Human Capital and Information Technology Professionals (IJHCITP)

A Systematic Literature Review on DevOps Capabilities and Areas

Daniel Teixeira
Instituto Universitário de Lisboa (ISCTE-IUL)
daniel_s_teixeira@hotmail.com

Ruben Pereira
Instituto Universitário de Lisboa (ISCTE-IUL)
ruben.filipe.pereira@iscte-iul.pt

Telmo Henriques
Instituto Universitário de Lisboa (ISCTE-IUL)
telmo_antonio_henriques@iscte.pt
Miguel Silva
Instituto Universitário de Lisboa (ISCTE-IUL)
marsa@iscte-iul.pt
João Faustino
Instituto Universitário de Lisboa (ISCTE-IUL)
joao_pedro_faustino@iscte-iul.pt

ABSTRACT

Businesses today need to respond to customer needs at unprecedented speed. Driven by this need for speed, many companies are rushing to the DevOps movement. DevOps, the combination of Development and Operations, is a new way of thinking in the software engineering domain that recently received much attention. Since DevOps has recently been introduced as a new term and novel concept, no common understanding of what it means has yet been achieved. Therefore, the definitions of DevOps often are only a part relevant to the concept. This research presents a Systematic Literature Review to identify the determining factors contributing to the implementation of DevOps, including the main capabilities and areas with which it evolves.

Keywords: DevOps, area, capabilities

1. Introduction

Making any change to a business is always a complex task and usually requires investment. As such, whenever an organization adopts any new technology, methodology, or approach, that adoption must be driven by a business need. This, in turn, leads to new working methodologies for IT projects.

Traditional and modern (agile) software development usually focuses exclusively on the software development teams. In either case, once the software has been developed, it is typically handed over to the IT operations team, which takes responsibility for its deployment, ongoing maintenance and support (Jones, Noppen, & Lettice, 2016). The Agile movement has brought together programmers, testers, and business representatives. Conversely, operations teams are isolated groups that ensure stability and enhance performance by applying practices such as the Information Technology Infrastructure Library (ITIL), which equates change to risk (Hüttermann, 2012). For Debois (2011), since both development and operations serve the same customer, the needs of both must be discussed simultaneously. Treated separately, they are like separate trains on separate tracks. No matter how fast they go, they can never meet. Due to this fact, the team commonly works in silos, which leads to a lack of information exchange. Lwakatare, Kuvaja, and Oivo (2015a) say that it is impossible to effectively transmit information about all the releases performed between two different teams in continuous release mode while de França, Jeronimo, and Travassos (2016) report that Development and Operations are left to themselves and will often struggle to talk to each other, much less collaborate, and will remain mired in manual processes. The result is employees who don't work well together, software that doesn't work reliably, and customers that are thinking about moving to competitors (de França et al., 2016).

When two separate teams do Dev and Ops, problems can emerge and lack of synergies can occur. Separations on a technical and organizational level as well as the use of different tools have experienced an increase among Dev and Ops teams (Silva, Faustino, Pereira, & Silva, 2018). This bottleneck between Dev and Ops can affect and/or compromise the product's quality. According to Hussaini (2014), "leveraging the critical success factors to deliver the change must have shared objectives of Dev and Ops.". So, there is a clear disconnect as the two teams speak two different languages and have traditionally been judged according to different reward structures (McCarthy, Herger, Khan, & Belgodere, 2015).

In the midst of such evidence, DevOps emerged. DevOps applies agile and lean principles throughout the entire software supply chain. This allows a business to maximize the speed of delivery of a product or service, from the initial idea to production release and all the way up to customer feedback to improvements based on that feedback (Sharma & Coyne, 2015).

Seeing as DevOps is a recent and novel set of practices for software development (Rajkumar, Pole, Adige, & Mahanta, 2016), there is no one, agreed-upon definition of it. In a more general approach, DevOps integrates a set of characteristics and principles for software delivery that focuses on: speed of delivery, continuous testing in a an environment where production takes place, being ready for shipping at any moment, continuous feedback, the ability to react to change more quickly, and teams working to accomplish a goal instead of a task (no more team boundaries causing a delay) (Sharma & Coyne, 2015). It involves an organizational paradigm shift from distributed siloed groups performing functions separately to cross-functional teams working on continuous operational feature deliveries. Instead of confining themselves to highly arti-

ficial process concepts that will never fly, organizations set up continuous delivery with small upgrades (Ebert, Gallardo, Hernantes, & Serrano, 2016).
Mohamed (2015) argues that this level of integration between development and operations will be revolutionary as releases can be driven by the business need, rather than the operational constraints. Virmani (2015) adds that this approach helps to deliver value faster and continuously, reducing problems caused by misunderstandings between team members, and help to accelerate problem resolution. In another perspective (Hüttermann, 2012), DevOps can be understood as rendering operations more agile.

Currently there is a lack of common understanding of what DevOps means for both academia and the practitioners' communities. This knowledge gap demonstrates that there is still a need for research about the DevOps phenomenon in order to examine how it impacts software development and operations areas. Based on what has been described, there is a clear opportunity to develop a Systematic Literature Review (SLR) on the subject, with the goal of deepening our understanding of what DevOps is.

2. DevOps as a Different Approach

A good cooperation between IT Development and IT Operation teams are viewed to be crucial in order to ensure successful deployment and operations of IT systems (Tessem & Iden, 2008). However, for historical reasons, most IT organizations are characterized by setting clear boundaries between these two teams, which have very different goals, mindsets and cultures (Gazivoda, 2018; Swanson & Beath, 1990).

According to Sharma and Coyne (2015) many organizations are not successful with software projects and their failures are related to the challenges in product development and delivery. Despite this, many companies also find that the development and delivery of software applications are crucial to their business, and that only 25 percent of companies consider their teams to be efficient (Sharma & Coyne, 2015). This gap in efficiency leads to many losses of business opportunities. This demonstrates that even a disruptive methodology can't be perfect for every project.
Given the distinct nature and typology of the functions of each of these teams, it is easy to understand why there are some conflicts when they interact. Such conflicts are essentially related to the different focuses of both teams. Despite actively seeking collaboration from all its stakeholders, most agile projects do not extend themselves to operations people (Diel, Marczak, & Cruzes, 2016). These two teams (Operations and Development) should maintain a close and agile relationship, as it is this relationship which represents the stream of values between the business (where requirements are defined) and the customer (where value is delivered) (Kim, 2015).
However, the relationship between Dev and Ops is not always linear and transparent enough to be able to create synergies capable of overcoming new problems that appear throughout the application's life cycle. While Dev is focused on faster innovation and doing new things, Ops is mainly focused on stability, control, and predictability (Tingley & Anderson, 1986). This cultural difference between the development and operations departments has been reported to lead to conflicts. For example, developers need to get used to operation personnel not having experience with working on projects (Humble & Molesky, 2011). When development and operations are divided into different departments, some processes cross departmental boundaries. This makes it difficult to automate these processes (DeGrandis, 2011). For Debois (2011), despite the fact that both development and operations serve the same customer, the needs of both should be discussed at the same time.

According to Virmani (2015), as part of the Agile transformation in the past few years, IT organizations have introduced Continuous Integration (CI) principles into their software delivery lifecycle, which has improved the efficiency of development teams. Over time, however, it became clear that the optimization resulting from CI was not helping to make the entire delivery lifecycle efficient nor to increase the efficiency of the organization. Unless all the pieces of a software delivery lifecycle work like a well-oiled machine, the efficiency of the delivery lifecycle cannot be optimized.
In order to address the problems between the development and operations teams a new agile approach appeared, namely DevOps. DevOps has been heralded as a novel paradigm to overcome the traditional boundaries between IT Development (Dev) and IT Operations (Ops) teams (Nielsen, Winkler, & Nørbjerg, 2017). According to Riungu-Kalliosaari et al. (2016), DevOps is a set of practices intended to reduce the time between making a change to a system and this change being placed into normal production, all the while ensuring high quality. The main goal associated with this concept is to avoid common problems when operations and developers are kept as separated teams (Bezemer, Eismann, Ferme, & Grohmann, 2018).

DevOps integrates the two worlds of development and operations, using automated development, deployment, and infrastructure monitoring (Ebert et al., 2016). For Sharma and Coyne (2015), because DevOps improves the way that a business delivers value to its customers, suppliers, and partners, it’s an essential business process, not just an IT capability.

3. Research methodology

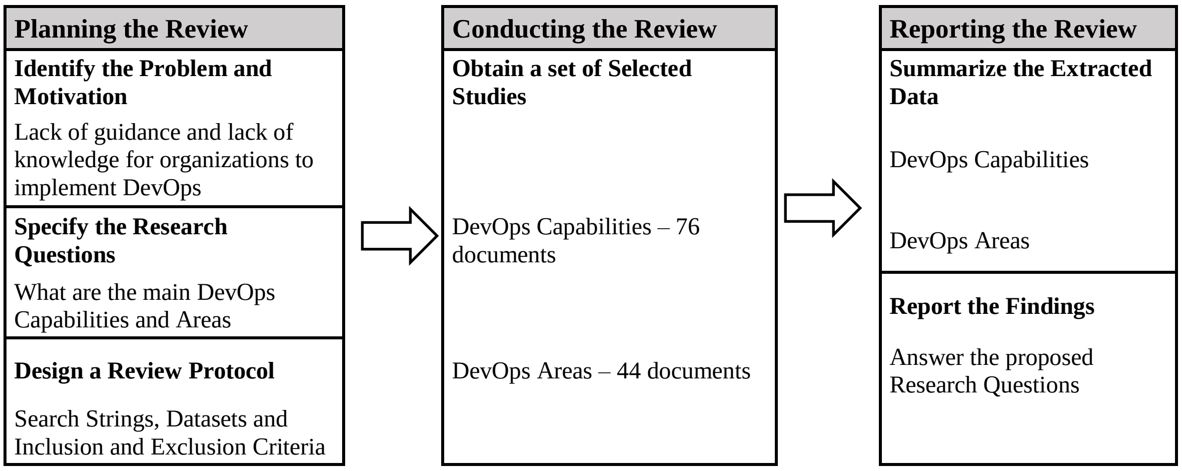
One of the major tools used to support an evidence-based paradigm in other domains is the generation of SLR, which is used to aggregate the experiences gained from a range of different studies in order to answer a specific research question (Khan, Kunz, Kleijnen, & Antes, 2004).

Our research is based on Kitchenhams Procedures for SLR (Kitchenham, 2004), complemented by the centric approach from Webster and Watson (Webster & Watson, 2002), which contains the following steps:

- **Planning.** It is necessary to confirm the need for such a review. In some circumstances systematic reviews are commissioned and in such cases a commissioning document needs to be written. It is also necessary to define the research question(s) that the systematic review will address and produce a review protocol (i.e. plan) which defines the basic review procedures.
- **Conducting.** Apply the review protocol previously designed in order to obtain studies which will be the object of the review.
- **Reporting.** The final phase of a systematic review, which involves writing up the results of the review and circulating these results to potentially interested parties.

The three SLR phases, described above, are represented in Figure 1, and have been specifically adapted to our research purposes.

We chose SLR as our Research Methodology since we wanted to summarize the existing evidence regarding DevOps’ capabilities and areas, with the aim of answering the proposed Research Questions.


Planning the Review
Identify the Problem and Motivation
Lack of guidance and lack of knowledge for organizations to implement DevOps
Specify the Research Questions
What are the main DevOps Capabilities and Areas
Design a Review Protocol
Search Strings, Datasets and Inclusion and Exclusion Criteria
Conducting the Review
Obtain a set of Selected Studies
DevOps Capabilities – 76 documents
DevOps Areas – 44 documents
Reporting the Review
Summarize the Extracted Data
DevOps Capabilities
DevOps Areas
Report the Findings
Answer the proposed Research Questions
Figure 1 - SLR Methodology for this research work

4. Planning the review

This section corresponds to the first step of the SLR Methodology. We begin by providing the Motivation for our work, followed by the Research Questions we aim to address and answer with our research. Finally, we propose our Review Protocol.

4.1. Motivation

The adoption of DevOps drives a challenging cultural shift towards collaboration and knowledge-sharing between software development, quality control and operations (Colomo-Palacios, Fernandes, Soto-Acosta, & Larrucea, 2018). The tremendous growth in demand for DevOps has, however, led to the appearance of new needs. For St, Ab, and Bosch (2017), despite wanting to implement DevOps, many companies find it difficult to understand what DevOps is and what advantages it will have. Furthermore, they ask themselves how to implement DevOps or how can they improve their DevOps practices. According to Rong, Zhang, and Shao (2016), an increasing number of software companies have adopted the DevOps paradigm in order to adapt to the ever-changing business environment.

According to Bucena and Kirikova (2017), many companies miss the maturity of the concept – with no clear definition of DevOps and its practices, no clear goals available and a lack of understanding about development workflow phases and responsibilities. There is both a lack of understanding around DevOps and a clear definition of what it is (Lwakatare, Kuvaja, & Oivo, 2015b). Therefore, organizations are not sure how to effectively implement DevOps capabilities (Qumer Gill, Loumish, Riyat, & Han, 2018).

There are other issues, in addition to the previous ones, that inhibit the adoption of DevOps by companies. As DevOps practices affect the operations and development teams in many ways throughout the software development life-cycle, it requires both cultural and technical transformations (Kamuto & Langerman, 2018). DevOps involves a thorough cultural change. Dev and Ops are traditionally implemented by different organizational structures and are imprinted by different organizational cultures. As such, the DevOps transformation, including not only many technical aspects but also deep cultural issues — especially across two different structures — represents a major challenge (Mikkonen, Lassenius, Männistö, Oivo, & Järvinen, 2018). Lack of trust is another issue. There is both a lack of trust in the idea of DevOps itself as well as in the individuals who promote and work on the DevOps adoption process. This lack of trust results from a lack of understanding or from missing/insufficient communication. It can also be caused by a fear of changes, a fear of potential failure, and a fear of measurements, which could draw one's attention to some unpleasant areas (Bucena & Kirikova, 2017).

The disruptive nature of the changes required to adopt DevOps leads to organizational and business stress. While L. Zhu, Bass, and Champlin-Scharff (2016) consider the organizational strains as being standard for new technologies, for Bucena and Kirikova (2017) the adoption of DevOps is not trivial and can require complex changes in an enterprise's process, organization and workflow. To succeed in adopting DevOps, the enterprises should have an understanding of the different aspects that are related to the DevOps approach and have a well-thought-out strategy. They should start the adoption process with a clear idea of what actions should be performed, how they should be prioritized, what tools could support these actions, and how to measure the success of the adoption process (Bucena & Kirikova, 2017). Adopting DevOps can be affected, both negatively and positively, by an organization's structure. The way an organization is structured may influence DevOps' adoption, for example, when discussing communication, common goals and practices, decision making, and systems thinking within the organization (Lassenius, Dingsøyr, & Paasivaara, 2015).

4.2. Research Questions

In search of a systematic perspective on what is known about DevOps, a focused literature review was undertaken, in order to answer the following research questions:

RQ1. What are the main DevOps areas?

RQ2. What are the DevOps core capabilities?

4.3. Review Protocol

The review protocol starts with a literature search, with the definition of the search string that will be used in the chosen datasets in order to retrieve the maximum number of studies that may address the proposed research questions. The search string which was used and respective datasets are listed below.

Search String.

For DevOps capabilities. *DevOps AND (Capability OR Capabilities OR Practice)*

For DevOps Areas. *DevOps AND (Area, Principles, View, Dimensions and Perspective)*

Datasets. Google Scholar, ScienceDirect, IEEEXplore, ACM.

After that, inclusion and exclusion criteria must be applied to filter the obtained documents. Our criteria are presented in Table 1.

Table 1 - Inclusion and Exclusion Criteria

Inclusion Criteria	Exclusion Criteria
Written in English or Portuguese	*Not written in English or Portuguese*
Scientific papers in conferences or journals and books	*Non-Free documents nor Master Thesis*
Title relevance regarding DevOps	*No title relevance DevOps*

Afterwards, the first set of documents is obtained. Then, in a first phase, the abstracts must be screened to decide their relevance to the research. Finally, these documents are read in order to obtain the final selection of studies to perform the review.

The review protocol is illustrated in Figure 2.

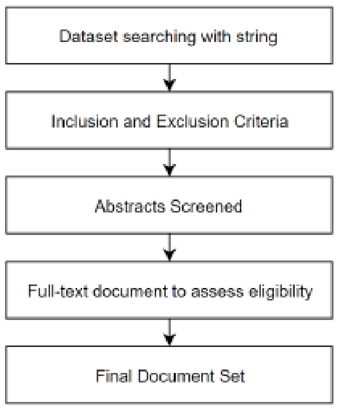

Dataset searching with string
Inclusion and Exclusion Criteria
Abstracts Screened
Full-text document to assess eligibility
Final Document Set

Figure 2 - Review Protocol

For easier understanding of the peers, as well as to add more scientific rigor to our research, the authors decided to follow the centric approach proposed by Webster and Watson (Webster & Watson, 2002).

5. Conducting the Review

This section corresponds to the second step of the SLR Methodology. We start by applying the review protocol previously defined and perform an analysis of the extracted data.

5.1. Selection of Studies

After applying the needed search string in the listed datasets, with the inclusion and exclusion criteria presented in Figure 2, 112 papers were obtained, excluding duplicates.

Afterwards, the abstracts were read to further determine the documents' relevance. This resulted in 82 documents, which were, in turn, individually read. As a result of this process, 76 relevant studies were obtained for our research.

Erro! A origem da referência não foi encontrada.3 shows the number of papers found. As shown**Erro! A origem da referência não foi encontrada.**, the search conducted aims to find all papers in which DevOps capabilities has been mentioned.

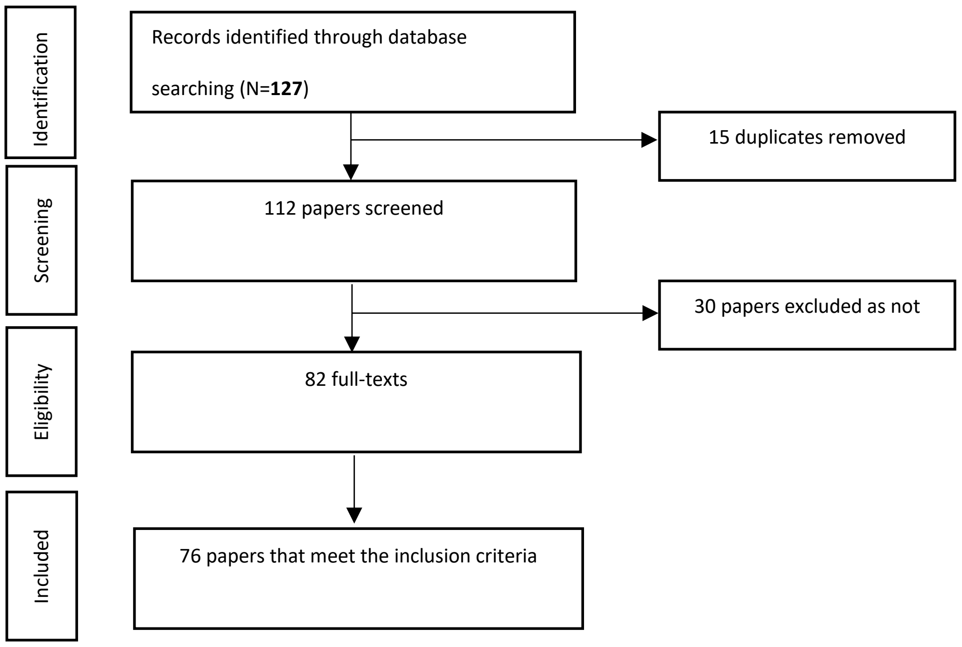

Identification
Records identified through database searching (N=**127**)
15 duplicates removed
Screening
112 papers screened
30 papers excluded as not
Eligibility
82 full-texts
Included
76 papers that meet the inclusion criteria

Figure 3 - Search strings, databases used and results from search conducted for DevOps capabilities

After applying the needed search string in the listed datasets, with the inclusion and exclusion criteria presented in Table 1, 82 papers were obtained, excluding duplicates.

Afterwards, the abstracts were read to further decide the documents' relevance. This resulted in 46 documents. Each one of these documents was rea and 44 relevant studies were obtained for our research.

Figure 4**Erro! A origem da referência não foi encontrada.** shows the number of papers found per database. As shown in Figure 4**Erro! A origem da referência não foi encontrada.**, the search conducted aims to find all papers in which DevOps areas have been mentioned.

5.2. Data Extraction Analysis

Looking at Figure 5 it is possible to see the distribution over the years of the articles which deal with DevOps capabilities. In 2011, only two capabilities were related with DevOps. Since then, there has been an increase in the number of documents and capabilities. This can be explained by the fact that DevOps gained popularity and the increase of interest over time would be expected, something that is reflected in the published articles. Since 2015, the quantity of documents rose slightly and in 2016 interest grew exponentially.

It is also possible to see that the interest in Continuous Integration and Continuous Deployment documents has remained above the interest in the rest of the capabilities over the years.

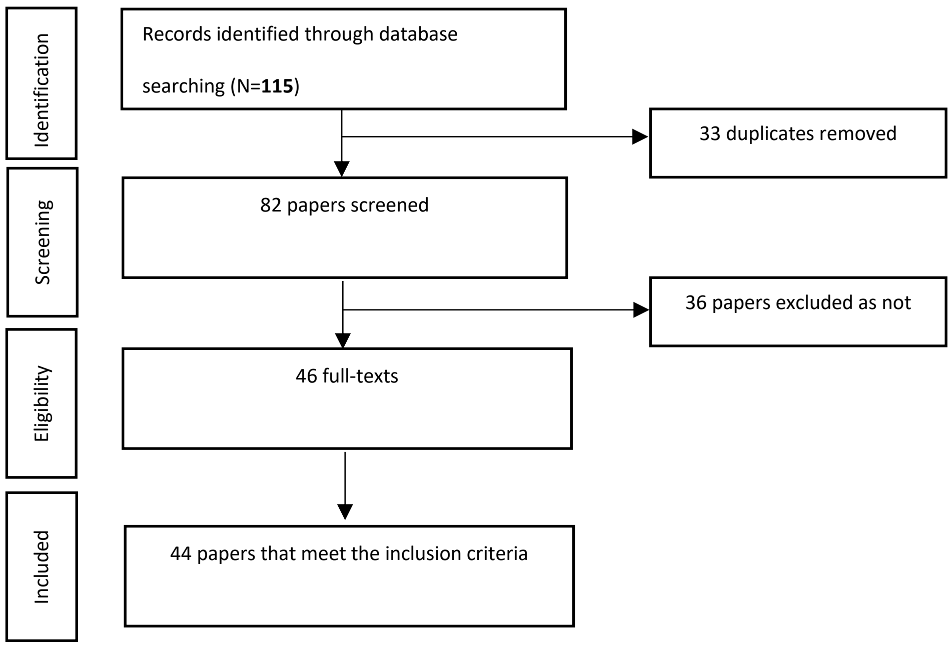

Identification
Screening
Eligibility
Included
Records identified through database searching (N=**115**)
33 duplicates removed
82 papers screened
36 papers excluded as not
46 full-texts
44 papers that meet the inclusion criteria

Figure 4 - Search strings, databases used and results from search conducted for DevOps areas

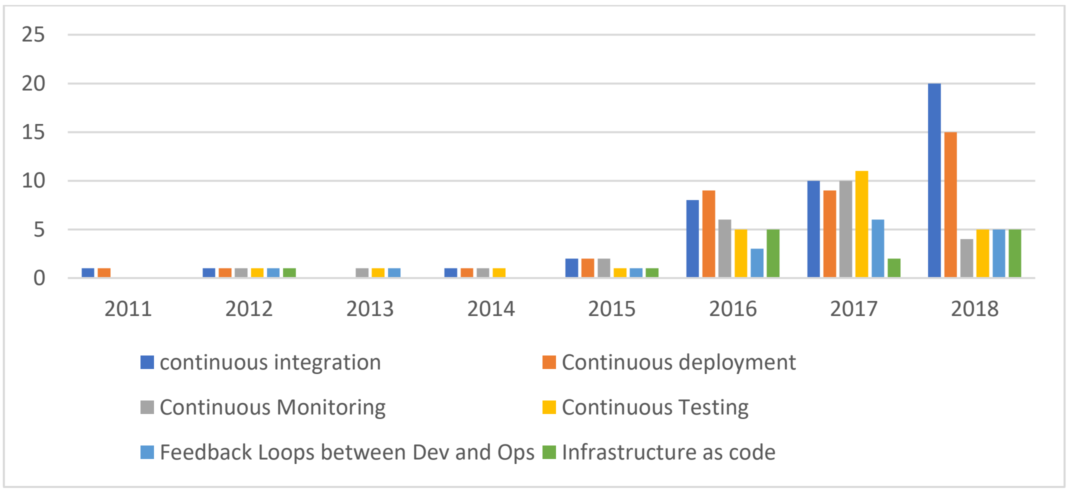


Figure 5 – DevOps Capabilities Articles Distribution per year

Looking at Figure 6, it is possible to see the distribution of the articles dealing with DevOps areas in the last years. As was pointed out in Figure 5, one can verify by looking at Figure 6 that interest in DevOps grew in 2015 and in 2016 it grew exponentially. Since then, the level of interest seems to have stabilized. The top area changed in 2018 but Culture is one of the most consistent areas and has generated more interest in recent years.

Measurement, Sharing and Automation have maintained the same level of interest in the past three years, while the interest in Technology, People and Process decreased in 2018 to half of what it was in 2017.

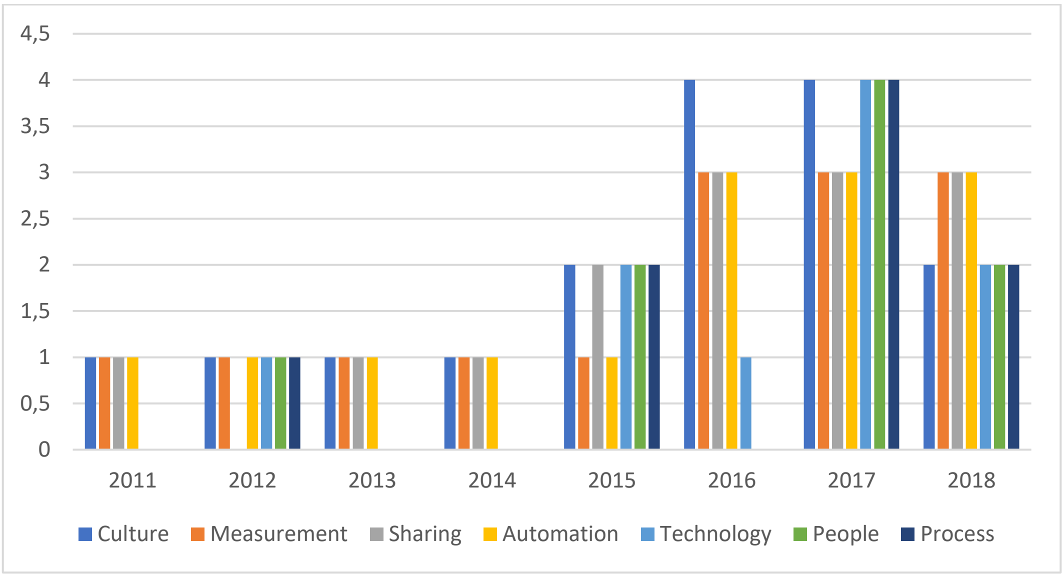


Figure 6 - DevOps Areas Articles Distribution per year

6. Reporting the Review

This section corresponds to the third and last step of the SLR Methodology, where we summarize the extracted data from the selected studies. We have identified two main topics, which integrate the following sub-sections: DevOps Capabilities and DevOps Areas.

6.1. DevOps Capabilities

A recent study was published (Jabbari, bin Ali, Petersen, & Tanveer, 2016a) where the authors have synthesized the practices that DevOps practitioners have applied so far (Table 2**Erro! A origem da referência não foi encontrada.**). Since this study seems to be complete and the authors did not find a single DevOps' practice that was not included in Jabbari's list, the authors decided to use this list assuming that is the most completed collection of DevOps practices among the literature. Other studies related to DevOps capabilities can be found among the literature - in Hüttermann (2012); Sharma (2017a), Punjabi and Bajaj (2017), Soni (2016), and Stoneham et al. (2017). However, they are not as exhaustive as the one presented in Table 2.

Within these studies, a capability can be written in a different way, depending on its context, but maintain the same meaning. As such, the authors have grouped these capabilities together by using vectors and basing such groupings on what they have understood of the meaning of the capability. Table 2 shows the grouping that the authors made for these vectors. Although the study was already quite complete, the authors decided to carry out a literature review that could corroborate these abilities presented in the study of Jabbari et al. (2016a).

Having analyzing Table 2 and seeing that there is a considerable gap between C6 and C7, the authors decided to describe all the capabilities from C1 and C6. The description of each capacity is presented considering the various definitions which have been found.

6.1.1. Continuous Integration

The Continuous Integration (CI) concept was first practiced and described as “doing everything in parallel, with frequent synchronizations” in the 1998 book *Microsoft Secrets* (Pang & Hindle, 2017). CI consists of established practices in modern agile software development (Steffens, Lichter, & Döring, 2018a). It accommodates rapid changes (Bai, Li, Pei, Li, & Ye, 2018a) and is widely considered to be the best in software development (Debroy, Miller, & Brimble, 2018). Developers integrate their work frequently (usually each person integrates at least daily), leading to multiple integrations per day (Jabbari et al., 2016a; St et al., 2017). For Sharma and Coyne (2015), continuous integration ensures that each team’s work is continuously integrated with that of other development teams and then validated. Continuous integration, thereby, reduces risk and identifies issues earlier in the software development life cycle.

Implementing CI this way ensures that bugs are caught earlier in the development cycle, which makes them less expensive to fix. Automated tests are run for every build, in order to ensure that builds maintain a consistent quality. The main objective of Continuous integration is to foster discussion and fast validation by peers (De Bayser, Azevedo, & Cerqueira, 2015). As Continuous Integration allows developers to immediately see the impact of their code changes and fix problems on the spot in the development environment, it became one of the major points of interest in the DevOps movement as smaller and more frequent changes reduced merge and integration issues (Debois, 2011).

6.1.2. Continuous Deployment

DevOps emphasizes the use of Continuous Deployment, which means deploying a number of smaller changes as soon as they are released, instead of waiting until a ‘full package’ of changes is ready, and follows directly from the practice of frequent releases. (Nielsen et al., 2017)**.** This allows users to benefit from the changes much earlier and developers to see whether their changes work in practice (Feitelson, Frachtenberg, & Beck, 2013). To (Düllmann, Paule, & Van Hoorn, 2018) one important DevOps practice is the usage of continuous deployment as it helps to automate many steps, ranging from a source code commit to the deployment of a software artifact to production. When commonly adopted, continuous integration and continuous deployment can cause the software development lifecycle to shorten (Tuma, Calikli, & Scandariato, 2018). For Debois (2011), this capability is just like exercise: “the more you practice deployment to production, the better you will get at it.

The implementation of continuous deployment should also reduce the effort required in order to carry out a task. Many of the tasks related to the release of DevOps are being automated, and manual tasks such as configurations are being dealt with automatically. As such, the pool of resources can be released immediately after the task is completed (Kuusinen et al., 2018). There is a strong relationship between the quality of the software developed and the agility of the organization to the DevOps practices of software development. Therefore, DevOps practices contribute to the enhancement of these software quality attributes within a continuous deployment process (I. D. Rubasinghe et al., 2017).

6.1.3. Continuous Monitoring

Continuous Monitoring collects data and metrics that come from the different stages of the application lifecycle, allowing all involved parties to react quickly in order to improve or modify the functionalities which are being used (Debois, 2011; Sharma & Coyne, 2015). Effective monitoring is essential to allow DevOps teams to deliver at speed, to get feedback from production, and to increase customers' satisfaction, acquisition and retention. By aligning development of monitoring with the development of the whole solution (implementing functional and nonfunctional requirements, building up the application, middleware, infrastructure), they will be able to improve monitoring continuously, to catch gaps in monitoring early, and to ensure that monitoring is always aligned with concrete needs (Hüttermann, 2012).

One of the major contributions is that continuous monitoring may enable early detection of quality-of-service problems, such as performance degradation, and also the fulfillment of service level agreements (Fitzgerald & Stol, 2014).

6.1.4. Continuous Testing

Continuous Testing means to test as soon as possible and continuously during the development lifecycle, leading to a development cost reduction as well as to a better software quality. This practice is viable using techniques such as test automation and virtualization, in order to simulate the production environments in which the tests are to be executed and in a scenario that is as realistic as possible (Sharma & Coyne, 2015; Soni, 2016). Also for Sharma and Coyne (2015), continuous testing is known as "shift-left testing", which stresses integrating development and testing activities to ensure that quality is built in as early as possible in the life cycle and nothing is left behind to later instances.

The importance of this capability is that the benefits of Continuous Testing will eventually increase customer satisfaction, as the customer has a larger and more immediate impact on the product. Because the continuous deployment pipeline relies heavily on testing, the quality of the system will improve over time, as fewer bugs are introduced into the system (Kuusinen et al., 2018). This capability also permits a reduction in overall costs, shortens later testing cycles and ensures continuous feedback on quality (Nielsen et al., 2017).

Table 2 - DevOps capabilities literature review

ID	Capabilities	Reference	# of References
C1	*Continuous Integration*	(Bai, Li, Pei, Li, & Ye, 2018b; Bucena & Kirikova, 2017; Chen, Kazman, Haziyev, Kropov, & Chtchourov, 2015; Cleveland et al., 2018; Colomo-Palacios et al., 2018; Croker & Hering, 2016; De Bayser et al., 2015; de França et al., 2016; Debois, 2011; Debroy et al., 2018; Düllmann et al., 2018; Fitzgerald & Stol, 2014; Hüttermann, 2012; Jabbari, bin Ali, Petersen, & Tanveer, 2016b; Kuusinen, Balakumar, Jepsen, & Larsen, 2018; Laukkarinen, Kuusinen, & Mikkonen, 2017, 2018; Lewerentz et al., 2018; Mackey, 2018; Mansfield-Devine, 2018; Marijan, Liaaen, & Sen, 2018; Mohan & Ben Othmane, 2016; Molto, Caballer, Perez, Alfonso, & Blanquer, 2017; Moore et al., 2016; Palihawadana et al., 2017; Pang & Hindle, 2017; Punjabi & Bajaj, 2017; Rahman, Mahdavi-Hezaveh, & Williams, 2018; Rodríguez et al., 2018; I. D. Rubasinghe, Meedeniya, & Perera, 2017; I. Rubasinghe, Meedeniya, Perera, & Practice, 2018; Shahin, Babar, & Zhu, 2016; Sharma, 2017a; Shivakumar, 2017; Snyder & Curtis, 2017; Soni, 2016; Steffens et al., 2018a; Stoneham et al., 2016; Tuma et al., 2018; Vassallo et al., 2017; Wiesche, 2018; Wongkampoo & Kiattisin, 2018; Xia, Zhang, Wang, Coleman, & Liu, 2018; H. Zhu & Bayley, 2018)	*44*
C2	*Continuous Deployment*	(Ali, Caputo, & Lawless, 2017; Bass, 2017; Bhattacharjee, Barve, Gokhale, & Kuroda, 2018; Bucena & Kirikova, 2017; Chen et al., 2015; Cleveland et al., 2018; Debois, 2011; Debroy et al., 2018; Düllmann et al., 2018; Farshchi, Schneider, Weber, & Grundy, 2015; Fitzgerald & Stol, 2014; Fördős & Cesarini, 2016; Hüttermann, 2012; Jabbari et al., 2016b; Karapantelakis et al., 2016; Kuusinen et al., 2018; Laukkarinen et al., 2018; Mackey, 2018; Mansfield-Devine, 2018; Mohan & Ben Othmane, 2016; Palihawadana et al., 2017; Pang & Hindle, 2017; Perera, Bandara, & Perera, 2017; Punjabi & Bajaj, 2017; Rahman et al., 2018; Rana & Staron, 2016; I. D. Rubasinghe et al., 2017; I. Rubasinghe et al., 2018; Shahin et al., 2016; Sharma, 2017a; Shivakumar, 2017; Soni, 2016; Steffens, Lichter, & Döring, 2018b; Steffens et al., 2018a; Stoneham et al., 2016; Tuma et al., 2018; Ur Rahman & Williams, 2016b; Wiesche, 2018; Xia et al., 2018; H. Zhu & Bayley, 2018)	*40*
C3	*Continuous Monitoring*	(Bai et al., 2018b; Bucena & Kirikova, 2017; Chen et al., 2015; de França et al., 2016; Debois, 2011; Düllmann et al., 2018; Fitzgerald & Stol, 2014; Hanappi, Hummer, & Dustdar, 2016; Hüttermann, 2012; John et al., 2015; Karapantelakis et al., 2016; Kuusinen et al., 2018; Li, Zhang, & Liu, 2017; Pang & Hindle, 2017; Perera, Bandara, et al., 2017; Rana & Staron, 2016; Roche, 2013; I. D. Rubasinghe et al., 2017; Rufino, Alam, & Ferreira, 2017; Sharma, 2017a; Shivakumar, 2017; Snyder	*26*

		& Curtis, 2017; Soni, 2016; Steffens et al., 2018b; Ur Rahman & Williams, 2016b; Vassallo et al., 2017)	
C4	*Continuous Testing*	(Bucena & Kirikova, 2017; Chen et al., 2015; Croker & Hering, 2016; de Feijter, Rob, Jagroep, Overbeek, & Brinkkemper, 2017; Debois, 2011; Fitzgerald & Stol, 2014; Hüttermann, 2012; Jabbari et al., 2016b; Kuusinen et al., 2018; Nielsen et al., 2017; Palihawadana et al., 2017; Pang & Hindle, 2017; Punjabi & Bajaj, 2017; Roche, 2013; I. Rubasinghe et al., 2018; Samarawickrama & Perera, 2018; Shahin et al., 2016; Sharma, 2017a; Shivakumar, 2017; Silva et al., 2018; Snyder & Curtis, 2017; Soni, 2016; St et al., 2017; Stoneham et al., 2016; Vassallo et al., 2017; Wiesche, 2018)	*26*
C5	*Feedback Loops between Dev and Ops*	(Bucena & Kirikova, 2017; de Feijter et al., 2017; Debois, 2011; Debroy et al., 2018; Hanappi et al., 2016; Hüttermann, 2012; Jabbari et al., 2016b; John et al., 2015; Kuusinen et al., 2018; Mikkonen et al., 2018; Nielsen et al., 2017; Pang & Hindle, 2017; Roche, 2013; Sharma, 2017a; Silva et al., 2018; St et al., 2017; Stoneham et al., 2016; Wongkampoo & Kiattisin, 2018)	*18*
C6	*Infrastructure as code*	(Bhattacharjee et al., 2018; Bucena & Kirikova, 2017; De Bayser et al., 2015; de França et al., 2016; Debois, 2011; Debroy et al., 2018; Düllmann et al., 2018; Fördős & Cesarini, 2016; Hüttermann, 2012; Jabbari et al., 2016b; Jimenez et al., 2017; Rahman et al., 2018; Rana & Staron, 2016; Shahin et al., 2016; Sharma, 2017a; Steffens et al., 2018b, 2018a)	*17*
C7	*Change Management*	(Abdelkebir, Maleh, & Belaissaoui, 2017; Debois, 2011; Hüttermann, 2012; Jabbari et al., 2016a; Mohamed, 2015; I. D. Rubasinghe et al., 2017; Science & Sciences, 2015; Sharma, 2017b; H. Zhu & Bayley, 2018)	*9*
C8	*Continuous planning*	(Fitzgerald & Stol, 2014; Hüttermann, 2012; Jabbari et al., 2016a; Kuusinen et al., 2018; Pang & Hindle, 2017; Sharma, 2017b; Ur Rahman & Williams, 2016a)	*7*
C9	*Prototyping application*	(Cleveland et al., 2018; De Bayser et al., 2015; Fitzgerald & Stol, 2014; Hüttermann, 2012; Jabbari et al., 2016a; Sharma, 2017b)	*6*
C10	*Process Standardization*	(Hüttermann, 2012; Jabbari et al., 2016a; Rana & Staron, 2016; Roche, 2013; Sharma, 2017b)	*5*
C11	*Stakeholder Participation*	(Hüttermann, 2012; Jabbari et al., 2016a; Sharma, 2017b)	*3*
C12	*Shift Left*	(de Feijter et al., 2017; Hüttermann, 2012; Sharma, 2017b)	*3*

6.1.5. Continuous Feedback

The goal of this practice is to get as much feedback as possible in order to perform the necessary corrections. Continuous feedback is developer – focused, which means that feedback relates to coding or architectural problems, build failures, test status and uploads of file releases(L. Zhu et al., 2016).

The new technologies provide the ability to monitor customer behavior, which allows the business team or any other interested parties to take the necessary actions to improve the software (Silva et al., 2018). Monitoring information and user feedback can be used for the purpose of improving the application and thereby enhancing the customer experience (Nielsen et al., 2017).

6.1.6. Infrastructure as code

Infrastructure as code involves fast scaling up and down of infrastructure on demand, treating the configuration code in the same way as the application code (Rana & Staron, 2016). It also emphasizes developing automation logic for deploying, configuring and upgrading software and infrastructure repeatedly and quickly, particularly in a cloud environment (Lwakatare et al., 2015b).

Teams avoid manual environmental configuration and enforce consistency through code to represent the desired state of their environments. Deployment of infrastructure as code is repeatable and prevents runtime problems due to configuration drift or lack of dependency. DevOps teams can work with a unified set of practices and tools to deliver applications and infrastructure support quickly, reliably and on a scale. The use of infrastructure as code was recurrently cited as a means of guaranteeing that everyone knows how the execution environment of an application is provided and managed (Luz, Pinto, & Bonifácio, 2018).

6.2. DevOps Areas

This section presents the findings from a thorough literature analysis aiming to find the DevOps dimensions that characterize this phenomenon. They are either categories that work as DevOps enablers or are expected outcomes of a DevOps adoption process. Table 3 presents the main findings related to DevOps dimensions.

Because there is no standard definition of DevOps and its related processes (Silva et al., 2018) and little has thus far been presented in order to describe and formalize what it constitutes (Lwakatare et al., 2015b) the authors will now go on to detail the areas that best define DevOps practices.

Having analyzing Table 3 and determining that there is a considerable gap between A7 and A8, the authors have decided to describe all the areas from A1 to A6. The description of each capacity will be presented considering the various definitions which have been found.

6.2.1. Culture

In DevOps, there is a culture of collaboration between the software development organization and the operations organization (Lwakatare et al., 2015b) where there is joint responsibility for the delivery of high quality software (Colomo-Palacios et al., 2018). For de França, Jeronimo, and Travassos (2016) the so-called DevOps culture recognizes trust as a relevant characteristic for influencing organizational change. The culture aims to change the dynamics in which development and operational teams interact, highlighting the tasks between design and operation, such as operational design, test-driven development and continuous integration (Diel et al., 2016).

The DevOps culture encourages small, multidisciplinary teams that work independently and collectively to take responsibility for the experience of actual users of their software (Sharma & Coyne, 2015). There's no place like production for a DevOps team. All they do is improve the live experience of customers. There are no silos and no blame-game, because the team is responsible for each other. DevOps teams stress being able to move fast, understand the impact and react quickly (Hüttermann, 2012).

6.2.2. Measurement

The ability to measure the development process by incorporating different metrics will help increase efficiency in product development (Lwakatare et al., 2015b). Based on data rather than instinct, decisions lead to an objective and irreproachable path to improvement. The data should be transparent, accessible to everyone, meaningful and capable of being viewed ad hoc. Furthermore, measurement includes monitoring high-level business metrics such as revenue or end-to-end transactions per unit time (Debois, 2011).

At a lower level, it requires careful choice of key performance indicators, since people change their behavior according to how they are measured (Nielsen et al., 2017). DevOps use various forms of measurements and monitoring which include business metrics (e.g. revenue) to metrics for a technical overview (Rana & Staron, 2016)

Table 3 - DevOps Areas Literature Review

ID	**Area**	**References**	**# of References**
A1	*Culture*	(Bang, Chung, Choh, & Dupuis, 2013; Bucena & Kirikova, 2017; Colomo-Palacios et al., 2018; de França et al., 2016; Debois, 2011; Diel et al., 2016; Erich, Amrit, & Daneva, 2014; Gupta, Kapur, & Kumar, 2017; Hüttermann, 2012; Jabbari et al., 2016b; Lassenius et al., 2015; Nielsen et al., 2017; Perera, Silva, & Perera, 2017; Rana & Staron, 2016; Sharma & Coyne, 2015; Silva et al., 2018)	*16*
A2	*Measurement*	(Bang et al., 2013; Colomo-Palacios et al., 2018; de França et al., 2016; Debois, 2011; Erich et al., 2014; Gupta et al., 2017; Hüttermann, 2012; Jabbari et al., 2016b; Lassenius et al., 2015; Luz et al., 2018; Nielsen et al., 2017; Perera, Silva, et al., 2017; Rana & Staron, 2016; Silva et al., 2018)	*14*
A3	*Sharing*	(Bang et al., 2013; Colomo-Palacios et al., 2018; de França et al., 2016; Debois, 2011; Erich et al., 2014; Gupta et al., 2017; Hüttermann, 2012; Jabbari et al., 2016b; Lassenius et al., 2015; Luz et al., 2018; Nielsen et al., 2017; Perera, Silva, et al., 2017; Rana & Staron, 2016; Silva et al., 2018)	*14*
A4	*Automation*	(Bang et al., 2013; Colomo-Palacios et al., 2018; de França et al., 2016; Debois, 2011; Erich et al., 2014; Gupta et al., 2017; Hüttermann, 2012; Jabbari et al., 2016b; Lassenius et al., 2015; Luz et al., 2018; Mohamed, 2015; Nielsen et al., 2017; Perera, Silva, et al., 2017; Rana & Staron, 2016; Silva et al., 2018)	*14*
A5	*Technology*	(Abdelkebir et al., 2017; Bucena & Kirikova, 2017; Diel et al., 2016; Gazivoda, 2018; Hussain, Clear, & MacDonell, 2017; Hüttermann, 2012; McCarthy et al., 2015; Sharma & Coyne, 2015; Silva et al., 2018; Sturm, Pollard, & Craig, 2017)	*10*
A6	*People*	(Abdelkebir et al., 2017; Bucena & Kirikova, 2017; Gazivoda, 2018; Hussain et al., 2017; Hüttermann, 2012; McCarthy et al., 2015; Sharma & Coyne, 2015; Silva et al., 2018; Sturm et al., 2017)	*9*
A7	*Process*	(Abdelkebir et al., 2017; Bucena & Kirikova, 2017; Gazivoda, 2018; Hussain et al., 2017; Hüttermann, 2012; McCarthy et al., 2015; Sharma & Coyne, 2015; Silva et al., 2018; Sturm et al., 2017)	*9*
A8	*Quality*	(Erich et al., 2014; Luz et al., 2018; Mohamed, 2015)	*3*
A9	*Collaboration*	(Luz et al., 2018; Mohamed, 2015)	*2*
A10	*Do it yourself deployments*	(Debois, 2011)	*1*
A11	*Agility*	(Luz et al., 2018)	*1*
A12	*Resilience*	(Luz et al., 2018)	*1*
A13	*Transparency*	(Luz et al., 2018)	*1*
A14	*Services*	(Erich et al., 2014)	*1*
A15	*Structures*	(Erich et al., 2014)	*1*
A16	*Standards*	(Erich et al., 2014)	*1*
A17	*Governance*	(Mohamed, 2015)	*1*

6.2.3. Automation

It is believed that manual, and repetitive tasks can be automated to reduce unnecessary effort and improve software delivery. Hence, automation would improve not only the delivery speed, but also the infrastructure consistency, productivity of teams, and repeatability of tasks (de França et al., 2016). Automation is used not just to save time, but it also prevents defects, creates consistency, and enables self-service. Automation is one of the main areas of DevOps: it allows for capabilities such as continuous integration and continuous deployment (Mohamed, 2015). Although transparency and sharing can be used to ensure collaboration even in manual tasks, with automation the points where silos may arise are minimized (Luz et al., 2018).

6.2.4. Technology

Technology enables people to focus on high-value creative work while delegating routine tasks to automation. Technology also allows teams of practitioners to leverage and scale their time and abilities (Sharma & Coyne, 2015).

A technology stack and tools are used to quickly and reliably operate and develop applications. These tools also help engineers carry out tasks independently (e.g. code deployment and infrastructure supply), which would normally require the assistance of other teams, and this further increases the speed of the team (Hüttermann, 2012).

6.2.5. People

The relations between colleagues should be based on trust and confidence. Transparency should be considered the rule of thumb for a DevOps team. The members of the team should also have common goals and incentives. and not only developers for delivering in time (Silva et al., 2018). For Sharma & Coyne (2015), people are the main characters of DevOps culture.

6.2.6. Process

The DevOps process can be considered a business process because it aims to affect the entire lifecycle of an application as being a collection of activities or tasks that produce a specific result for customers (Hüttermann, 2012). When the DevOps approach is in place within an organization, all parties involved from the highest level of the business down to the operations should be able to have transparency and cooperate in the entire lifecycle of a change (Silva et al., 2018).

6.3. Relationship between DevOps Capabilities and Areas

The authors decided to relate the articles of the DevOps capabilities and areas to see if there is any relationship between the two types of concepts. By analyzing Figure 7, the relationship between DevOps and the two concepts can be seen.

Unsurprisingly, some of the areas relate more to certain capabilities. For example, the articles that reference process as a DevOps area, also relate it to Feedback Loops. Culture seems to be one of the major areas, as it is the one that most relate to all capabilities. Measurement, Sharing and Automation are the areas that relate more to ability, after culture. Culture is the area that most relates to all the capabilities. Every area seems to have an impact on Feedback Loops as it influences almost all areas with the same weight. Sharing and Automation are areas that also have an important relationship with Continuous Deployment. Except in the Feedback Loops, Process is the less capability influencer area. Automation and Sharing does have an important share in all the capabilities as these areas keep a high number of documents that relate them to most of the practices. Sharing, Automation and Culture are also areas that seems to have influence in Continuous Integration.

We shall take a closer look at the areas and the capabilities that they most relate to: Culture seems to be the most related area of Continuous Testing, Continuous Integration and Feedback Loops. Measurement is significantly an influencer for Continuous Testing, Continuous Integration and Feedback Loops. Sharing and Automation are areas that seem to both relate the same with Feedback Loops, Continuous Testing, Continuous Deployment and Continuous Integration. Technology, People and Process have a close relationship with Feedback Loops and Continuous Testing.

Furthermore, it is possible to analyze the capabilities and the areas that they most relate to, and some conclusions can be reached by analyzing Figure 7. Continuous Integration and Continuous Deployment, Continuous Monitoring and Continuous Testing relate the most to culture, Measurement, Sharing and Automation. Feedback Loops seems to only have a closer relation with Culture and Infrastructure as code seems to be influenced by Culture, Measurement and Sharing.

By this analysis, it became clear that some areas tend to relate more with some capabilities than with others. One of the main reasons may be due to the fact that some areas function as an enabler for the specific capability practitioners.

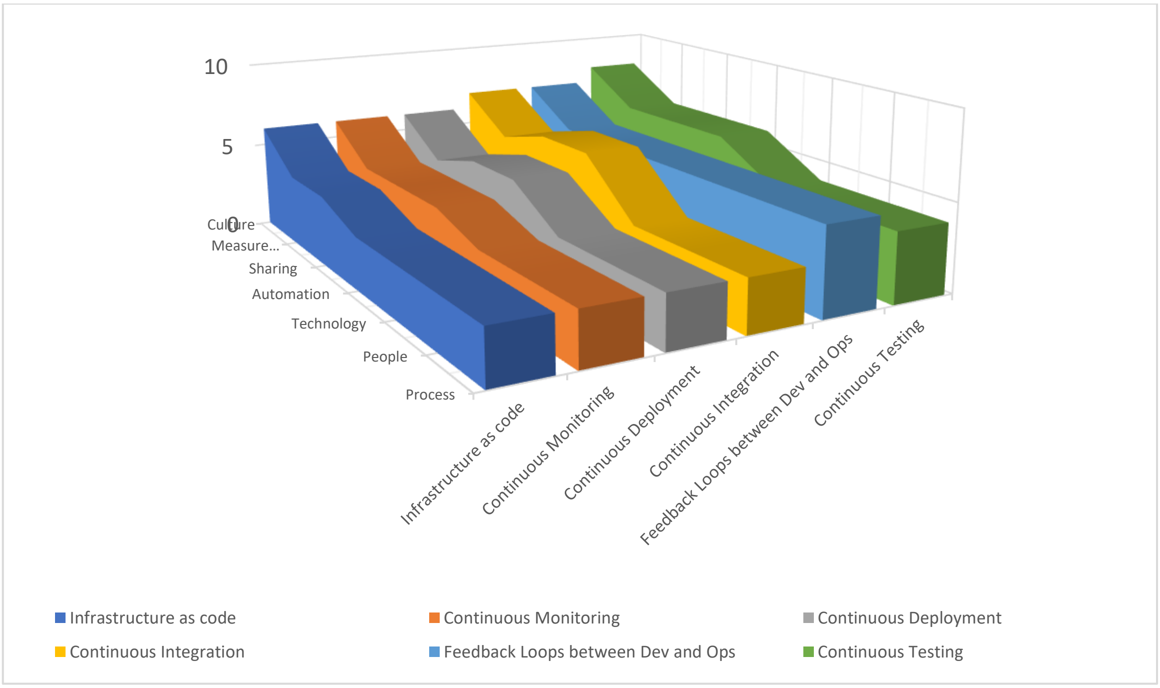


Figure 7 - Relationship between DevOps Capabilities and Areas

7. Conclusion

In this research an SLR was conducted to respond to the call by researchers and practitioners for a deeper theoretical and practical understanding of DevOps capabilities and areas that could work as determinant factors and contribute to the implementation of DevOps. Basing themselves on the previous sections the authors are able to argue that a proper answer has been provided for each proposed Research Question.

Regarding RQ1, the main DevOps areas were elicited and described, and they specifically include culture, measurement, sharing, automation, technology, people and process. Concerning RQ2, the main DevOps capabilities have been also identified and detailed. The elicited capabilities include continuous integration, continuous deployment, continuous testing, feedback loops between Dev and Ops and infrastructure as code.

As the number of DevOps practitioners continues to increase, the studies that focus on DevOps areas and its capabilities, have also increased since 2015. The authors also identified some relationships between the DevOps areas and capabilities based on the analysis of Figure 7. The documents that focus on the DevOps culture are most likely to relate it to all of the main capabilities found. On the other hand, it is more difficult to find a document that relates Technology, People and Process with the main capabilities.

The capabilities of Continuous Integration and Continuous Deployment are the more relevant in the literature. The areas that most relate with them are Culture, Sharing and Automation. These three areas are the most referred DevOps areas in the literature. Processes seems to be the area that less influences the capabilities, while Infrastructure as Code is the capability which the fewest studies tend to relate with DevOps.

This research aimed to find the main DevOps capabilities and areas. This research has brought contributions to the academic and scientific community by exploring a field that had not yet been explored. It has also improved the knowledge base and endeavored to lay down new bases for further research.

This research is a new systematized contribution to knowledge, through the identification of patterns that have been recognized in the literature - and that, as such, corresponds to a new level of knowledge in the approach to the topic. This research also provides some contributions for professionals and practitioners. In the absence of studies exploring the DevOps main capabilities and DevOps areas, and even the relationship between them, this research brings new insights on how and why practitioners should adopt DevOps practices and which areas they have to change or, at least, keep in mind as being relevant for an effective adoption of DevOps.

Regarding limitations, we were not able to gather enough information and present a robust conclusion regarding specific topics, such as Outcomes, since DevOps is a recent subject. The present research cannot fully avoid biases since we excluded literature written in other languages or unavailable in electronic databases.

In the future, research should be carried out into the most referenced capabilities, Continuous Integration and Continuous Deployment and the most referenced areas, Culture, Sharing and Automation, as they seem to be essential in the DevOps movement. Also, it would be interesting to deeply explore the relationship between Continuous Integration and Culture, Sharing and Automation, as these areas seem to relate the most with the main capability found among this literature review.

Based on these findings, and using the summarized information provided in this work as a starting point, the authors will deepen the identified DevOps areas and capabilities to be an a priori and open model, which will be the target of a subsequent research project - which will aim to test and refine this systematized view (in the form of a maturity model), having not only implications for existing scientific knowledge but also being useful for organizational practices of DevOps.

References

Abdelkebir, S., Maleh, Y., & Belaissaoui, M. (2017). An Agile Framework for ITS Management In Organizations. *Proceedings of the 2nd International Conference on Computing and Wireless Communication Systems - ICCWCS'17*, 1–8. https://doi.org/10.1145/3167486.3167556

Ali, E., Caputo, A., & Lawless, S. (2017). Entity attribute ranking using learning to rank. *CEUR Workshop Proceedings*, *1883*, 19–24. https://doi.org/10.1145/1235

Bai, X., Li, M., Pei, D., Li, S., & Ye, D. (2018a). Continuous delivery of personalized assessment and feedback in agile software engineering projects. *Proceedings - International Conference on Software Engineering*, *Part F1373*, 58–67. https://doi.org/10.1145/3183377.3183387

Bai, X., Li, M., Pei, D., Li, S., & Ye, D. (2018b). Continuous delivery of personalized assessment and feedback in agile software engineering projects. *Proceedings - International Conference on Software Engineering*, *Part F1373*, 58–67. https://doi.org/10.1145/3183377.3183387

Bang, S. K., Chung, S., Choh, Y., & Dupuis, M. (2013). A grounded theory analysis of modern web applications. *Proceedings of the 2nd Annual Conference on Research in Information Technology - RIIT '13*, 61. https://doi.org/10.1145/2512209.2512229

Bass, L. (2017). The Software Architect and DevOps. *IEEE Software*, *35*(1), 8–10. https://doi.org/10.1109/MS.2017.4541051

Bezemer, C., Eismann, S., Ferme, V., & Grohmann, J. (2018). How is Performance Addressed in DevOps? A Survey on Industrial Practices. (arXiv:1808.06915v1 [cs.SE]), (August). https://doi.org/10.1145/nnnnnnn.nnnnnnn

Bhattacharjee, A., Barve, Y., Gokhale, A., & Kuroda, T. (2018). (WIP) CloudCAMP: Automating the Deployment and Management of Cloud Services. In *2018 IEEE International Conference on Services Computing (SCC)* (pp. 237–240). IEEE. https://doi.org/10.1109/SCC.2018.00038

Bucena, I., & Kirikova, M. (2017). Simplifying the devops adoption process. In *CEUR Workshop Proceedings* (Vol. 1898).

Chen, H. M., Kazman, R., Haziyev, S., Kropov, V., & Chtchourov, D. (2015). Architectural Support for DevOps in a Neo-Metropolis BDaaS Platform. *Proceedings of the IEEE Symposium on Reliable Distributed Systems*, *2016–Janua*, 25–30. https://doi.org/10.1109/SRDSW.2015.14

Cleveland, S. B., Dooley, R., Perry, D., Stubbs, J., Fonner, J. M., & Jacobs, G. A. (2018). Building Science Gateway Infrastructure in the Middle of the Pacific and Beyond. *Proceedings of the Practice and Experience on Advanced Research Computing - PEARC '18*, (Ci), 1–8. https://doi.org/10.1145/3219104.3219151

Colomo-Palacios, R., Fernandes, E., Soto-Acosta, P., & Larrucea, X. (2018). A case analysis of enabling continuous software deployment through knowledge management. *International Journal of Information Management*, *40*(October), 186–189. https://doi.org/10.1016/j.ijinfomgt.2017.11.005

Croker, M., & Hering, M. (2016). *DevOps: Delivering at the speed of today's business DevOps: A matter of survival in the digital age*.

De Bayser, M., Azevedo, L. G., & Cerqueira, R. (2015). ResearchOps: The case for DevOps in scientific applications. *Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management, IM 2015*, *59*(3), 1398–1404. https://doi.org/10.1109/INM.2015.7140503

de Feijter, R., Rob, van V., Jagroep, E., Overbeek, S., & Brinkkemper, S. (2017). Towards the adoption of DevOps in software product organizations: A Maturity Model Approach, (May), 1–173.

de França, B. B. N., Jeronimo, H., & Travassos, G. H. (2016). Characterizing DevOps by Hearing Multiple Voices. *Proceedings of the 30th Brazilian Symposium on Software Engineering - SBES '16*, 53–62. https://doi.org/10.1145/2973839.2973845

Debois, P. (2011). DevOps: A software Revolution in the Making. *Cutter IT Journel*, *24*(8), 34. https://doi.org/10.1016/B978-0-08-025947-5.50004-2

Debroy, V., Miller, S., & Brimble, L. (2018). Building lean continuous integration and delivery pipelines by applying DevOps principles: a case study at Varidesk. *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering - ESEC/FSE 2018*, 851–856. https://doi.org/10.1145/3236024.3275528

DeGrandis, D. (2011). Devops: So you say you want a revolution? *Cutter IT Journal*, *24*(8), 34–39.

Diel, E., Marczak, S., & Cruzes, D. S. (2016). Communication Challenges and Strategies in Distributed DevOps. In *2016 IEEE 11th International Conference on Global Software Engineering (ICGSE)* (pp. 24–28). IEEE. https://doi.org/10.1109/ICGSE.2016.28

Düllmann, T. F., Paule, C., & Van Hoorn, A. (2018). Exploiting devops practices for dependable and secure continuous delivery pipelines. *Proceedings - International Conference on Software Engineering*, *Part F1378*, 27–30. https://doi.org/10.1145/3194760.3194763

Ebert, C., Gallardo, G., Hernantes, J., & Serrano, N. (2016). DevOps. *IEEE Software*, *33*(3), 94–100. https://doi.org/10.1109/MS.2016.68

Erich, F., Amrit, C., & Daneva, M. (2014). Cooperation between information system development and operations. In *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM '14* (pp. 1–1). https://doi.org/10.1145/2652524.2652598

Farshchi, M., Schneider, J.-G., Weber, I., & Grundy, J. (2015). Experience report: Anomaly detection of cloud application operations using log and cloud metric correlation analysis. *2015 IEEE 26th International Symposium on Software Reliability Engineering (ISSRE)*, 24–34. https://doi.org/10.1109/ISSRE.2015.7381796

Feitelson, D. G., Frachtenberg, E., & Beck, K. L. (2013). Development and deployment at facebook. *IEEE Internet Computing*, *17*(4), 8–17. https://doi.org/10.1109/MIC.2013.25

Fitzgerald, B., & Stol, K.-J. (2014). Continuous software engineering and beyond: trends and challenges. *Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering - RCoSE 2014*, 1–9. https://doi.org/10.1145/2593812.2593813

Fördős, V., & Cesarini, F. (2016). CRDTs for the configuration of distributed Erlang systems. *Proceedings of the 15th International Workshop on Erlang - Erlang 2016*, 42–53. https://doi.org/10.1145/2975969.2975974

Gazivoda, M. (2018). Application of DevOps Approach in Developing Business Intelligence System in Bank, (June), 11–14.

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

Hanappi, O., Hummer, W., & Dustdar, S. (2016). Asserting reliable convergence for configuration management scripts. *ACM SIGPLAN Notices*, *51*(10), 328–343. https://doi.org/10.1145/3022671.2984000

Humble, J., & Molesky, J. (2011). Why {Enterprises} {Must} {Adopt} {Devops} to {Enable} {Continuous} {Delivery}. *Cutter IT Journal*, *24*(8), 6–12.

Hussain, W., Clear, T., & MacDonell, S. (2017). Emerging trends for global DevOps: A New Zealand perspective. In *Proceedings - 2017 IEEE 12th International Conference on Global Software Engineering, ICGSE 2017* (pp. 21–30). IEEE. https://doi.org/10.1109/ICGSE.2017.16

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

Hüttermann, M. (2012). Introducing DevOps. In *DevOps for Developers (Part I)* (pp. 15–31). https://doi.org/10.1007/978-1-4302-4570-4_2

Jabbari, R., bin Ali, N., Petersen, K., & Tanveer, B. (2016a). What is DevOps? In *Proceedings of the Scientific Workshop Proceedings of XP2016 on - XP '16 Workshops* (pp. 1–11). https://doi.org/10.1145/2962695.2962707

Jabbari, R., bin Ali, N., Petersen, K., & Tanveer, B. (2016b). What is DevOps? *Proceedings of the Scientific Workshop Proceedings of XP2016 on - XP '16 Workshops*, (March 2018), 1–11. https://doi.org/10.1145/2962695.2962707

Jimenez, I., Sevilla, M., Watkins, N., Maltzahn, C., Lofstead, J., Mohror, K., … Arpaci-Dusseau, R. (2017). The popper convention: Making reproducible systems evaluation practical. *Proceedings - 2017 IEEE 31st International Parallel and Distributed Processing Symposium Workshops, IPDPSW 2017*, 1561–1570. https://doi.org/10.1109/IPDPSW.2017.157

John, W., Meirosu, C., Pechenot, B., Sköldström, P., Kreuger, P., & Steinert, R. (2015). Scalable Software Defined Monitoring for Service Provider DevOps. *Proceedings*

- *European Workshop on Software Defined Networks, EWSDN*, 61–66. https://doi.org/10.1109/EWSDN.2015.62

Jones, S., Noppen, J., & Lettice, F. (2016). Management challenges for DevOps adoption within UK SMEs. In *Proceedings of the 2nd International Workshop on Quality-Aware DevOps - QUDOS 2016* (pp. 7–11). https://doi.org/10.1145/2945408.2945410

Kamuto, M. B., & Langerman, J. J. (2018). Factors inhibiting the adoption of DevOps in large organisations: South African context. In *RTEICT 2017 - 2nd IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, Proceedings* (Vol. 2018–Janua, pp. 48–51). https://doi.org/10.1109/RTEICT.2017.8256556

Karapantelakis, A., Liang, H., Wang, K., Vandikas, K., Inam, R., Fersman, E., … Giannokostas, V. (2016). DevOps for IoT applications using cellular networks and cloud. In *Proceedings - 2016 IEEE 4th International Conference on Future Internet of Things and Cloud, FiCloud 2016* (pp. 340–347). https://doi.org/10.1109/FiCloud.2016.55

Khan, K., Kunz, R., Kleijnen, J., & Antes, G. (2004). Systematic Reviews to Support Evidence-Based Medicine: How to Review and Apply Findings of Healthcare Research. *Postgraduate Medical Journal*. https://doi.org/10.1016/S1036-7314(00)70624-2

Kim, G. (2015). Top 11 Things You Need to Know about DevOps. *IT Revolution Press*, 1–20. Retrieved from http://info.leankit.com/top-eleven-things-to-know-about-devops

Kitchenham, B. (2004). *Procedures for performing systematic reviews. Joint Technical Report, Computer Science Department, Keele University*. https://doi.org/10.1.1.122.3308

Kuusinen, K., Balakumar, V., Jepsen, S. C., & Larsen, S. H. (2018). A Large Agile Organization on its Journey towards DevOps, 60–63. https://doi.org/10.1109/SEAA.2018.00019

Lassenius, C., Dingsøyr, T., & Paasivaara, M. (2015). Agile processes, in software engineering, and extreme programming: 16th international conference, XP 2015 Helsinki, Finland, may 25-29, 2015 proceedings. *Lecture Notes in Business Information Processing*, *212*, 166–177. https://doi.org/10.1007/978-3-319-18612-2

Laukkarinen, T., Kuusinen, K., & Mikkonen, T. (2017). DevOps in regulated software development: Case medical devices. In *Proceedings - 2017 IEEE/ACM 39th International Conference on Software Engineering: New Ideas and Emerging Results Track, ICSE-NIER 2017* (pp. 15–18). https://doi.org/10.1109/ICSE-NIER.2017.20

Laukkarinen, T., Kuusinen, K., & Mikkonen, T. (2018). Regulated software meets DevOps. *Information and Software Technology*, *97*, 176–178. https://doi.org/10.1016/j.infsof.2018.01.011

Lewerentz, M., Bluhm, T., Daher, R., Dumke, S., Grahl, M., Grün, M., … Werner, A. (2018). Implementing DevOps practices at the control and data acquisition system of an experimental fusion device. *Fusion Engineering and Design*, (November), 0–

1. https://doi.org/10.1016/j.fusengdes.2018.11.022

Li, Z., Zhang, Y., & Liu, Y. (2017). Towards a full-stack devops environment (platform-as-a-service) for cloud-hosted applications. *Tsinghua Science and Technology*, *22*(1), 1–9. https://doi.org/10.1109/TST.2017.7830891

Luz, W. P., Pinto, G., & Bonifácio, R. (2018). Building a collaborative culture. In *Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM '18* (Vol. 18, pp. 1–10). New York, New York, USA: ACM Press. https://doi.org/10.1145/3239235.3240299

Lwakatare, L. E., Kuvaja, P., & Oivo, M. (2015a). Dimensions of devOps. In *Lecture Notes in Business Information Processing* (Vol. 212, pp. 212–217). https://doi.org/10.1007/978-3-319-18612-2_19

Lwakatare, L. E., Kuvaja, P., & Oivo, M. (2015b). Dimensions of devOps. *Lecture Notes in Business Information Processing*, *212*, 212–217. https://doi.org/10.1007/978-3-319-18612-2_19

Mackey, T. (2018). Building open source security into agile application builds. *Network Security*, *2018*(4), 5–8. https://doi.org/10.1016/S1353-4858(18)30032-1

Mansfield-Devine, S. (2018). DevOps: finding room for security. *Network Security, 2018*(7), 15–20. https://doi.org/10.1016/S1353-4858(18)30070-9

Marijan, D., Liaaen, M., & Sen, S. (2018). DevOps Improvements for Reduced Cycle Times with Integrated Test Optimizations for Continuous Integration. *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, 22–27. https://doi.org/10.1109/COMPSAC.2018.00012

McCarthy, M. A., Herger, L. M., Khan, S. M., & Belgodere, B. M. (2015). Composable DevOps: Automated Ontology Based DevOps Maturity Analysis. *Proceedings - 2015 IEEE International Conference on Services Computing, SCC 2015*, 600–607. https://doi.org/10.1109/SCC.2015.87

Mikkonen, T., Lassenius, C., Männistö, T., Oivo, M., & Järvinen, J. (2018). Continuous and collaborative technology transfer: Software engineering research with real-time industry impact. *Information and Software Technology, 95*(October), 34–45. https://doi.org/10.1016/j.infsof.2017.10.013

Mohamed, S. I. (2015). DevOps Shifting Software Engineering Strategy Value Based Perspective. *IOSR Journal of Computer Engineering Ver. IV, 17*(2), 2278–2661. https://doi.org/10.9790/0661-17245157

Mohan, V., & Ben Othmane, L. (2016). SecDevOps: Is it a marketing buzzword? Mapping research on security in DevOps. *Proceedings - 2016 11th International Conference on Availability, Reliability and Security, ARES 2016*, 542–547. https://doi.org/10.1109/ARES.2016.92

Molto, G., Caballer, M., Perez, A., Alfonso, C. De, & Blanquer, I. (2017). Coherent Application Delivery on Hybrid Distributed Computing Infrastructures of Virtual Machines and Docker Containers. *Proceedings - 2017 25th Euromicro International Conference on Parallel, Distributed and Network-Based Processing, PDP 2017*, 486–490. https://doi.org/10.1109/PDP.2017.29

Moore, J., Kortuem, G., Smith, A., Chowdhury, N., Cavero, J., & Gooch, D. (2016).

DevOps for the Urban IoT. *Proceedings of the Second International Conference on IoT in Urban Space - Urb-IoT ’16*, 78–81. https://doi.org/10.1145/2962735.2962747

Nielsen, P. A., Winkler, T. J., & Nørbjerg, J. (2017). Closing the IT development-operations gap: The devops knowledge sharing framework. *CEUR Workshop Proceedings*, *1898*.

Palihawadana, S., Wijeweera, C. H., Sanjitha, M. G. T. N., Liyanage, V. K., Perera, I., & Meedeniya, D. A. (2017). Tool support for traceability management of software artefacts with DevOps practices. *3rd International Moratuwa Engineering Research Conference, MERCon 2017*, 129–134. https://doi.org/10.1109/MERCon.2017.7980469

Pang, C., & Hindle, A. (2017). Continuous maintenance. *Proceedings - 2016 IEEE International Conference on Software Maintenance and Evolution, ICSME 2016*, 458–462. https://doi.org/10.1109/ICSME.2016.45

Perera, P., Bandara, M., & Perera, I. (2017). Evaluating the impact of DevOps practice in Sri Lankan software development organizations. *16th International Conference on Advances in ICT for Emerging Regions, ICTer 2016 - Conference Proceedings*, (December), 281–287. https://doi.org/10.1109/ICTER.2016.7829932

Perera, P., Silva, R., & Perera, I. (2017). Improve software quality through practicing DevOps. In *2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer)* (Vol. 2018–Janua, pp. 1–6). IEEE. https://doi.org/10.1109/ICTER.2017.8257807

Punjabi, R., & Bajaj, R. (2017). User stories to user reality: A devops approach for the cloud. In *2016 IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, RTEICT 2016 - Proceedings* (pp. 658–662). https://doi.org/10.1109/RTEICT.2016.7807905

Qumer 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. https://doi.org/10.1108/VJIKMS-02-2017-0007

Rahman, A., Mahdavi-Hezaveh, R., & Williams, L. (2018). A systematic mapping study of infrastructure as code research. *Information and Software Technology*, (November). https://doi.org/10.1016/j.infsof.2018.12.004

Rajkumar, M., Pole, A. K., Adige, V. S., & Mahanta, P. (2016). DevOps culture and its impact on cloud delivery and software development. In *Proceedings - 2016 International Conference on Advances in Computing, Communication and Automation, ICACCA 2016*. https://doi.org/10.1109/ICACCA.2016.7578902

Rana, R., & Staron, M. (2016). First International Workshop on Emerging Trends in DevOps and Infrastructure. *Proceedings of the Scientific Workshop Proceedings of XP2016 on - XP ’16 Workshops*, 1–3. https://doi.org/10.1145/2962695.2962706

Riungu-Kalliosaari, L., Mäkinen, S., Lwakatare, L. E., Tiihonen, J., & Männistö, T. (2016). DevOps adoption benefits and challenges in practice: A case study. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* (Vol. 10027 LNCS, pp. 590–597). https://doi.org/10.1007/978-3-319-49094-6_44

Roche, J. (2013). Adopting DevOps practices in quality assurance. *Communications of the ACM*, *56*(11), 38–43. https://doi.org/10.1145/2524713.2524721

Rodríguez, P., Mäntylä, M., Oivo, M., Lwakatare, L. E., Seppänen, P., & Kuvaja, P. (2018). Advances in Using Agile and Lean Processes for Software Development. *Advances in Computers*. https://doi.org/10.1016/bs.adcom.2018.03.014

Rong, G., Zhang, H., & Shao, D. (2016). CMMI guided process improvement for DevOps projects. *Proceedings of the International Workshop on Software and Systems Process - ICSSP '16*, 76–85. https://doi.org/10.1145/2904354.2904372

Rubasinghe, I. D., Meedeniya, D. A., & Perera, I. (2017). Towards Traceability Management in Continuous Integration with SAT-analyzer. *Proceedings of the 3rd International Conference on Communication and Information Processing*, 77–81. https://doi.org/10.1145/3162957.3162985

Rubasinghe, I., Meedeniya, D., Perera, I., & Practice, A. T. (2018). Automated Inter-artefact Traceability Establishment for DevOps Practice, 211–216. https://doi.org/10.1109/ICIS.2018.8466414

Rufino, J., Alam, M., & Ferreira, J. (2017). Monitoring V2X applications using DevOps and docker. *2017 International Smart Cities Conference, ISC2 2017*. https://doi.org/10.1109/ISC2.2017.8090868

Samarawickrama, S. S., & Perera, I. (2018). Continuous scrum: A framework to enhance scrum with DevOps. In *17th International Conference on Advances in ICT for Emerging Regions, ICTer 2017 - Proceedings* (Vol. 2018–Janua, pp. 19–25). https://doi.org/10.1109/ICTER.2017.8257808

Science, C., & Sciences, N. (2015). Full-scale Software Engineering FsSE 2015. *Full-Scale Software Engineering FsSE*, 31–36.

Shahin, M., Babar, M. A., & Zhu, L. (2016). The Intersection of Continuous Deployment and Architecting Process. *Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement - ESEM '16*, 1–10. https://doi.org/10.1145/2961111.2962587

Sharma, S. (2017a). DevOps Plays for Driving Innovation. In *The DevOps Adoption Playbook* (pp. 189–260). https://doi.org/10.1002/9781119310778.ch5

Sharma, S. (2017b). *The DevOps Adoption Playbook*. https://doi.org/10.1002/9781119310778

Sharma, S., & Coyne, B. (2015). *DevOps For Dummies*. *John Wiley & Sons, Inc.* (Vol. 1). https://doi.org/10.1017/CBO9781107415324.004

Shivakumar, S. K. (2017). *DevOps for Digital Enterprises Brief Introduction to DevOps Scope*.

Silva, M. A., Faustino, J., Pereira, R., & Silva, M. M. (2018). Productivity gains of DevOps adoption in an IT team: a case study. *27th International Conference on Information Systems Development*. Retrieved from https://repositorio.iscte-iul.pt/handle/10071/16388

Snyder, B., & Curtis, B. (2017). Using Analytics to Guide Improvement during an Agile-DevOps Transformation. *IEEE Software*, *35*(1), 78–83.

https://doi.org/10.1109/MS.2017.4541032

Soni, M. (2016). End to End Automation on Cloud with Build Pipeline: The Case for DevOps in Insurance Industry, Continuous Integration, Continuous Testing, and Continuous Delivery. In *Proceedings - 2015 IEEE International Conference on Cloud Computing in Emerging Markets, CCEM 2015* (pp. 85–89). https://doi.org/10.1109/CCEM.2015.29

St, D., Ab, E., & Bosch, J. (2017). Continuous Practices and DevOps : Beyond the Buzz , What Does It All Mean ? *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA) Continuous*, 440–448. https://doi.org/10.1109/SEAA.2017.78

Steffens, A., Lichter, H., & Döring, J. S. (2018a). Designing a Next-Generation Continuous Software Delivery System : Concepts and Architecture, 1–7.

Steffens, A., Lichter, H., & Döring, J. S. (2018b). Designing a next-generation continuous software delivery system. *Proceedings of the 4th International Workshop on Rapid Continuous Software Engineering - RCoSE '18*, 1–7. https://doi.org/10.1145/3194760.3194768

Stoneham, J., Thrasher, P., Potts, T., Mickman, H., DeArdo, C., & Limoncelli, T. A. (2016). DevOps Case Studies, 46.

Stoneham, J., Thrasher, P., Potts, T., Mickman, H., DeArdo, C., & Limonchelli, T. A. (2017). *DEVOPS CASE STUDIES: The Journey to Positive Business Outcomes* (1st ed.). Oregon, Portland: IT Revolution Press.

Sturm, R., Pollard, C., & Craig, J. (2017). DevOps and Continuous Delivery. In *Application Performance Management (APM) in the Digital Enterprise* (pp. 121–135). Elsevier. https://doi.org/10.1016/B978-0-12-804018-8.00010-3

Swanson, E. B., & Beath, C. M. (1990). Departmentalization in software development and maintenance. *Communications of the ACM*. https://doi.org/10.1145/78973.78976

Tessem, B., & Iden, J. (2008). Cooperation between developers and operations in software engineering projects. In *Proceedings of the 2008 international workshop on Cooperative and human aspects of software engineering - CHASE '08*. https://doi.org/10.1145/1370114.1370141

Tingley, G. A., & Anderson, R. M. (1986). Environmental sex determination and density-dependent population regulation in the entomogenous nematode Romanomermis culicivorax. *Parasitology, 92*(02), 431. https://doi.org/10.1017/S0031182000064192

Tuma, K., Calikli, G., & Scandariato, R. (2018). Threat analysis of software systems: A systematic literature review. *Journal of Systems and Software*, *144*, 275–294. https://doi.org/10.1016/j.jss.2018.06.073

Ur Rahman, A. A., & Williams, L. (2016a). Security practices in DevOps. *Proceedings of the Symposium and Bootcamp on the Science of Security - HotSos '16*, 109–111. https://doi.org/10.1145/2898375.2898383

Ur Rahman, A. A., & Williams, L. (2016b). Software security in DevOps. *Proceedings of the International Workshop on Continuous Software Evolution and Delivery -*

CSED '16, 70–76. https://doi.org/10.1145/2896941.2896946

Vassallo, C., Zampetti, F., Romano, D., Beller, M., Panichella, A., Di Penta, M., & Zaidman, A. (2017). Continuous delivery practices in a large financial organization. *Proceedings - 2016 IEEE International Conference on Software Maintenance and Evolution, ICSME 2016*, 519–528. https://doi.org/10.1109/ICSME.2016.72

Virmani, M. (2015). Understanding DevOps & bridging the gap from continuous integration to continuous delivery. In *5th International Conference on Innovative Computing Technology, INTECH 2015* (pp. 78–82). https://doi.org/10.1109/INTECH.2015.7173368

Webster, J., & Watson, R. T. (2002). Analyzing the Past to Prepare for the Future: Writing a Literature Review. *MIS Quarterly*, *26*(2), xiii–xxiii. https://doi.org/10.1.1.104.6570

Wiesche, M. (2018). ARE YOU READY FOR DEVOPS ? REQUIRED SKILL SET FOR DEVOPS TEAMS DevOps teams. *Twenty-Sixth European Conference on Information Systems (ECIS2018).*

Wongkampoo, S., & Kiattisin, S. (2018). Atom-Task Precondition Technique to Optimize Large Scale GUI Testing Time based on Parallel Scheduling Algorithm. *ICSEC 2017 - 21st International Computer Science and Engineering Conference 2017, Proceeding*, *6*, 229–232. https://doi.org/10.1109/ICSEC.2017.8443913

Xia, C., Zhang, Y., Wang, L., Coleman, S., & Liu, Y. (2018). Microservice-based cloud robotics system for intelligent space. *Robotics and Autonomous Systems*, *110*, 139–150. https://doi.org/10.1016/j.robot.2018.10.001

Zhu, H., & Bayley, I. (2018). If Docker is the Answer, What is the Question? *Proceedings - 12th IEEE International Symposium on Service-Oriented System Engineering, SOSE 2018 and 9th International Workshop on Joint Cloud Computing, JCC 2018*, 152–163. https://doi.org/10.1109/SOSE.2018.00027

Zhu, L., Bass, L., & Champlin-Scharff, G. (2016). DevOps and Its Practices. *IEEE Software*, *33*(3), 32–34. https://doi.org/10.1109/MS.2016.81