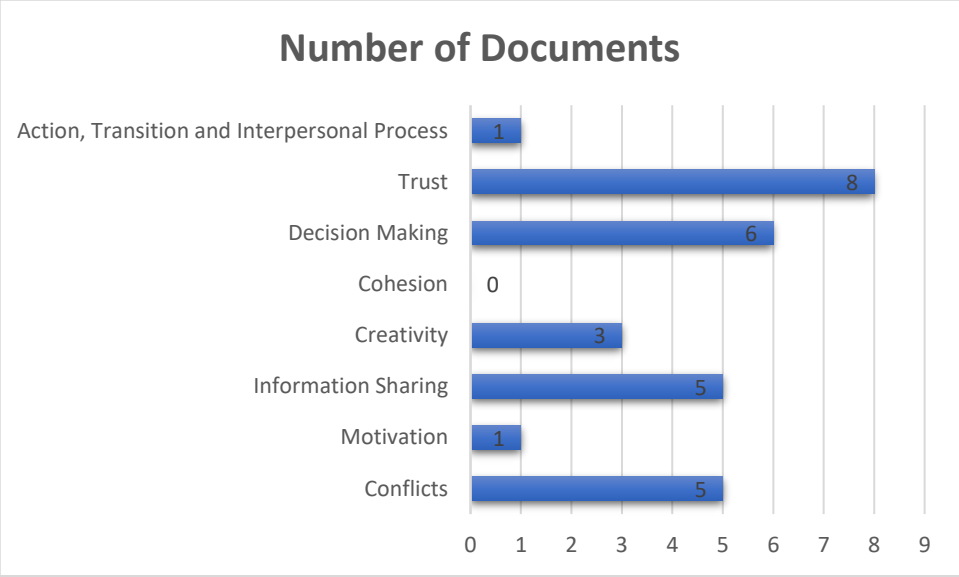

Figure 3 – Distribution of Documents by Mediating Mechanism

3.3 Reporting the review

In this section, all the relevant information found during the SLR from the documents obtained between September and December of 2020 is presented. The information is

aggregated by topic, related and summarized in order to better understand the documents and their relevance to this research.

Conflicts

Regarding conflicts, some references obtained touch on this concept but in a context of non-DevOps and refer to the conflicts as a gap between Development and Operations teams and as a challenge toward DevOps implementation (Hussaini, 2014; Luz et al., 2019; Lwakatare et al., 2019). Other work mentioned that openness can mitigate conflicts, whereas biased attitude can do the opposite. However, in the same document conflicts are also considered as positive, when there are shared opinions, and often, the relation with conflicts depends of how individuals perceive them (Basirati et al., 2020).

The unique mention of conflicts in DevOps context points to conflicts on tools and processes as a possible delay for the delivery time, as referred by Basirati et al. (2020).

In this SLR, documents with team management strategies to face conflicts in DevOps implementation were not found.

Motivation

Regarding the Agile approach, Kropp et al. (2020) verified satisfaction in professionals working this way, but there is no reference to conclusions in the DevOps approach, although, DevOps is based on the Agile approach.

However, in this SLR documents with team management strategies to promote motivation in DevOps implementation were not found.

Information Sharing

In non-DevOps teams, information sharing is considered lacking (Kumar & Goyal, 2020) and in DevOps teams, it is indicated as a characteristic (Dennehy & Conboy, 2017; Gupta et al., 2017; Jedlitschka et al., 2014) and as having significant impact on the quality of software or products (Mishra & Otaiwi, 2020).

During a process of DevOps implementation, it is normal that the people involved (developers and operators) show resistance to the change and fear the lack of knowledge about the new methodology and technologies. As a solution, it is possible to create a framework were the common processes, methods and tools are present to encourage collaboration and information sharing. It is also recommended that communication and information should be transparent among the teams, business and other relevant stakeholders (Kumar & Goyal, 2020).

Creativity

Several universities apply Agile, Scrum and DevOps methodologies in software engineering students' studies to improve their creativity (Castillo-Salinas et al., 2020; López-Alcarria et al., 2019; Melegati et al., 2019).

Nonetheless, during this SLR documents with team management strategies to increase creativity in DevOps implementation were not found.

Table 2 – Resulting research based on exclusion criteria present in Table 1

Keywords	Library	Total	F1	F2	F3	F4	Result
DevOps AND Conflict	IEEE Xplore	4	4	4	1	1	1
	Elsevier	123	122	85	6	6	6
	Scopus	12	12	12	11	11	1
	Web of Science	6	6	6	6	6	1
DevOps AND Motivation	IEEE Xplore	5	5	5	5	5	0
	Elsevier	114	113	91	91	91	6
	Scopus	17	17	16	7	7	0
	Web of Science	11	11	11	7	7	0
DevOps AND "Information Sharing"	IEEE Xplore	0	0	0	0	0	0
	Elsevier	27	27	24	24	24	3
	Scopus	14	14	12	10	10	1
	Web of Science	2	2	2	2	2	1
DevOps AND Creativity	IEEE Xplore	0	0	0	0	0	0
	Elsevier	32	30	18	18	18	3
	Scopus	21	21	14	14	14	2
	Web of Science	1	1	1	0	0	0
DevOps AND Cohesion	IEEE Xplore	0	0	0	0	0	0
	Elsevier	17	16	11	11	11	1
	Scopus	15	15	14	12	12	0
	Web of Science	1	1	1	1	1	0
DevOps AND "Decision Making"	IEEE Xplore	5	5	5	5	5	0
	Elsevier	141	139	91	91	91	2
	Scopus	204	204	181	175	144	8
	Web of Science	10	10	10	10	9	1
DevOps AND Trust	IEEE Xplore	7	7	7	7	7	1
	Elsevier	155	154	103	103	102	5
	Scopus	17	12	12	12	12	4
	Web of Science	11	11	11	11	11	2
DevOps AND "Action Process"	IEEE Xplore	0	0	0	0	0	0
	Elsevier	1	1	1	1	1	0
	Scopus	0	0	0	0	0	0
	Web of Science	0	0	0	0	0	0
DevOps AND "Transition Process"	IEEE Xplore	1	1	1	1	1	0
	Elsevier	1	1	1	1	1	0
	Scopus	1	1	1	1	1	0
	Web of Science	0	0	0	0	0	0
DevOps AND "Interpersonal Process"	IEEE Xplore	0	0	0	0	0	0
	Elsevier	0	0	0	0	0	0
	Scopus	2	2	2	2	2	1
	Web of Science	0	0	0	0	0	0
							26

Cohesion

In this SLR documents with team management strategies to increase cohesion in DevOps context were not found.

Decision Making

Convergence between Development and Operations teams during a release process can help the Operations team to find problems faster, react to them and contribute to the decision about the next actions. This convergence is strongly related to DevOps principles (Schermann et al., 2018).

Decision making is a skill that is recommended for DevOps members to have as a way to get fast decisions, take responsibilities and be self-organized. With an ideal DevOps team, members are able to appreciate decision making and the process is shared between both teams (Hemon et al., 2020; Wiedemann & Schulz, 2017; Wiedemann & Wiesche, 2018).

To better decision making, Wettinger & Andrikopoulos (2015) proposed a holistic approach to organize knowledge in a DevOps Knowledge Base. That way, there is a sharing of information among the teams to better decisions.

Trust

DevOps is presented as an enabler for trust between Development and Operations teams (Alonso et al., 2019; Farroha & Farroha, 2014; Mishra & Otaiwi, 2020) and it motivates developers (link with “Motivation” section above) (Farroha & Farroha, 2014). Furthermore, the Agile approach and mutual trust are directly related (Wen et al., 2020) and trust is seen as a crucial element to Agile (Jesse, 2019).

A different point of view is presented by Luz et al. (2019) and Masombuka & Mnkandla (2018), where trust is crucial for DevOps culture implementation. Thus, three types of trust were identified by Masombuka & Mnkandla (2018) as needed by DevOps teams: 1) trust between both teams; 2) trust between the Development and QA teams; and 3) trust between the Product Manager and the Operations teams.

During the implementation process of DevOps, Nybom et al. (2016) observed mistrust when it was necessary to give administration access to the Development team, nevertheless, through the process of teaching/learning from others, sharing responsibilities and more collaboration between both teams; Development and Operations teams increased their levels of trust. One of the consequences was increasing information sharing (link with section “Information Sharing”).

Action, Transition and Interpersonal Process

In the document of Shropshire et al. (2017), some interpersonal aspects are relevant to moderate uncertainty toward DevOps: 1) conscientiousness, 2) extroversion and 3) neuroticism. Otherwise, the others are not so significant: 1) openness and 2) agreeableness. As a conclusion, personalities matter in order to lead with uncertainty toward DevOps. It is important that managers know that different personalities have different impacts and reactions. Clarification and affirmation can reduce levels of uncertainty.

In this SLR documents with action and transition process concept in DevOps context were not found.

3.4 Related Work Synthesis

The results of this SLR demonstrate a lack of investigation in DevOps teams and strategies to manage those teams. Some of the resulted documents report benefits of applying DevOps and how this can improve some vectors like conflicts, for example. However, there is no found previous work that studied a DevOps application and which managing strategies must be used in order to achieve better performance in those teams. Based on this lack of knowledge and looking for better understanding of managing those types of teams, it is important to study the impact of those mediators on DevOps teams.

Chapter 4 – Methodology

In this section, how this study is conducted in order to understand how team management strategies can assist and improve DevOps implementation is presented. For such a goal, DevOps teams have been chosen to be studied, through a case study, following the approach of Thomas (2016) like presented in Figure 4.

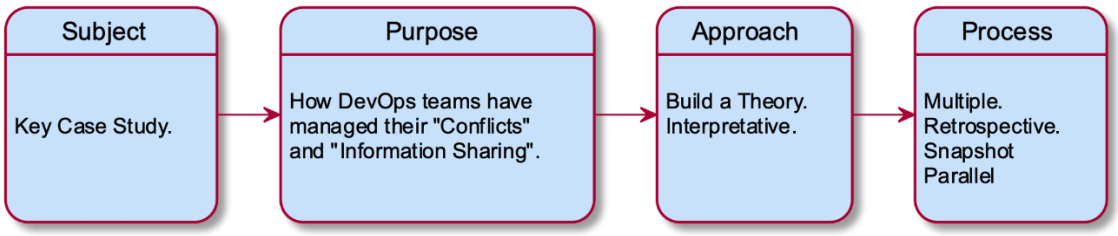

Figure 4 – Methodology Approach by Thomas (2016)

The *Subject* of this research is a "key case study" because DevOps teams that will be considered are exemplary cases where an in-depth study will be made (Thomas, 2016). Those teams will facilitate deep understanding of theory of Mathieu et al. (2019) regarding team effectiveness and other authors, such as Amy C. Edmonson (2012) and Haas & Mortensen (2016) presented in section 2.2, where managing "conflicts", "information sharing" are crucial for team effectiveness.

Thus, the *Purpose* of this case study is categorized as:

- *Instrumental* – given that the case study is a way to understand a specific situation
- *Exploratory* – because DevOps teams are recent and there is a scarcity of information about management strategies applied in DevOps context.

Based on this principle and following the Storyboard presented in Figure 5, the purpose of this study is unfolded with the following question: How have DevOps teams managed their "conflicts" and "information sharing"?

Regarding the *Approach*, the case study will focus on:

- *Build a Theory* – because there are no theories founded in literature about managing "conflicts" and "information sharing" in DevOps teams

- *Interpretative* – it is intended to get a deep understanding about DevOps teams and how they can be managed toward effectiveness and better performance.

For this research, there is no initial theory to test. However, after the research it is possible that some ideas can emerge and, perhaps, can help to build a theory for further studies. Otherwise, this research is expected to immerse deep in the environment of each DevOps team and get a deep understanding of how those teams are managed regarding conflicts and information sharing, as well as highlight the best team management strategies that assist DevOps and promote team effectiveness (Thomas, 2016).

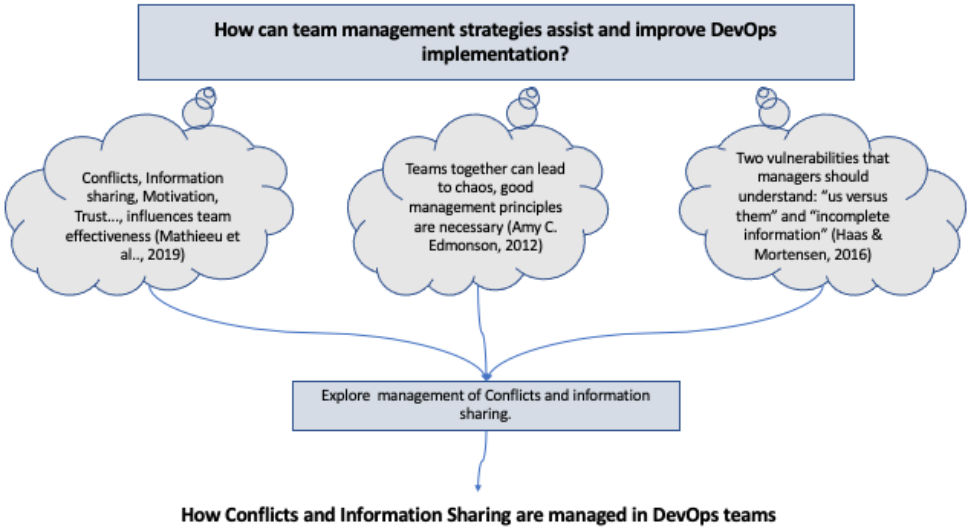

Figure 5 – Initial Storyboard of exploration of ideas to Case Study

The interest of this study is supported on its own diversity of *Subjects* studied. Thus, it is especially important to seek different DevOps teams in different environments. Regarding time, it is important to study when the DevOps team was formed and how conflicts and information sharing were managed, just as how the same aspects are managed today. Each team studied will be compared with the other to better understand the whole impact of management strategies (Thomas, 2016).

Resuming, the *Process* recommended by Thomas (2016) can be described as:

- *Multiple* – more than one team will be needed to better understand the environment and the *Subjects*
- *Retrospective* – the study will focus on the time when the team was formed
- *Snapshot* – the study will also focus on the current time
- *Parallel* – the study of each DevOps team will be parallelized.

This research was conducted for 3 months (March to May of 2021) over 8 different multinational companies (Telecommunications, Bank, Automobile and Technology companies), with a sample of 10 teams and a sample of 15 interviewees (see Figure 6), for a qualitative approach.

In this research, there are only 2 elements present that had experience merging transformation of Development and Operations teams in their actual teams when they were interviewed: a DevOps Lead (who manages different DevOps teams) and a Developer. The other interviews did not have that experience inside their teams at the time of the interviews.

The collection of the elements for this sample was made initially through colleague contacts and each new element was asked about new elements that could participate in this study. At some point these were not enough, and other resources were used, like professional networks.

Each interview was conducted online using Portuguese language (native language for the interviewer and all interviewees) and all the participants authorized the recording of their interviews. The interviews did not have a fixed questionnaire with fixed questions, but had a structure: 1) experience, team and role, 2) interpersonal conflicts, 3) task conflicts, 4) process conflicts, 5) “us versus them” conflicts, and 6) information sharing. The option for non-fixed questions was based on Thomas (2016) principles: the interviewer most feel the best way to achieve the needed answers, with open questions preferentially, exploring some topics and using the answers as the base to new topics and hidden information.

In order to better understand the given answers, each recorded interview was reviewed and the relevant topics were passed to a written support with the same words that were

spoken. The written information was arranged with the same interview structure mentioned above. Afterwards, each written interview was compared with the same team interviews and with other team interviews and conclusions were taken through the correlation of the information. The overall conclusions were translated to English language to be aligned with the same language of the present study.

To support the analysis in this report, the names presented on Figure 6 are used for better identification of Teams and Roles. Roles are divided into two different colours: lighter to identify Managing Roles and darker for non-Managing Roles. In Table 3, there is context of each Interviewee`s role and time of experience.

Table 3 – Interviewees' roles and experiences

Interviewee	Current Role	Overall Experience
Interviewee 1	- DevOps Engineer (3 months in this role at current Company)	- More than 10 years of IT Consultant at different Companies
Interviewee 2	- Automation Tester and QA Environment Manager (8 months in this role at current Company)	- More than 8 years of Software Engineer at different Companies
Interviewee 3	- DevOps Engineer (5 months in the role at current Company)	- More than 10 years of experience at different Companies, such as Team Leader, Scrum Master, Software Engineer, etc.
Interviewee 4	- DevOps Engineer (5 months in the role at current Company)	- 1 and a half years at current Company More than 4 years of experience as Data Scientist at current and different Companies
Interviewee 5	- Full Stack Developer (6 years in the role at current Company)	- More than 12 years of experience at current and different Companies as IT Consultant
Interviewee 6	- Manager (1 year in the role at current Company)	- More than 9 years at current Company. - More than 20 years of experience at current and different Companies, such as System Administrator and IT Manager, etc.
Interviewee 7	- Manager (2 years in the role at current Company)	- More than 11 years at current Company. - More than 15 years of experience at current and different Companies, such as IT Consultant, Project Manager, etc.
Interviewee 8	- Tech Lead and Scrum Master (2 years in this role at current Company)	- More than 10 years of experience at current and different Companies, such as Developer, Configuration Manager, DevOps Senior Consultant, etc.
Interviewee 9	- Developer (more than 3 years in this role at current Company)	- More than 12 years as Configuration Manager and DevOps Consultant at current Company
Interviewee 10	- Developer (almost 2 years in this role at the current Company)	- More than 4 years as Developer at current and different Companies.
Interviewee 11	- Business Analyst (1 month in this role and in the current team)	- More than 15 years of experience as IT Consultant at current and different Companies and clients.
Interviewee 12	- DevOps Lead (almost 2 years in this role at the current Company) - Experienced a Development and Operations merging.	- More than 12 years of experience at current and different Companies, such as Software Engineer, IT Consultant, etc.
Interviewee 13	- Developer (5 years in this role only at the current Company) - Experienced a Development and Operations merging.	- 5 years of experience as Developer only at the current Company
Interviewee 14	- Head of DevOps Practices (2 months in this role at the current Company)	- More than 2 years of experience leading teams at the current Company - More than 7 years of experience at current and different Companies: Project manager, Process Manager and Incident Manager
Interviewee 15	- Head of Automation Tests (more than 2 and a half years in this role at the current Company)	- More than 12 years of experience at current and different Companies as Developer

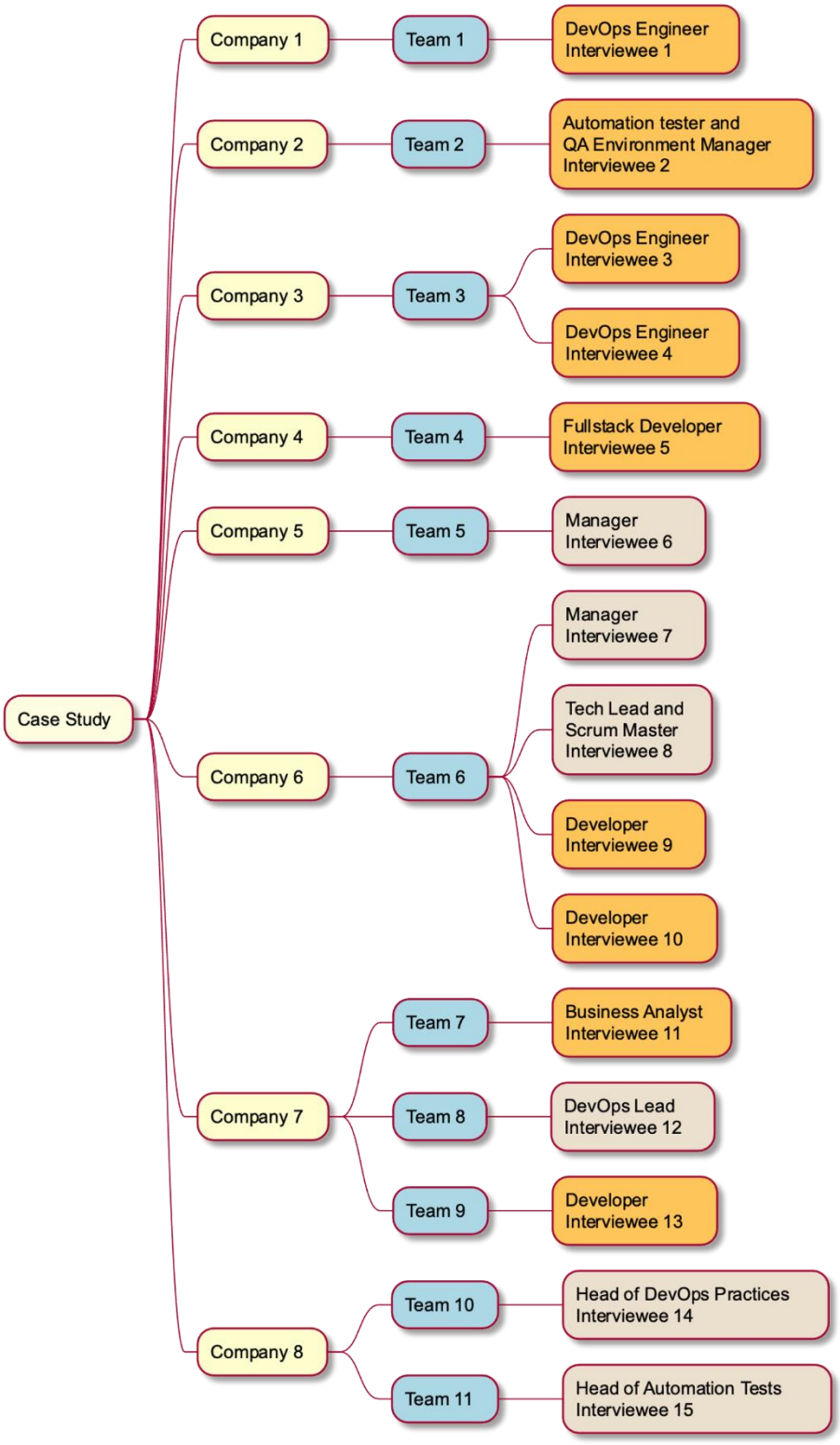

Figure 6 – Case Study Interviews Diagram with different colours for Managing and non-Managing Roles

Chapter 5 – Reporting the Findings

5.1 Qualitative Evaluation

5.1.1 Interpersonal Conflicts

It is all about personalities. When the interviewees tried to explain why they have interpersonal conflicts, the main global explanations were that it is the way people are, "personalities that sometimes clash" (Interviewee 5), "different personalities" (Interviewee 9), or even, "the human part is very difficult" (Interviewee 3).

There were not profound explanations. Most of them were abstract and generalized, without confidence in the real reasons. Nevertheless, there were some more specific, like "team dimension (…), there are people who are more assertive and others more relaxed, people that can put more pressure on themselves and on others, team communication problems and verbalization difficulties" (Interviewee 9) – these verbalization problems have grown with pandemic situation; "different perspectives, (…) reactive or stressed people, (…) lack of emotional management" (Interviewee 13) and "leaders without perspective" (Interviewee 15). A deeper understanding was difficult to get.

However, the highlighted aspects that were mentioned as promoters to reduce conflicts are focused on the team characteristics: "the personalities work very well, cordial, compatible and fit well with each other" (Interviewee 1), "good sense of humour and good team spirit" (Interviewee 2), "mutual help" (Interviewee 11) and "team union is indispensable" (Interviewee 13). The team knowledge and experience are outlined too, "the most important thing is the people that have knowledge, they know what they are talking about and what they are doing" (Interviewee 2).

The communication problems are mentioned as a big handicap in these types of conflicts because "the way people express themselves sometimes seems aggressive in others' point of view" (Interviewee 4), "the majority of my colleagues do not do anything, they simply accept" (Interviewee 3) and "we do not do anything to resolve conflicts" (Interviewee 13).

In order to solve the communication problems, the recommended strategies highlight "open communication, dialogue between the involved parts" (Interviewee 4), exposing the situation when it "was not comfortable, and in the end, chatting about it generally

works" (Interviewee 3) and "speak what you feel (…) with transparency and honesty" (Interviewee 15). Without good communication, many times the conflicts are in our head, "we think there is conflict but really there is not" (Interviewee 4).

The dialogue is the base for team members to clarify and resolve conflicts, and it must be clear, straight, fast and early.

For leaders, it is recommended to be a promoter of "good connection, good environment, positive and constructive speech" (Interviewee 1), encouraging "conflicts of ideas" (Interviewee 7) and "feedback" (Interviewee 10), observing the personal relationships in the group, for example, "during lunch or a coffee break" (Interviewee 7) to understand the team´s relationships and some leaders are doing "emotional (…) and close management without doing micro-management" (Interviewee 8).

Some topics are managed carefully, "soccer, politics and religion are considered taboo to avoid arguments, these topics are mentioned in light way" (Interviewee 8) as a way to control the conflicts. When it is necessary to lead and align some attitudes, it is recommended to "talk privately and never humiliate the person in public: notes and observations are always in private" (Interviewee 8).

A few qualities of a good leader were emphasized, "have an innate capacity to manage conflicts, be human and emphatic, the more empathic the better, to take the micro-expressions" (Interviewee 8) and understand the teams' emotions in order to "recognize the best positive moments and make them ride this wave" (Interviewee 8).

These recommendations are aligned with the findings of the SLR made during this study, where Shropshire et al. (2017) highlight the importance of leaders understanding and knowing how to manage different personalities.

In terms of formal events, leaders must arrange one-on-one meetings with each member, at least one per month is recommended, in order for each one to "talk about what is good, what is wrong, what can be better for each one and receive feedback" (Interviewee 2). Other options were presented as a fixed guide "with 4 mandatory questions: 1) What are you happy about? 2) What are you unhappy about? 3) What do you want to improve? and 4) Do you have any idea that you would like to bring to the company? (Interviewee 14). And it is important to conduct the conversation with

questions without judgement to “understand the person in a non-vitiated way in order to know the root cause” (Interviewee 7) of the conflicts.

As a way to ensure that everyone can express themselves, it is advised to make “anonymous mechanisms to get people’s feedback which allows for the evaluation of different levels: happiness, wellbeing, work/personal life conflicts, training and others” (Interviewee 14).

Another suggestion highlights the beginning of team meetings, where the leaders can check the current state of mind of each member where “each one chooses one word to describe the way they feel” (Interviewee 13) – especially in remote work, it is more difficult for the team to understand the emotions of the others. And invest in team talks to “demystify emotions, stress, anxiety and overwhelm” (Interviewee 13).

In the interviews for new positions in the team, the leaders try to carefully choose the best match for the team, not just technical skills, but also in terms of conduct and attitude; and it is interesting to know that there is some “fear to hire someone (…) who could be conflicting” (Interviewee 12).

For the companies, there are some recommended ideas, such as, a Happiness Department which is responsible for “guaranteeing the happiness in the whole company” (Interviewee 3). Another relevant aspect is a top-down strategy to clarify and align all the company toward a direction, this can resolve some conflicts, because “the company culture overlap the management style” (Interviewee 4). The company culture is shown to be very important, “a culture of let’s talk and not guilt the others and a blameless culture is very important” (Interviewee 6) and “the culture makes it a lot easier” (Interviewee 13) to build team spirit.

When all these strategies fail and there is, for example, “a person that does not fit the team motto” (Interviewee 7), the last options mentioned are “try to move the person to another team or client” (Interviewee 7), “give different topics and responsibilities to avoid those elements working directly together” (Interviewee 12), or even, the extreme scenario, “fire the person” (Interviewee 5).

In Table 4, a summary of given strategies is presented.

5.1.2 Task Conflicts

The main reasons indicated to have these conflicts are mainly because of unclear responsibilities like in the case of a "Product Owner and Scrum Master, (..) some stress, (…) changes of scope, (…) lack of open communication" (Interviewee 4), "resistance to change" (Interviewee 15) and lack of good and clear instructions and specifications, which is considered "a transversal problem in Software Engineering" (Interviewee 9).

Regarding tasks, the knowledge and experience of the team as a whole and of each member can make the difference in having or not having a lot of this type of conflicts. The experience continues to be respected as a decisional factor and helps to avoid some conflicts: "as the most senior member in the team, I tend to have not many conflicts, because people rely on me, even when I am not so sure" (Interviewee 3) and "it is clear noted who has experience and knows the solutions and because of that, it works well" (Interviewee 11).

In some teams, the leader's opinions can "overlap the other member's opinions because of the former's background" (Interviewee 13) and this situation deserves caution, because "when there is someone who has a pronounced opinion, it is implied that it is the leader who is right" (Interviewee 13). This is a sensitive situation that must be managed with good sense.

Otherwise, some leaders transform the task conflicts into an opportunity to implement a democratic decisional system where "we join more people, and we look for the best way to do it and, what is decided, is what we will do" (Interviewee 5). There is no judgement, "all the opinions are welcome" (Interviewee 8).

It is important that leaders follow some advice to better manage those moments: instead of imposing the solution, trying to "lead the solution, listening to all the proposed solutions and motivating the team to express their opinions" (Interviewee 8), "being constructive" (Interviewee 15). It is also important to be an adaptable leader and understand that the set of team ideas/solutions could be better than their own. During all the process, the leader must "stop thinking and use active listening" (Interviewee 15). At the end, "decisions are made based on facts" (Interviewee 8) and the leader must be a "good mediator" (Interviewee 8).

In terms of team organization, the events of the SCRUM framework are very used and recommended as the refinements: meetings where all the team refine each requirement, since “a very accurate, very wide, very extensible refinement meeting with all team” (Interviewee 2) adds to the team efficiency. Thus, the team has the opportunity to “discuss everything, including task delivery” (Interviewee 2), and when it is time to develop, the team have “everything structured, schematized, planned and this make us more successful” (Interviewee 2).

Continuing in the field of SCRUM events, it is advised to get daily`s, which are daily meetings of fifteen minutes where all the team is together, speaking about what each one did in the workday before, and what they are going to do in that day. This type of meeting can “help the team in managing task conflicts and doubts” (Interviewee 10).

A Kanban Board is also mentioned as good for task management and avoiding conflicts. In addiction, there are some referred tools that can “allow constant and asynchronous feedback about the work” (Interviewee 14) like Slack or Teams and it is “important to have a centralized point to see all the work” (Interviewee 14) like Jira or ServiceNow.

Another tip for team organization is to work with checklists in order to not forget the crucial steps before launching a product or a system. This way, it works like a pilot checking everything before “entering a plane and everyone understand” (Interviewee 6) the benefits of these procedures.

Sometimes, when the conflict between the parts is difficult to solve, escalating can be a good choice. Leaders can have an important role here and can show the parts involved “a different vision and thinking of the reasons why that situation happened” (Interviewee 4) in order to best understand the other’s opinion or behaviour and the environment.

It is also easy to turn a task conflict into an interpersonal conflict. And sometimes there are doubts to understand what is the root cause of such conflict, because “it is a little interpersonal and task conflict” (Interviewee 4). Both conflicts are very correlated.

In Table 4, it is possible to see a summary of the founded strategies.

5.1.3 Process Conflicts

There are many reasons indicated as valid to create process conflicts. The use of the SCRUM framework, for example, is highlighted as “the big reason to have conflicts” (Interviewee 3) because of some confusion with the Product Owner role and Scrum Master role and because “the Product Owner is not focused on improving processes” (Interviewee 6), giving preference to business features.

It is not because of the framework itself, but sometimes the people, teams and companies that are making their transformation processes or adapting business and, for example, some “conflicts between the main company and succursal happen” (Interviewee 3). This situation causes different interpretations of the transformation process, with “old and bureaucratic processes” and sometimes with no tangible strategy defined or with wrong expectations: “if everything is urgent, nothing is urgent” (Interviewee 3). All of this causes undefined failures in processes, resulting in conflicts.

Nevertheless, there are other teams better adapted to the SCRUM framework and see daily`s as “helping to resolve process conflicts” (Interviewee 10) and this perception came with time and “with a great level of Agile maturity” (Interviewee 11). This makes “everything very outlined and people follow all the steps” (Interviewee 11) of the processes.

For the leaders, it is recommended to “look for process failures in order to protect people” (Interviewee 6) and, in the planning meetings of the SCRUM framework, the presence of the leader is crucial. The objective is to enhance the “importance of non-functional requirements and plan the future” (Interviewee 6) of the system or application, since the Product Owner tends to have a business vision of the backlog and not a processual or technical vision.

There are some examples where processes are established and without conflicts. This happens when leaders “have the processes well defined and structured (…) and people agree with them” (Interviewee 11), and because of that, “everybody complies” (Interviewee 5).

Sometimes, DevOps teams have to challenge the current processes and practices. Some guidance was given in the spirit of continuous improvement. It is necessary “to verify how these new processes or practices accommodate the old processes”

(Interviewee 15). A good collection of evidence is necessary: “where the process works and what the pain points are” (Interviewee 15). After that, having “meetings to promote alignment where there is misalignment and discussing changes” (Interviewee 10), because people are the big threat: “people are not easy to change” (Interviewee 3) and offer resistance to new changes.

However, this guidance can be applied to all the teams because a process is something established, but is also something that is never closed. Its efficacy can change with the environment changes, number of people, number of systems, etc. The goal is looking for “continuous feedback and adapt all we define: practices, processes or tools; in order to respond to the needs of everyone and simplify their lives” (Interviewee 15). It is important to find which improvement in processes “can protect people” (Interviewee 6) from mistakes.

And the application of a new process can itself bring other conflicts, so, “it is better to have the main people involved in the discussion of the process, with their arguments, in order to better find the best common ground” (Interviewee 15).

In Table 4, a summary of main strategies to manage these conflicts is presented.

5.1.4 “Us versus Them” Conflicts

Some reasons were presented as catalysers to conflicts between different teams that should work and cooperate to the same goal. Sometimes, “different cultures and time zones” (Interviewee 1) create an extra layer of situations to manage and “influence the appearance of conflicts” (Interviewee 1).

However, conflicts between Development and Operations teams were more present at the beginning, when the teams had just joined and tried to align with DevOps practices. Regarding this type of conflicts, during the DevOps implementation process, it is possible to conclude that there is misunderstanding about old and new roles, such as difficulties for developers to take new operations responsibilities: “So, now am I going to do prevention, receive alerts and render service to Production?” (Interviewee 6). Levels of competition and fear emerge from both teams as they are trying to merge processes and, at the same time, leaders are trying to convince people to accept the change and break the “resistance” (Interviewee 7). Some old habits and old suppliers can drag the DevOps

culture down for years. Nevertheless, “attracting people, explaining that the process will be laborious, but showing the advantages” (Interviewee 7) of DevOps implementation is highly recommended.

In those situations, a direct message from the board of the company can make the difference: “DevOps is the only way, nobody has apologizes for being misaligned” (Interviewee 7). Next, the awareness of the leaders to correct the old linguistic habits as “they” or “their process” (Interviewee 12) and reinforce that everybody “is in the same boat” (Interviewee 12) is also a good strategy.

It is important for leaders to focus on solid foundations and guide the teams through “Agile methodologies, with rituals, like Planning of Sprints, with both teams involved and, in some cases, with businesspeople” (Interviewee 14). In summary, all the people involved in the software cycle development “must be involved in the same meetings to gain awareness” (Interviewee 13) and work closely to guarantee functional and non-functional requirements. “Typically, Operations teams have non-functional requirements that are not contemplated into the Development phase” (Interviewee 14) but they should be. Some examples are: “type of monitoring, performance, type of performance impact, availability, etc.” (Interviewee 14).

Another recommendation is to put someone inside the teams that can evangelize the new DevOps practices and culture, and to “understand what operational responsibility means” (Interviewee 6). For the DevOps culture to grow, “analysing, understanding and not judging” (Interviewee 7) are good principles to facilitate the acculturation.

When cooperation between different teams is necessary, “a spirit of mutual help, even when people are separated” (Interviewee 2, 11) is essential to a good DevOps implementation. Some reasons are indicated as motivators to control this type of conflicts. The “seniority and maturity of everyone” (interviewee 2) and “people looking at themselves as only one team” (Interviewee 11) is recommended to have successful joining teams, as well as “no lacks of communication” (Interviewee 2).

In Table 4, a summary of strategies to manage these types of conflicts is presented.

5.1.5 Information Sharing

There are some complaints about the information sharing. There are usually two models to share information: first, by written documentation in documents, task managers and wiki platforms; and second, by interactive information though meetings, “shadowing” (Interviewee 10), “pair programming” (Interviewee 3, 10), talks, emails and chat platforms.

Teams face some problems with written documentation. Sometimes the documentation is missing, documentation that only serves for “those who already know” (Interviewee 1), “duplicated documentation” (Interviewee 15) and, decentralized and outdated documentation, because “things evolve fast” (Interviewee 1, 12).

With interactive information, if there is a “big people rotation on the team, the team finds a problem of knowledge” (Interviewee 5) since the information is, mostly, in people’s heads and is based on “trust that the other person noted everything” (Interviewee 13). However, some leaders verify benefits, such as “agility, fluidity (…) and fewer costs” (Interviewee 12).

In the onboarding process, some difficulties were related. It is not a well-defined process for some teams and it is “a despair for the one who just entered and sees millions of things happening” (Interviewee 3), “it is difficult to understand what is necessary to implement” (Interviewee 9) or it is even, “scary to read all the documentation” (Interviewee 9).

As for strategies to improve information sharing, there are some recommendations for the leaders like “centralization of documentation” (Interviewee 1) in one unique place and “democratization of all the information by all the company” (Interviewee 8). The principles like “the documentation is open by default, it is only restricted if there is an exception, (…) radical transparency, (…) and all the teams should have access to the data that allows them to manage the application” (Interviewee 14) are highly recommended. The usual problem of Development accessing Production data should not be present in a DevOps culture. If there is “sensitive data, like client’s information, that should be anonymized” (Interviewee 14). These recommendations are aligned with the findings of the SLR made during this study, specifically, the documentation centralization (Wettinger & Andrikopoulos, 2015) and transparency (Kumar & Goyal, 2020).

Another strategy is to use written documentation for information that the team wants to be "historic and that will help in the future" (Interviewee 4), for example: explanations of processes, flows, "team analysis, project definitions" (Interviewee 2), "generic solutions" (Interviewee 6), "trouble-shooting, sensitive information" (Interviewee 9), etc. All this documentation must be stored in a central repository "with a version control system and when anyone makes a change, all the team has to approve" (Interviewee 6) in order to get reliable documentation. Note, if the company needs to share documentation with clients, an "information space for clients and another for the team" (Interviewee 8, 10) must be created.

Use the interactive information model when "discussions between the team" (Interviewee 2), "generic doubts and issues, information that is necessary to be spread rapidly by the team" (Interviewee 3) and when "remote work" (Interviewee 6) are necessary. The use of "SCRUM events are good for this type of information sharing" (Interviewee 5) and it is necessary to guarantee the use of a "good communication" (Interviewee 13). To avoid missing information caused by people rotation, "try to have redundancy" (Interviewee 7, 8) of roles to spread the knowledge.

For onboarding processes, in order to better host the new member, "have a boot camp" (Interviewee 9) to welcome them, assign a 'buddy'– "a person to follow-up with the new team member" (Interviewee 11, 12); and "show documentation smoothly" (Interviewee 11) throughout the process.

The more common tools found to manage all the information were: Jira, ServiceNow and Azure DevOps as task managers, Confluence as wiki platform, Slack, Teams, Skype as chat platforms; and Teams, GIT, Sharepoint as document repositories.

In Table 4, there is a summary of founded strategies to manage information sharing.

Table 4 – Summary of Strategies to Manage Conflicts and Information Sharing

Role	Strategies to Manage Interpersonal Conflicts	Strategies to Manage Task Conflicts	Strategies to Manage Process Conflicts	Strategies to Manage "Us versus Them" Conflicts	Strategies to Manage Information Sharing
Team Members	- Use the dialogue as the base to resolve conflicts. - Open communication. - Speak what you are feeling with transparency and honesty. - Be aware that sometimes the conflict is in your head. - The dialogue must be clear, straight, fast and early.	- No evidence founded.	- No evidence founded.	- No evidence founded.	- No evidence founded.
Leaders	- Promote good connection, good environment, positive and constructive speech. - Encourage conflicts of ideas and feedback. - Observe and understand the personal relationships of the team in group. - Do emotional and close management without micro-management. - Manage taboo topics carefully. - Align attitudes privately and never humiliate in public. - Be empathic and catch the emotional expressions. - Do one-to-one meetings with each member, at least one per month, to promote understanding and receive feedback. - Implement anonymous mechanisms to receive feedback. - Evaluate different aspects: happiness, wellbeing, work/personal life conflicts, training and others.	- Lead the solution, listening to all the proposed solutions and motivate the team to express their opinions. - Understand that the set of team ideas/solutions are better than their own. - During the decisions process, stop thinking and use active listening. - Take decisions based on facts and be a good mediator. - SCRUM rituals and Kanban board are good facilitators to management of these conflicts. - Use checklists with everything to do regarding some tasks or processes. - Escalate a conflict when a new vision is needed.	- Look for process failures in order to protect people. - Make meetings to promote alignment where there is not. - Help Product Owners and Team with non-functional requirements and on future thinking. - Define and structure well the processes with the whole team agreement. - In a DevOps transformation, analyse the old processes and how they will fit in the new DevOps philosophy. - Continuous feedback and improvement of processes. - When new processes are needed, involve the affected people on their definition.	- Explain the advantages of work in a DevOps philosophy. - Avoid splitter terms or words and reinforce the group as a unique team based in mutual help. - Analyse, understand and not judge during the DevOps application. - Use Agile and SCRUM rituals to involve all the people in software development cycle (Dev, Ops, Business, etc..). - Focus on non-functional requirements: type of monitoring, performance, type of performance impact, availability, etc.. - Put someone inside the DevOps teams to evangelize DevOps practices and culture	- Centralization of documentation. - Democratization of the information for the whole company. - Write documentation when the information is historical and/or is important for the future. - Use of a versioning repository to store and manage the documentation. - Use a different repository for clients if it is necessary to provide client documentation. - Use personal or virtual interactions when discussions, doubts, alignment and spreading information rapidly are necessary. - SCRUM rituals are good for information sharing. - Implement redundancy of roles to spread information. - In onboarding processes, create a boot camp, assign a buddy and present information smoothly
Company Board	- Create a Happiness Department to analyse and promote happiness. - Define a top-down strategy to clarify and align all the company toward a direction. - Promote a blameless culture	- No evidence founded.	- No evidence founded.	- Clearly define the DevOps as the only way and guarantee the whole company's alignment.	- No evidence founded.

5.1.6 Saturation

During this research, a saturation point started to emerge during the interviews progress due to repeated topics and the new topics tend to be lesser. As it is possible to see in Table 5, the contribution of the new interviewees with new topics started to decrease. Of course, this study started without a fixed list of available topics to be discussed during the interviews, so, it is not possible to say that all the topics were aborded. However, based on this distribution of the topics over this sample, a repetition of topics is expected. As a result, was chosen to stop after 15 interviews in order to move to conclusions phase (Saunders et al., 2018).

Table 5 – Appearance of new Topics by Studied Area

Interv.	Interpersonal Conflicts	Task Conflicts	Process Conflicts	"Us versus them" Conflicts	Information Sharing	New Topics
I1	- Personalities - Communication - Judgment		- Team Meetings	- Hierarchy - Culture	- Written Documentation - Outdated Documentation - Central Repository - Missing Documentation	10
I2	- Team Spirit - Management - Feedback	- Agile Practices		- Seniority - Team Spirit	- Agile Practices - Frameworks	8
I3	- Happiness Department	- Seniority	- Agile Practices - Transformation Processes - Prioritization		- Sharing Practices - Onboarding	7
I4		- Hierarchy - Communication - Debate				3
I5	- Members Allocation	- Organization	- Defined Processes			3
I6	- Hierarchy - Culture	- Continuous Improvement	- Continuous Improvement	- Roles	- Culture	6
I7				- Organization - Judgment		2
I8		- Management - Decision-making			- Democratization of Information - Management	4
I9		- Requirements				1
I10						0
I11			- Organization - Seniority	- Personalities		3
I12					- Efficiency	1
I13	- Stress - Efficiency			- Communication - Management	- Communication - Team Spirit	6
I14	- Well-being			- Agile Practices - Requirements - Feedback	- Data Access	5
I15	- Geographical Distance	- Transformation Processes				2

Chapter 6 – Conclusion

With this study it is possible to verify that there are conflicts of different types in analysed teams and some problems with information sharing were found, especially with sharing of experiences between the team members.

The main achievements of this study are:

- It was possible to verify the existence of interpersonal, task and process conflicts and difficulties in information sharing in DevOps teams, and understand all the different impacts on the teams
- Some recommendations are indicated as good strategies to better manage the conflicts and information difficulties in DevOps teams
- Different strategies to manage DevOps teams and promote the conditions to achieve better performance were also reported
- There is a perspective of DevOps application in real environments with real problems.

During this research, many limitations were found that deserve some considerations:

- The number of interviews is low and deserves further studies, as well as a low number of participants of the same team
- DevOps is more spread as a concept than in real working teams
- The pandemic situation and the lockdown during this study, which did not allow physical visits and personal interactions with entities, companies and people, in a way that would have been more personal and more inviting. As a result, it was not possible to observe the teams working
- Number of teams and people working in DevOps and announcing themselves as DevOps Engineers. During the interviews period, it was possible to find teams and people that had a different perception and agreement about the DevOps term when compared with the scientific description of Dyck et al. (2015) in section 2.1. For example, in this research a team whose goal was to provide tools and best practices to other company clients toward DevOps implementation was found. As well as this, there was an interviewee with a role called DevOps Engineer, but who operated as the old role of Configuration Manager and was not working in a team with Development and Operations

teams working together. Instead, they worked in a team of the company to provide tools to other teams and help to implement DevOps on those teams. These different points of view of DevOps team organizations reveal a dispersion of the DevOps term. So, in those cases, the interviews were conducted to understand how conflicts and information sharing are managed on teams where interviewees are directly involved in DevOps teams and on the teams where interviewees were helping other teams implementing DevOps

- There is a sub-set of conflict types, many others were not studied
- The Case Study is based in only one approach: interviews.

It is also necessary to highlight that this study does not have measures and metrics that can prove that the recommended strategies have real impact in performance. The strategies are recommended based on interviewees' experience.

In future studies it is recommended to:

- Study other teams, increasing the sample of people and teams involved
- Diversify the methodologies and approaches: surveys, focus group, observation, documentation, etc.
- Study other conflicts and difficulties that teams face and that could impact the effectiveness and performance
- Verify if there are other managing strategies that this study missed
- Implement the mentioned strategies in this study in real teams and verify the impact on effectiveness and performance
- Analyse if there is differences between managing strategies for DevOps teams and managing strategies for generic teams

In conclusion, with all these different situations, the study demonstrates: (1) some enlightenment over the DevOps real application, (2) how the philosophy is spread by the companies, (3) the main difficulties regarding conflicts and information sharing; and (4) the recommended managing strategies to deal with conflicts and information sharing.

Bibliography

A. Brunnert et al. (2015). *Performance-Oriented DevOps: A Research Agenda,*. Standard Performance Evaluation Corp. http://arxiv.org/pdf/1508.04752.pdf.

Alonso, J., Escalante, M., Farid, L., Lopez, M. J., Orue-Echevarria, L., & Dutkowski, S. (2019). Towards supporting the extended devops approach through multi-cloud architectural patterns for design and pre-deployment a tool supported approach. *ICSOFT 2018 - Proceedings of the 13th International Conference on Software Technologies*, *Icsoft*, 813–823. https://doi.org/10.5220/0006856008130823

Amy C. Edmonson. (2012). Teamwork on the fly. *Spotlight on the Secrets of Great Teams*, 72–80.

Balalaie, A., Heydarnoori, A., & Jamshidi, P. (2016). Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture. *IEEE Software*, *33*(3), 42–52. https://doi.org/10.1109/MS.2016.64

Bang, S. K., Chung, S., Choh, Y., & Dupuis, M. (2013). A grounded theory analysis of modern web applications: Knowledge, skills, and abilities for DevOps. *RIIT 2013 - Proceedings of the 2nd Annual Conference on Research in Information Technology*, *October 2013*, 61–62. https://doi.org/10.1145/2512209.2512229

Basirati, M. R., Otasevic, M., Rajavi, K., Böhm, M., & Krcmar, H. (2020). Understanding the relationship of conflict and success in software development projects. *Information and Software Technology*, *126*(April). https://doi.org/10.1016/j.infsof.2020.106331

Beck, K., Beedle, M., Bennekum, A. Van, Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. (2001). *Manifesto for Agile Software Development*. The Agile Alliance. http://agilemanifesto.org/

Betz, C., Olagunju, A. O., & Paulson, P. (2016). The impacts of digital transformation, Agile, and DevOps on future IT curricula. *SIGITE 2016 - Proceedings of the 17th Annual Conference on Information Technology Education*, *September*, 106. https://doi.org/10.1145/2978192.2978205

Castillo-Salinas, L., Sanchez-Gordon, S., Villarroel-Ramos, J., & Sánchez-Gordón, M. (2020). Evaluation of the implementation of a subset of ISO/IEC 29110 Software Implementation process in four teams of undergraduate students of Ecuador. An empirical software engineering experiment. *Computer Standards and Interfaces*, *70*(November 2018), 103430. https://doi.org/10.1016/j.csi.2020.103430

Cois, C. A., Yankel, J., & Connell, A. (2015). Modern DevOps: Optimizing software development through effective system interactions. *IEEE International Professional Communication Conference*, *2015-Janua*. https://doi.org/10.1109/IPCC.2014.7020388

Dennehy, D., & Conboy, K. (2017). Going with the flow: An activity theory analysis of flow techniques in software development. *Journal of Systems and Software*, *133*, 160–173. https://doi.org/10.1016/j.jss.2016.10.003

Dyck, A., Penners, R., & Lichter, H. (2015). Towards definitions for release engineering and DevOps. *Proceedings - 3rd International Workshop on Release Engineering, RELENG 2015*, *May*, 3. https://doi.org/10.1109/RELENG.2015.10

Erich, F., Amrit, C., & Daneva, M. (2014). DevOps Litterature Review. *Product-Focused Software Process Improvement*, *October*, 1–27. https://doi.org/10.1007/978-3-319-13835-0

Farroha, B. S., & Farroha, D. L. (2014). A framework for managing mission needs, compliance, and trust in the DevOps environment. *Proceedings - IEEE Military*

Communications Conference MILCOM, 288–293. https://doi.org/10.1109/MILCOM.2014.54

Gu, M., Yang, L., & Huo, B. (2020). The impact of information technology usage on supply chain resilience and performance: An ambidexterous view. *International Journal of Production Economics*, *October*, 107956. https://doi.org/10.1016/j.ijpe.2020.107956

Gupta, V., Kapur, P. K., & Kumar, D. (2017). Modeling and measuring attributes influencing DevOps implementation in an enterprise using structural equation modeling. *Information and Software Technology*, *92*, 75–91. https://doi.org/10.1016/j.infsof.2017.07.010

Haas, M., & Mortensen, M. (2016). The Secrets of Great Teamwork. *Harvard Business Review*, *13*(June), 70–77.

Hassan, S. Z., & Saeed, K. A. (2003). *A framework for determining IT effectiveness: an empirical approach*. *00*(c), 11. https://doi.org/10.1109/hicss.1999.772780

Hemon, A., Lyonnet, B., Rowe, F., & Fitzgerald, B. (2020). From Agile to DevOps: Smart Skills and Collaborations. *Information Systems Frontiers*, *22*(4), 927–945. https://doi.org/10.1007/s10796-019-09905-1

Hussaini, S. W. (2014). Strengthening harmonization of Development (Dev) and Operations (Ops) silos in IT environment through systems approach. *2014 17th IEEE International Conference on Intelligent Transportation Systems, ITSC 2014*, 178–183. https://doi.org/10.1109/ITSC.2014.6957687

Jedlitschka, A., Kuvaja, P., Kuhrmann, M., Männistö, T., Münch, J., & Raatikainen, M. (2014). Product-Focused Software Process Improvement: 15th International Conference, PROFES 2014 Helsinki, Finland, December 10-12, 2014 Proceedings. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8892*(April 2016). https://doi.org/10.1007/978-3-319-13835-0

Jesse, N. (2019). Agility eats legacy-the long good-bye. *IFAC-PapersOnLine*, *52*(25), 154–158. https://doi.org/10.1016/j.ifacol.2019.12.464

Kitchenham, B. (2007). *Guidelines for performing Systematic Literature Reviews in Software Engineering*. https://doi.org/10.1145/1134285.1134500

Kozlowski SW, I. DR. (2006). Enhancing the effectiveness of work groups and teams. *Psychol. Sci. Public Interest*, *7*, 77–124.

Kropp, M., Meier, A., Anslow, C., & Biddle, R. (2020). Satisfaction and its correlates in agile software development. *Journal of Systems and Software*, *164*, 110544. https://doi.org/10.1016/j.jss.2020.110544

Kumar, R., & Goyal, R. (2020). Modeling continuous security: A conceptual model for automated DevSecOps using open-source software over cloud (ADOC). *Computers and Security*, *97*, 101967. https://doi.org/10.1016/j.cose.2020.101967

L. Bass, I. Weber, and L. Z. (2015). DevOps: A Software Architect's Perspective,. *Addison-Wesley Professional*.

López-Alcarria, A., Olivares-Vicente, A., & Poza-Vilches, F. (2019). A systematic review of the use of Agile methodologies in education to foster sustainability competencies. *Sustainability (Switzerland)*, *11*(10), 1–29. https://doi.org/10.3390/su11102915

Lu, Y., Mao, X., Wang, T., Yin, G., & Li, Z. (2019). Improving students' programming quality with the continuous inspection process: a social coding perspective. *Frontiers of Computer Science*, *14*(5). https://doi.org/10.1007/s11704-019-9023-2

Luz, W. P., 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*, 1–16. https://doi.org/10.1016/j.jss.2019.07.083

Lwakatare, L. E., Kilamo, T., Karvonen, T., Sauvola, T., Heikkilä, V., Itkonen, J., Kuvaja, P., Mikkonen, T., Oivo, M., & Lassenius, C. (2019). DevOps in practice: A multiple case study of five companies. *Information and Software Technology*, *114*(June), 217–230. https://doi.org/10.1016/j.infsof.2019.06.010

Masombuka, T., & Mnkandla, E. (2018). A DevOps collaboration culture acceptance model. *ACM International Conference Proceeding Series*, 279–285. https://doi.org/10.1145/3278681.3278714

Mathieu, J. E., Gallagher, P. T., Domingo, M. A., & Klock, E. A. (2019). Embracing Complexity: Reviewing the Past Decade of Team Effectiveness Research. *Annual Review of Organizational Psychology and Organizational Behavior*, *6*, 17–46. https://doi.org/10.1146/annurev-orgpsych-012218-015106

Melegati, J., Chanin, R., Wang, X., Sales, A., & Prikladnicki, R. (2019). Perceived benefits and challenges of learning startup methodologies for software engineering students. *SIGCSE 2019 - Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 204–210. https://doi.org/10.1145/3287324.3287382

Mishra, A., & Otaiwi, Z. (2020). DevOps and software quality: A systematic mapping. *Computer Science Review*, *38*, 100308. https://doi.org/10.1016/j.cosrev.2020.100308

Nybom, K., Smeds, J., & Porres, I. (2016). On the impact of mixing responsibilities between Devs and Ops. *Lecture Notes in Business Information Processing*, *251*, 131–143. https://doi.org/10.1007/978-3-319-33515-5_11

Saunders, B., Sim, J., Kingstone, T., Baker, S., Waterfield, J., Bartlam, B., Burroughs, H., & Jinks, C. (2018). Saturation in qualitative research: exploring its conceptualization and operationalization. *Quality & Quantity*, *52*(4). https://doi.org/10.1007/s11135-017-0574-8

Schermann, G., Cito, J., Leitner, P., Zdun, U., & Gall, H. C. (2018). We're doing it live: A multi-method empirical study on continuous experimentation. *Information and Software Technology*, *99*(March), 41–57. https://doi.org/10.1016/j.infsof.2018.02.010

Shropshire, J., Menard, P., & Sweeney, B. (2017). Uncertainty, personality, and attitudes toward devops. *AMCIS 2017 - America's Conference on Information Systems: A Tradition of Innovation*, *2017-Augus*(Violino 2016), 1–10.

Thomas, G. (2016). *How To Do Your Case Study* (2nd Editio). Sage.

Wahaballa, A., Wahballa, O., Abdellatief, M., Xiong, H., & Qin, Z. (2015). Toward unified DevOps model. *Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS*, *2015-Novem*, 211–214. https://doi.org/10.1109/ICSESS.2015.7339039

Wen, M., Siqueira, R., Lago, N., Camarinha, D., Terceiro, A., Kon, F., & Meirelles, P. (2020). Leading successful government-academia collaborations using FLOSS and agile values. *Journal of Systems and Software*, *164*, 110548. https://doi.org/10.1016/j.jss.2020.110548

Wettinger, J., & Andrikopoulos, V. (2015). Automated Capturing and Systematic Usage of DevOps Knowledge. *Proceedings of the IEEE …*.

Wiedemann, A., & Schulz, T. (2017). Key capabilities of devops teams and their influence on software process innovation: A resource-based view. *AMCIS 2017 - America's Conference on Information Systems: A Tradition of Innovation*, *2017-Augus*(Mullaguru 2015), 1–10.

Wiedemann, A., & Wiesche, M. (2018). Are you ready for Devops? Required skill set for Devops teams. *26th European Conference on Information Systems: Beyond Digitization - Facets of Socio-Technical Change, ECIS 2018*.