Universidad de Valladolid

ESCUELA DE INGENIERÍAS INDUSTRIALES

UNIVERSIDAD DE VALLADOLID

ESCUELA DE INGENIERIAS INDUSTRIALES

Grado en Ingeniería de Organización Industrial

DevOps: IT Development in the Era of Digitalization

Autor: Pérez Hoyos, Laura

Rey Martínez, Francisco Javier

Technische Universität Dresden

Valladolid, julio 2018.

TFG REALIZADO EN PROGRAMA DE INTERCAMBIO

TÍTULO: DevOps: IT Development in the Era of Digitalization

ALUMNO: Laura Pérez Hoyos

FECHA: 26.06.2018

CENTRO: Technische Universität Dresden

TUTOR: Prof. Dr. Susanne Strahringer

RESUMEN

El propósito de la tesis es transmitir una visión clara de cómo funciona DevOps en una organización. Los objetivos del estudio son proporcionar una definición detallada del concepto DevOps, identificar los beneficios de la implementación de DevOps en empresas, los desafíos que enfrentan las organizaciones durante el proceso de adopción así como identificar las prácticas de DevOps correspondientes a cada pilar fundamental, designado con el acrónimo CAMS. Se realiza una revisión sistemática de la literatura (SLR) como método para identificar las prácticas de adopción de DevOps y extraer una conclusión. El listado obtenido con las correspondientes prácticas podrá servir para obtener una mejor comprensión del funcionamiento y adopción de DevOps, para investigadores y profesionales del software, además de base para futuras investigaciones en el campo

PALABRAS CLAVE: *DevOps, CAMS, Agile, software development, software operations*

ABSTRACT

The purpose of the thesis is to provide a clear understanding of how DevOps works in an organization to researchers and software practitioners. The objectives of the study are to provide a detailed definition of DevOps concept, to identify the benefits of implementing DevOps in organizations where agile development is in practice, the challenges faced by organizations during DevOps adoption and to identify the DevOps practices corresponding to each main pillar termed as CAMS. A Systematic Literature Review is conducted as a method to identify the practices of DevOps adoption and leading to the conclusion. The final list of practices can aid researchers and software practitioners in obtaining a better understanding regarding the functioning and adoption of DevOps. Also, it has been observed that there is a need for more empirical research in this domain and some future research ideas are proposed.

KEYWORDS: *DevOps, CAMS, Agile, software development, software operations*

TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Wirtschaftswissenschaften

Professur für Wirtschaftsinformatik,
insb. Informationssysteme in Industrie und Handel

DevOps: IT Development in the Era of Digitalization

Bachelor Diplomarbeit

zur Erlangung des akademischen Grades

„Bachelor of Science in Industrial Engineering“

Name: Pérez Hoyos, Laura
Adresse: Estocolmo 67, 47008 Valladolid, Spain
Matrikelnummer: 4754189
Übermittelt an: Prof. Dr. Susanne Strahringer
Übermittlungsdatum: 16.07.2018

ABSTRACT

Within the last decade, the software industry has been marked by a growing trend of software companies' ability to deploy new software features rapid and frequently in short release cycle times. The companies' software release cycles have been shortened to hours and minutes instead of months. To enable the transformation towards short release cycle times, companies have adopted several different strategies, including the DevOps approach. DevOps in the software industry emerged to represent a professional movement emphasizing the collaboration between software development and operations. In practice, DevOps affects the company culture, processes, products, associated technologies and organizational structures used in software development and operations processes. The multifaceted nature of DevOps makes the concept ambiguous and difficult for software companies to undertake as there are many distinct paths to its adoption.

The purpose of the thesis is to provide a clear understanding of how DevOps works in an organization to researchers and software practitioners. The objectives of the study are to provide a detailed definition of DevOps concept, to identify the benefits of implementing DevOps in organizations where agile development is in practice, the challenges faced by organizations during DevOps adoption and to identify the DevOps practices corresponding to each main pillar termed as CAMS. A Systematic Literature Review is conducted as a method to identify the practices of DevOps adoption. 34 primary studies relevant to the research are identified for conducting the SLR and leading to the conclusion. The final list of practices can aid researchers and software practitioners in obtaining a better understanding regarding the functioning and adoption of DevOps. Also, it has been observed that there is a need for more empirical research in this domain and some future research ideas are proposed.

Keywords: DevOps, CAMS, Agile, software development, software operations, practices

ACKNOWLEDGEMENTS

I would like to extend thanks to the many people, in many countries, who so generously contributed to the work presented in this thesis.

Special mention goes to my thesis supervisor, Susanne Strahringer. My studying abroad programme has been an amazing experience and I thank Susanne wholeheartedly, not only for her academic support but also for giving me the opportunity of studying in the chair of Information Systems at TU Dresden.

Similar, profound gratitude goes to Javier Rey who has been a truly helpful mentor at my home university. Without forgetting all my professors, whereof I am very grateful, for this four years studying Industrial Engineering both at the University of Valladolid and at the Polytechnic University of Valencia.

I am also hugely appreciative to Erasmus+ Program, specifically for all people who work for making it possible, because this has been the best experience of my life so far. I also seize this opportunity to encourage those students hesitating about studying abroad: just go!

Finally, but by no means least, thanks go to mum, dad and Anita for almost unbelievable support. They are the most important people in my world and I dedicate this thesis to them.

INDEX OF CONTENTS

INDEX OF TABLES

INDEX OF FIGURES

1 Introduction

1.1 Research motivation

The need give a rapid response in the Information System field has driven the IT community to enhance, control and improve existing software development methodologies such as Agile or Scrum to suit the rapid changes in software development industry and meet the client requirements.

Some of this requirements like shorten time to market, customer satisfaction, stabilization and reliability, better product quality and are nowadays essential to any modern software development organization that wants to gain a stable position in the market.

Therefore, this research is motivated by the growing trend of the software industry's transitioning towards continuous delivery and deployment (Lwakatare et al., 2016). Continuous deployment is an industrial paradigm characterised by the frequent, rapid and automated release of software changes, including new features, to customers as soon as the changes are commited by software developer so as to gain quick feedback.

Whenever new changes are submitted, it is necessary to update the connected artifacts, repositories to avoid code defects and effort consumption due to wrong or out-of-date information. A movement addressed these challenges to improve the processes to keep up with advancement in the software industry, this movement is termed the DevOps.

1.2 Aims and objectives

The purpose of this thesis is to improve the DevOps practices to help organizations enhancing DevOps core values while implementing DevOps. The description of DevOps practices benefit other researchers and organizations alike by addressing and identifying the contemporary state of DevOps in the industry and academia, which in turn enables to reduce time consumption and bypass potential obstacles. The primary outcome of this thesis is to address the practices to adopt DevOps and enhance its core values via gathering information from different fonts.

- The first research challenge relates to ambiguity in the concept of DevOps and the subsequent need to understand its underlying assumptions that would be beneficial to assess its practices (Lassenius, Dingsøyr, & Paasivaara, 2015). This research challenge is also important to stir a discussion on whether the concept and the underpinning assumptions of DevOps are novel or antecedents to the established knowledge base of software development methods. This research challenge was also true for the agile concept in software development in the early years of its introduction (Ramamurthy, 2016).
- The second research challenge relates to the identification of benefits and challenges that software-intensive organizations are likely to experience when adopting and applying DevOps practices. Existing research on collaboration is software development projects report on the positive impact of careful coordination between several individuals across the different stages of software development process on software project success.
- The third research challenge concerns how DevOps is implemented in practice. As there is no generally agreed framework for DevOps adoption and implementation, the manifestation of DevOps concept varies from one organization to another. From a conceptual understanding, software practitioners of the DevOps movement specify moving away from software development approaches that encourage the separation of software development and operations teams into silos.

1.3 Research scope

The aim of the thesis is to provide a detailed description of the adoption and implementation of DevOps in software development practice, by understanding the definition, benefits and challenges of the concept. And finally, an analysis of some articles on the topics, in relation to the four pillars of the DevOps concept (culture, automation, measurement and shared use) with its corresponding related practices.

1.4 Outline of the thesis

This thesis contains five chapters, whose contents are logically structured as follows.

In the first chapter, the research motivation, challenges and scope are described. In the second and third, there is a theoretical part about the background in DevOps concept and the method used for the Systematic Literature Review. In the fourth, the practical part of the analysis of the data extracted for the SLR. And to sum up, in the last chapter, a conclusion to sum up the contributions and some ideas for future research.

2 Background

2.1 DevOps concept

DevOps is the aggregate of practices, cultural philosophies, and equipment that increase the potential to deliver applications and service at high delivery (Zhu, Bass, & Champlin-Scharff, 2016). It is far referred to a software development methodology so as to emphasize communication, collaboration, and integration between the historically separate departments (Callanan & Spillane, 2016): software program developers and IT experts. Thus, DevOps is a response to the interdependence between software development and operational development.

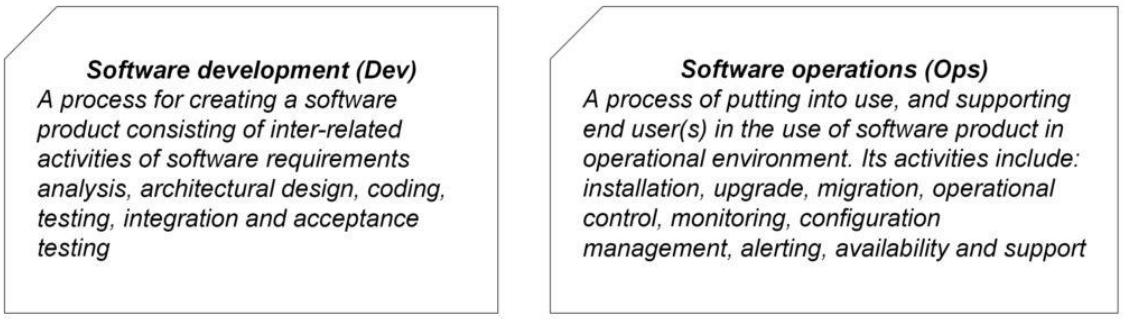

Figure 1 - Description of software development and operations.

DevOps emerged first as a movement inside the software industry. The term 'DevOps' become first coined in 2009 when beginning a practitioners' event called DevOps Days (Bang, Chung, Choh, & Dupuis, 2013). The event was arranged by Patrick Debois (Shafer, Debois, Gat, & Debois, 2011), a leading pioneer of the DevOps movement, who had recognized a growing interest amongst other practitioners on the need to improve the process of delivering software quickly to production. The DevOps movement emphasizes collaboration between Software Development and Operations with the purpose of reaching a fast launch of software program functions. Instead of seeing those as two distinct groups who are responsible for their particular tasks but do not truly work together, the DevOps methodology recognizes the interdependence of the two groups. By integrating these functions into one team or department, DevOps enables an organization deploy software more often while maintaining service stability and gaining the velocity vital for extra innovation.

Adopting DevOps will not be a straightforward challenge considering that it may require that an enterprise introduces method, employees and technological changes and improvements. As in any software program system improvement initiative, the path to a successful DevOps adoption is unique to each organisation. Nonetheless, it is feasible to analyze from demanding situations experienced throughout DevOps adoptions in order to plan future DevOps adoption initiatives. And, finally, everybody is capable of delivering the satisfactory outcomes and overall experience possible to the client.

2.2 Fundamental principles: CAMS

The concept DevOps is based on four main pillars (Fitzgerald & Stol, 2014). Like every movement, DevOps had defined the core targeted values that impact the contemporary requirement of software development lifecycle, the acronym that describes those values is CAMS (Rajkumar, Pole, Adige, & Mahanta, 2016) stands for Culture, Automation, Measurement, and Sharing.

• Culture — DevOps requires a cultural change of accepting joint responsibility for delivery of high quality software to the end-user. Which means that code no longer can be "thrown over the wall" to operations. DevOps aims to dispose of the silos among groups and enhance interdepartmental communication. Companies and individuals are consuming plenty of time because of the conventional way of interdepartmental communication (handoff documentation, queues), and in some companies, it might be difficult to ask other people for help outside the official channels. DevOps enhance a healthy communication to permit all teams to communicate and get rid of the communication barriers. Encouraging a communicative culture is a key to having an effective environment.

• Automation — DevOps is based on complete automation of the build, deployment, and testing with a view to achieving short lead times, and therefore rapid delivery and feedback from end-users. Possibly the most visible aspect of DevOps. Many people focus on the productivity gains (output per worker per hour) as the primary cause to adopt DevOps. Not only automation used to save time, but also to prevent defects, create consistency, and permit self-service.

• Measurement — Gaining an understanding of the current delivery functionality and setting goals for improving it can only be carried out through measuring. This varies from tracking business metrics (e.g., sales) to test coverage and the time to deploy a new version of the software. DevOps helps continuous delivery and deployment. Continuous delivery requires continuous improvement of the product, thus, DevOps supports measurement and makes it one of its core values because the movement believes it is difficult to improve without the ability to measure.

Decisions based on visible and easy-to-read data are the key to having the proper choices. The data should be available, transparent, accessible, related visually.

• Sharing — Sharing happens at distinct stages, from sharing know-how (e.g. about new functionality in a release), sharing tools and infrastructure, as well as sharing in celebrating successful releases to bring development and operations teams closer together. DevOps found out the energy of sharing and the great impact it brings. Within the business enterprise sharing tools, findings, defects, and experiences enable the people who share similar interests and needs to meet. This kind of move makes the method of finding new opportunities to collaboration a lot simpler, removing duplicated work along with having a powerful sense of commitment. DevOps also, promote sharing resources (code, documentation… and so on.) outside the organization with the related community. It enables the company to discover new features, find defects and energize the people.These principles again reflect the trends of changing perspectives rely on responsibility and measurement which are also found in Enterprise Agile. Furthermore, automation emphasizes a greater reliance on tools and a sound development and delivery infrastructure.

2.3 DevOps toolchain and lifecycle

A toolchain is a set of programming tools which can be used to carry out a complex software development task or to create a software product, that is commonly some other computer program or a set of related programs. In DevOps field, they useful resource in the delivery, development, and management of applications during the software development lifecycle, as coordinated by using an organization that uses DevOps practices.

As DevOps is a set of practices that emphasize the collaboration and communication of both software developers and other information technology (IT) professionals. While automating the process of software program delivery and infrastructure changes, its implementation can consist of the definition of the series of tools used at diverse stages of the lifecycle. Due to the fact that DevOps is a cultural shift and collaboration between development and operations, there may be no one product that can be considered a single DevOps tool. Rather a set of tools, probably from a variety of vendors, are utilized in one or extra stages of the lifecycle.

To sum up, DevOps lifecycle consist of the following stages (Edwards, 2010): Plan, Create, Verify, Package, Release, Configure, and Monitor.

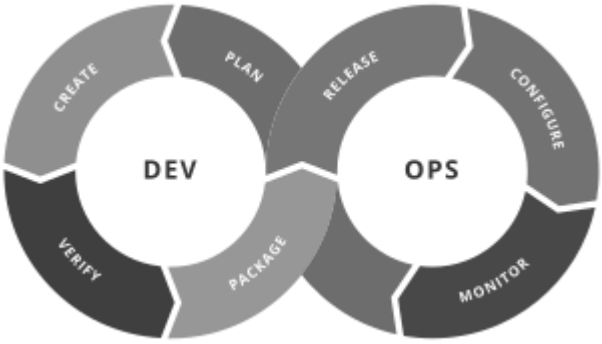

Figure 2 - Illustration showing stages in DevOps lifecycle

2.3.1 Plan

To aid the development and operations groups to regulate the rapid changes in enterprise environments, there is a want to have agile enterprise plans in the area. DevOps allows this through having a product backlog which can be prioritized all of the time, and a non-stop channel for feedback from clients. It is crucial that planning is achieved often to deal with short changes in business environments, and there needs to be tight integration among planning and execution. The continuous planning(Anand Srivatsav & Swetha Bindu, 2016) is a cyclic procedure and performed through planning small batches, executing, obtaining feedback, responding to the feedback and adjusting the plan if needed.

Thus, plan refers to the business value and the application requirements and some of the activities included are:

- Production metrics, objects and feedback
- Requirements
- Business metrics
- Update release metrics
- Release plan, timing and business case
- Security policy and requirement

2.3.2 Create

Create is composed of the building, coding, and configuring of the software development process. Tools and vendor in this category often overlap with other categories. Because DevOps is about breaking down silos, this is reflective on the activities and product solutions. It is also needed a continuous integration between development and operations teams. To achieve this, integration must be done early, changes should be shared with teams without keeping them localized for long, and code should be validated continuously. Continuous integration requires some form of automation, and has interconnected steps like compiling code, running unit and acceptance steps, validating code coverage and building deployment packages. Automation must be done in such way that as soon as a developer makes a change, it is detected by the build system and a build is triggered, sanity tests are carried out, and the build is posted to a repository.

The specific activities are:

- Design of the software and configuration
- Coding including code quality and performance
- Software build and build performance
- Release candidate

2.3.3 Verify

Verify is directly associated with ensuring the quality of the software release; activities designed to ensure code quality is maintained and the highest quality is deployed to production. Automation of all test cases is necessary to achieve continuous testing. Existing manual test cases should be automated, and the test cases should be executed on every generated software build without any manual intervention. Continuous testing aims at eliminating root causes and involves automation of testing cases so that the time between introduction and detection of errors is reduced.

The main activities in this are:

- Acceptance testing
- Regression testing
- Security and vulnerability analysis
- Performance
- Configuration testing

2.3.4 Package

Packaging refers to the activities involved once the release is ready for deployment, often also referred to as staging or preproduction. This often includes tasks and activities such as:

- Approval/preapprovals
- Package configuration
- Triggered releases
- Release staging and holding

2.3.5 Release

Release related activities include schedule, orchestration, provisioning and deploying software into production and targeted environment. The specific Release activities include:

- Release coordination
- Deploying and promoting applications
- Fallbacks and recovery
- Scheduled/timed releases

Continuous deployment involves automatically deploying the software to some environment, but not necessarily to the customers. Continuous deployment is a prerequisite for achieving continuous delivery.

Continuous delivery involves automatically releasing valid software builds to the customers. Continuous deployment is necessary for achieving continuous delivery, but the reverse is not usually the case. Continuous delivery is the ability to deliver software whenever the organization wants to. In continuous delivery, as soon as new features are finished, they are deployed into production.

2.3.6 Configure

Configure activities fall under the operation side of DevOps. Once software is deployed, there may be additional IT infrastructure provisioning and configuration activities required. Specific activities including:

- Infrastructure storage, database and network provisioning and configuring
- Application provision and configuration.

2.3.7 Monitor

Monitoring is an important link in a DevOps toolchain. It allows IT organization to identify specific issues of specific releases and to understand the impact on end-users. Continuous monitoring of infrastructure and user-behavior provides feedback loops for the planning and development processes to improve and optimize the service. Different quality parameters can be observed, and we can react to any surprises in a timely manner. In DevOps, feedback is the data collected from operating the service as an input in planning and development. This data contains information about the performance of infrastructure, and how and when the users interact with the service.

A summary of Monitor related activities are:

- Performance of IT infrastructure
- End-user response and experience
- Production metrics and statistics

Information from monitoring activities often impacts Plan activities required for changes and for new release cycles.

2.4 Benefits and challenges of DevOps

In this chapter, the benefits and challenges of DevOps (Gottesheim, 2015) are provided. The benefits of DevOps are the ones pertaining to a successful implementation of DevOps in software companies.

It is to be noted that most of the identified benefits are the perceived benefits of adopting DevOps in organizations. Many companies have started to adopt DevOps these days in an attempt to leverage the benefits (see Table 1).

Table 1 – Benefits of DevOps

Benefits of DevOps
• The communication, collaboration and trust between development and operations team member's increases.
• Improved quality of software deployment.
• Responsiveness to business needs increases.
• The software reliability increases.
• The code quality will be improved.
• Implementing the customer requirements throughout the software development process becomes easier.
• Reduced time to software release.
• Reduces the average release cycle time from weeks to hours.
• Rapid delivery of products.
• Increased customer satisfaction.
• DevOps quantifies the aspects of the development process, and the focus on metrics subsequently improves the product development.
• DevOps helps to deliver higher quality services in a more efficient way.
• Smooth and faster work flow.

Note. These are the benefits observed at primary study.

The challenges of DevOps (see Table 2) are offered in two categories. First are those that present obstacles/limitations/hindrances to its adoption. In the second category are the challenges stemming from the implementation of DevOps.

Table 2 – Challenges of DevOps

Challenges of DevOps
• A lack of proper management structure in an organization can hinder the adoption process.
• There is no proper training in place for DevOps in several organizations, so the DevOps engineers are usually hired based on their experience as system administrators and software developers.
• Unclear definition and goals of DevOps adoption.
• Organizational structure may impact the DevOps adoption process.
• In some cases, the customer might fancy a different approach to development, such as including different practices and process with strict procedures on deployment. DevOps might not be suitable for product development in these cases.
• Geographical distribution of the development and operations teams can hinder the DevOps adoption.
• People may perceive DevOps as a buzzword, and this negative perception may cause them to try to resist its adoption.
• Because DevOps requires all personnel to have both development and operations skills, people may not be open to the change due to fear of not having in-depth expertise in both areas.
• People may be interested in only their area of expertise, not in what other teams do.
• The monolithic system architecture can be hindering to the various DevOps practices like continuous builds, testing, and deployment.
• Differences between the development, testing and production environments can complicate the collaboration and also continuous delivery and deployment.
• Multiple production environments can cause complexity, and hinder the use of common tools and processes, which can affect continuous delivery.
• Moving to DevOps is more difficult with legacy software.
• Specifying the software architectures to make them work in DevOps scenarios can be challenging for software architects.
• Lack of buy-in from senior management can be challenging.
• Resistance from employees can also hinder the adoption process.
• Hardware dependency and compatibility with different software versions.
• Limited visibility of customer production environments when configuring test environments.
• Lack of automation tools for deployment of new features in the embedded systems domain.
• If the system performance data collected by companies does not contain feature usage data, it hinders DevOps adoption.

Note. Those challenges were identified while adopting DevOps in organizations.

3 Methodology

3.1 Systematic Literature Review

The objective of the study is to provide a deep understanding of DevOps adoption and implementation in software development practice by comprehending the concept definition, practices, benefits and challenges. A systematic literature review was conducted to identify the main four core values of DevOps.

The Kitchenham (Kitchenham & Charters, 2007) guidelines were followed to conduct the SLR. Kitchenham et al. defined systematic literature review as "a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest". SLR is mostly used for summarizing all the existing evidence related to a topic, identifying gaps in current research, and to provide a background for planning the new research activities accordingly.

SLR offers the needed techniques to minimize bias to produce reliable findings that lead to efficient decision making. The systematic literature review follows the steps proposed by Kitchenham and Charters. The steps are:

1. The search questions
2. The search process
3. Study selection
4. Quality assessment
5. Data extraction process
6. Data extraction results
7. Discussion of research questions
8. Study limitation

After developing the research questions, aims, and objectives a review protocol is created to select the related studies. This protocol is important for data extraction. After that, the searches for resources based on the specified searching and protocol criteria for the studies selection are carried out. The process of data extraction takes place after finishing the practical screening and quality appraisal. Finally, combines the collected facts and start the analysis to prepare for the final review.

The steps that have to be followed to conduct the SLR are identification of keywords, formulation of search strings, devising of inclusion/exclusion criteria, devising of quality criteria, conducting the database search, selection of initial set of primary studies, obtaining a final set of primary studies, data extraction and analysis.

Figure 3 - SLR Process

3.2 Search Strategy

Initially, a database search was conducted to obtain the primary studies. Keywords were identified from the research questions, and search strings were generated by combining these keywords using the Boolean operators "AND" and "OR". Synonyms of the keywords were also added to the search strings to ensure that all relevant studies are obtained.
The keywords identified for this study are: DevOps, Benefits, Challenges, Principles, Practices, Software Development, Development, Operations, CAMS, Interdepartamental.

The following sources were used:

- EBSCO
- IEEE
- Springer Link
- Science Direct
- ACM Digital Library

Table 3 – Databases and keywords

EBSCO	Devops AND (architecture OR tools OR benefits OR challenges OR CAMS OR principles)
IEEE	((DevOps) AND (adoption OR impediments OR challenges OR hindrances OR principles OR practices OR processes OR techniques OR methods OR methodologies OR benefits OR advantages OR organizations OR development OR operations))
Springer Link	DevOps AND "DevOps" AND (adoption OR principles OR practices OR methods OR techniques OR processes OR methodologies OR hindrances OR challenges OR impediments OR benefits OR advantages OR software OR IT OR organizations OR development OR operations)
Science Direct	(DevOps) AND (adoption OR impediments OR challenges OR hindrances OR principles OR practices OR processes OR techniques OR methods OR methodologies OR benefits OR advantages OR software OR IT OR organizations OR development OR operations)
ACM Digital Library	(+DevOps adoption principles practices methods techniques methodologies processes hindrances challenges impediments benefits advantages software IT organizations development operations)

3.3 Inclusion/Exclusion Criteria

"Study selection criteria are used to determine which studies are included in, or excluded from, a systematic review. It is usually helpful to pilot the selection criteria on a subset of primary studies". The following inclusion/exclusion criteria has been defined to remove irrelevant and duplicate studies from the database search results. This criteria is later applied to the search results to obtain a set of primary studies.

Table 4 – I/E Criteria

Inclusion Criteria	**Exclusion Criteria**
• Studies that are available in English language. • Studies that are available in full text. • Conference articles and journal articles. • Studies published between 2011-2018. • Peer-reviewed studies. • Studies that focus on the organizational aspects of DevOps. • Studies that focus on the benefits, challenges of adopting DevOps in organizations. • Studies that focus on the principles and practices of DevOps.	• Studies that are not available in English language. • Studies that are not available in full text. • Outdated conference papers. • Duplicate studies. • Grey literature. • Studies that focus only on cloud computing.

3.4 Quality Assessment Criteria

The following quality criteria has been devised to assess the quality of the selected studies. Kitchenham et al. (Kitchenham & Charters, 2007) state that quality assessment is critical to determine the strength of the inferences of primary studies. The quality criteria is applied to the finalized set of primary studies obtained after the database search.

Table 5 – Quality Assesment Criteria

	Yes	No	Partially
• Are the aims and objectives of the research clear?	X		
• Does the study report on the benefits of adopting DevOps in software organizations?	X		
• Does the study report on the challenges faced while adopting DevOps in software organizations?			X
• Does the study describe about the various DevOps principles and practices?	X		
• Is the research methodology clearly specified?	X		
• Are the results of the research clear?			X
• Are the conclusions of the research clearly stated?	X		

3.5 Conducting SLR

Based on the keywords and strings search defined in the previous section, conducted the search in the different databases. Excluded articles which are not related to information technology, computer science, and software engineering to ensure the relevancy of the papers. However, since DevOps is a new concept, and the thesis does not have a wide range of related papers, used the selection criteria mentioned in the previous section to sort the papers before preparing them for extraction. To ensure selecting high-quality papers which might contribute to the thesis, a set of keywords defined to search for the metadata of the papers at first and then conducted a full- text search. To summarize the results of the search, the articles which contribute to the thesis are few. However, the researcher has to extract the data from the articles to draw the desired conclusion.

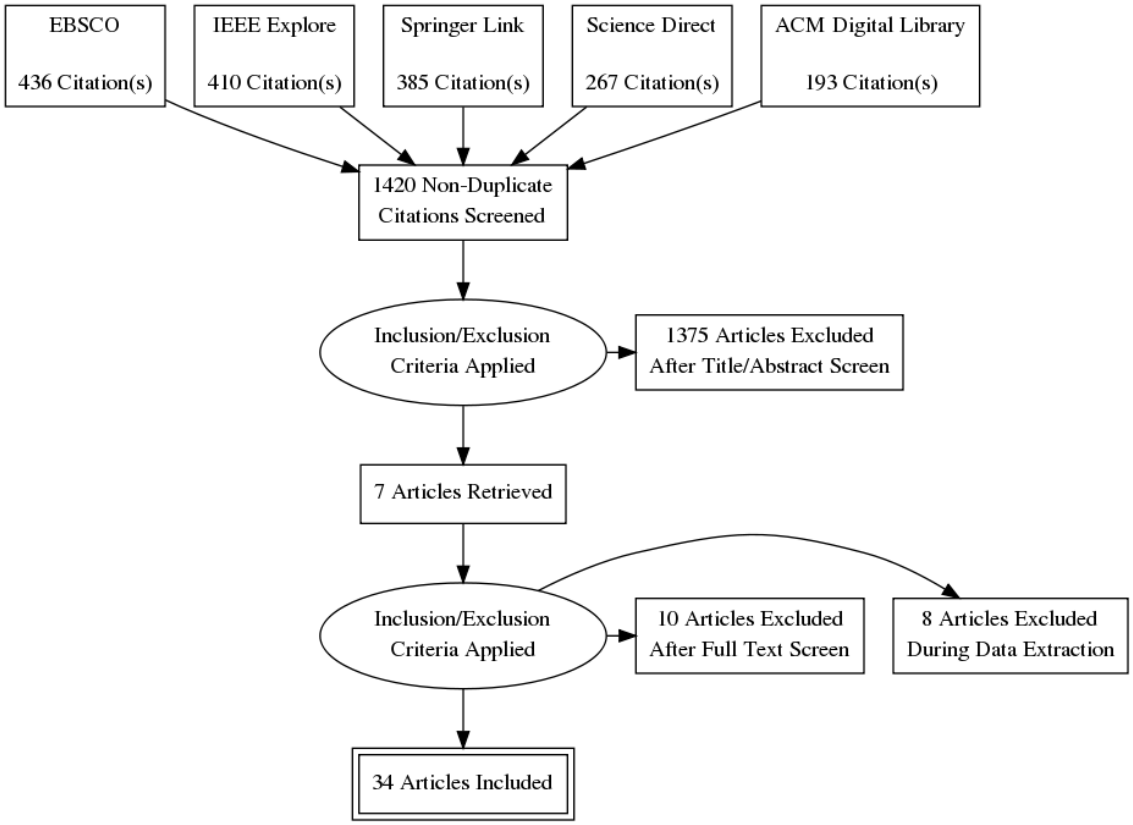

Figure 4 - Data selection process

After all process, it can be concluded that there are 34 papers selected (see Figure 3). Those articles have explicitly mentioned one of the defined keywords search strings and relevant to the field of the research. Moreover, it was noticeable the different level of relevance to the topic. To eliminate the non-relevant texts, the introduction and the conclusion of each paper have to be checked for the empirical data and any connections and links which might relate to DevOps core values based on the evaluation criteria discussed in the previous section.

4 SLR Analysis

4.1 Selected papers

The 34 papers that are finally selected for the analysis are shown below in Table 4.

Table 6 – Selected papers

Author	Year
Cito, Oliveira, Leitner, Nagpurkar, & Gall	2017
Shahin	2015
Ferry, Chauvel, Song, & Solberg	2015
de França, Jeronimo, & Travassos,	2016
Perera, Bandara, & Perera,	2017
Zhu et al.,	2016
Bass, Ravichandran, Taylor, & Waterhouse,	2017
Bang, Chung, Choh, & Dupuis,	2013
Olszewska & Waldén,	2015
Elberzhager & Arif,	2017
Anand Srivatsav & Swetha Bindu,	2016
Yli-Huumo, Maglyas, & Smolander,	2016
Austel et al.,	2015
Aljundi,	2018
Colomo-Palacios, Fernandes, Soto, & Larrucea,	2018
Kattepur & Nambiar,	2017
Wettinger, Andrikopoulos, & Leymann,	2015
Magoutis et al.,	2015
Lwakatare, Kuvaja, & Oivo,	2015
Dennehy & Conboy,	2017
Abdelkebir, Maleh, & Belaissaoui,	2017
Jones, Noppen, & Lettice,	2016
Stahl et al.,	2017
Dittrich,	2016
Fitzgerald & Stol,	2014
Krusche & Alperowitz,	2014
Familiar & Barnes,	2017
Furfaro, Gallo, Garro, Saccà, & Tundis,	2016
Van Heesch, Theunissen, Zimmermann, & Zdun,	2017
Vallon, da Silva Estácio, Prikladnicki, & Grechenig,	2018
Janes, Lenarduzzi, & Stan,	2017
Stillwell & Coutinho,	2015
Di Nitto, Jamshidi, Guerriero, Spais, & Tamburri,	2016
Shahin, Babar, & Zhu,	2016

Since DevOps is a relatively new concept, the resources of the publication year should be limited to the years 2014-2018. The massive majority of the studies related to DevOps published after 2015. Figure 5 shows the number of selected papers per year.

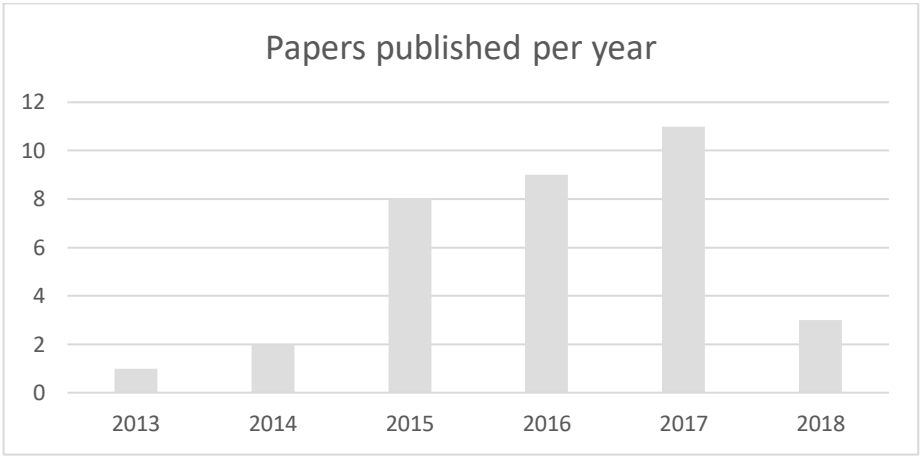

Figure 5 – Papers published per year

4.2 Papers analysis

To make the analysis of the paper less complicated, the papers categorized based on correspondence to one of four of DevOps core values culture, automation, measurement, and sharing. Any such classification enables to draw a more accurate conclusion which describes the mentioned practices related to each one of the core values CAMS observed in DevOps. The analysis is separated into four sections, each section reflects one of DevOps core values CAMS. Additionally, each dimension has one or more patterns to a set of DevOps practices.

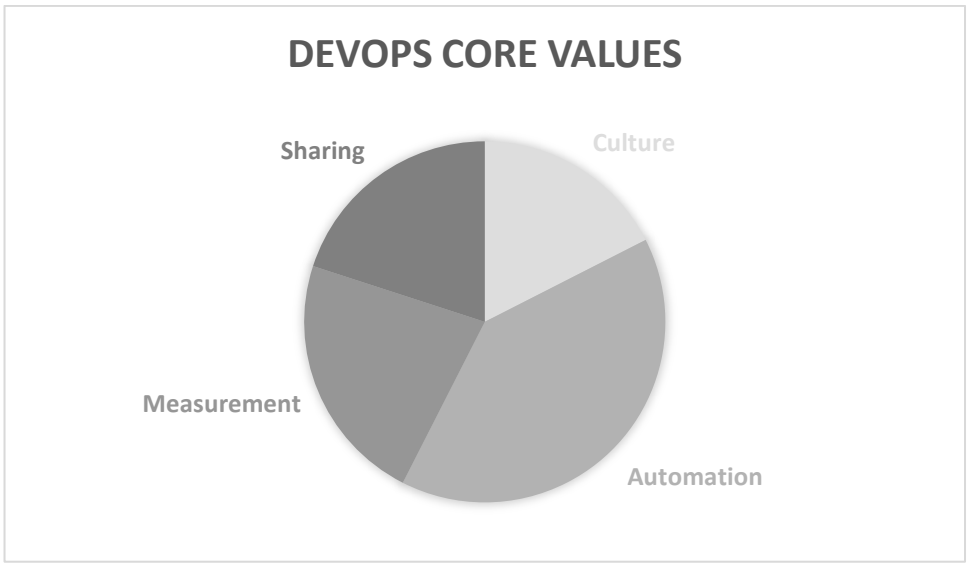

Figure 6 – DevOps Core Values

Table 7 – SLR DevOps core values

SLR	Reference	Culture	Automation	Measurements	Sharing
1	(Cito, Oliveira, Leitner, Nagpurkar, & Gall, 2017)	X			X
2	(Shahin, 2015)	X	X		
3	(Ferry, Chauvel, Song, & Solberg, 2015)	X	X		
4	(de França, Jeronimo, & Travassos, 2016)	X		X	
5	(Perera, Bandara, & Perera, 2017)	X	X		
6	(Weber, Nepal, & Zhu, 2016)	X			
7	(Bass, Ravichandran, Taylor, & Waterhouse, 2017)	X	X	X	
8	(Bang et al., 2013)		X		X
9	(Olszewska & Waldén, 2015)		X		
10	(Elberzhager & Arif, 2017)			X	X
11	(Anand Srivatsav & Swetha Bindu, 2016)			X	
12	(Yli-Huumo, Maglyas, & Smolander, 2016)			X	
13	(Austel et al., 2015)			X	
14	(Aljundi, 2018)		X	X	X
15	(Colomo-Palacios, Fernandes, Soto-Acosta, & Larrucea, 2018)	X		X	
16	(Kattepur & Nambiar, 2017)			X	
17	(Wettinger, Andrikopoulos, & Leymann, 2015)				X
18	(Magoutis et al., 2015)				X
19	(Lwakatare, Kuvaja, & Oivo, 2015)	X			X
20	(Dennehy & Conboy, 2017)				X
21	(Abdelkebir, Maleh, & Belaissaoui, 2017)			X	X
22	(Jones, Noppen, & Lettice, 2016)		X		
23	(Stahl, Martensson, & Bosch, 2017)	X	X		
24	(Dittrich, 2016)	X	X	X	
25	(Fitzgerald & Stol, 2014)	X	X	X	
26	(Krusche & Alperowitz, 2014)				X
27	(Familiar & Barnes, 2017)		X		
28	(Furfaro, Gallo, Garro, Saccà, & Tundis, 2016)	X	X		
29	(Van Heesch, Theunissen, Zimmermann, & Zdun, 2017)		X		
30	(Vallon, da Silva Estácio, Prikladnicki, & Grechenig, 2018)		X		X
31	(Janes, Lenarduzzi, & Stan, 2017)		X	X	
32	(Stillwell & Coutinho, 2015)	X	X		X
33	(Di Nitto, Jamshidi, Guerriero, Spais, & Tamburri, 2016)	X	X		
34	(Shahin, Babar, & Zhu, 2016)	X	X		

Table 7 shows the papers and their relevance to one or more of DevOps core values classifications.

4.2.1 Culture

Sixteen papers related to culture, one of DevOps core values and reflect the practices reported in the literature. Cultural values in DevOps context are related to the individual's attitudes and the way they communicate with their colleagues within their team and other teams. DevOps is about culture (Stahl et al., 2017): a "culture that stresses collaboration and integration between software developers, operations personnel, and everyone involved in the design, creation, development, and delivery of software". DevOps culture related values focus more on collective performance evaluation to assess the overall performance of the teams. Moreover, DevOps aims to establish a high level of trust among teams and individuals through encouraging effective communication and mutual learning, since it creates a stable basis for DevOps culture. Setting up DevOps cultural foundation among teams make it simpler to expand open to changes mentality, individual responsibility and respect among team members. Table 8 lists the practices collected on the systematic literature review and the papers where it was located.

Table 8 – Culture practices reported in the literature

Nº	**Practices related values**	**SLR Paper**
1	Patterns of empathy, support a good working environment between software development and operations	2
2	Having both software developers and operations personnel responsible for handling production incidents	2
3	Culture of trust and collective performance evaluation	3
4	Effective interdepartmental communication	4
5	Mutual learning	5
6	Taking care of international / national relevant aspects	5
7	Respect and cooperation among team members and managers	5
8	Shared sense of vision across multiple teams, directly aligned to the business and its customers	6
9	Creating new shared values and behaviors across IT teams	6
10	Collective emphasis on service as a whole, not on discreet functional elements or processes	6
11	Organizational culture is strong and supports the integration of new personnel and ideas that count on the support of relevant people inside the structure	15
12	In micro services development, teams and sub teams should develop services and test them individually	32
13	Sharing responsibility among different team members will increase the speed and the frequency to the demanded level by DevOps	32

14	Keeping Operations team aware of any changes, configurations and release	32
15	Prioritization of work and systematic testing practices	33
16	Constant communication channels with customers to get feedback	33
17	Collaborating closely with stakeholders to articulate intent and agreement on the risk-mitigation strategies	33
18	Rewarding or recognizing teams or groups as a whole, avoiding individuality	34
19	Central Dashboard for all the stakeholder to help to monitor the performance and make better decisions	34
20	Increased scope of responsibilities for developers	34

4.2.2 Automation

Nineteen papers of the selected ones discuss automation along with its tools and best practices. Automation supports DevOps ecosystem by allowing organizations to automate their testing, deployment, and integration. Moreover, Automation helps the company to reduce time and resources consumption on repetitive tasks and increases the efficiency of handling different tasks by focusing on consistency. Automation received reputation when the software quality and speed to market became an critical requirement for the success inside the industry, many tasks were repetitive and consume a lot of time, effort, and resources. As a consequence, the need for automation has become vital.

DevOps support and enhance continuously building, deploying, testing configuring, and so on. From there the automation has become one of DevOps values due to the support it gives the continuous environment DevOps aim to achieve. An instance of the practices in the automation dimension included the use of tools, such as Chef and Puppet, to automate and keep code for infrastructure provisioning and configuration settings. Table 9 lists the practices collected on the systematic literature review and the papers where it was determined.

Table 9 – Automation practices reported in the literature

Nº	**Practices Related values**	**SLR Paper**
1	Deployment automation, data management and continuous integration tasks are usually done by small teams of experts	2
2	The implementation also included automatic verification of software deployment to verify that the system behaved as expected after each software deployment	2
3	Keeping changes independent and preventing / minimizing manual tests from occurring in a release	2
4	When automation fail to restart the components, an emergency team is notified to resolve production incidents	3
5	Accelerating through automation to adapt to rapid change	5
6	Automating repetitive tasks	7
7	Infrastructure automation / consistency	7
8	Automation of the collection of metrics	8
9	Automation will be key for testing to become established within both continuous integration and continuous delivery processes	9
10	Test Automation for avoiding applications that are rushed through the development cycle without adequate testing often encounter costly defects that impact the customer relationship	13
11	Using Agile / Scrum methodology	23
12	Continuous: delivery, deployment and integration	25
13	Continuous software quality assurance monitoring. Code smell removal and detection	31
14	Continuous analytics for business needs	33
15	Automation the collection of the operational data to monitor the application state in production	34
16	Isolating running services. Host environment through either virtualization or operating system specific methods of containerization	32

4.2.3 Measurement

Measurement plays a significant role it plays to help improve the code's quality and performance. Measurement helps address the level of quality. To improve, it should be able to measure first. Software measurement helps to identify vulnerability detection, quality assessments, productivity enhancements compliance management and development practice improvements. Table 10 lists the practices gathered on the systematic literature review and the papers where it was found.

Table 10 – Measurement practices reported in the literature

Nº	**Practices Related values**	**SLR Paper**
1	Quality assurance monitoring to help developers to detect code smell	4
2	Measurement should focus on essential problems and have an aim behind it. Any meaningless measurement should be avoided	4
3	Enhance Built-in quality mentality with tools to closely monitor quality during production	6
4	Measurement for understanding code and environment	
5	Monitoring business through metrics	7
6	Monitoring operations for improving product time-to-market	7
7	Monitoring teams through metrics	7
8	Creating the necessary framework for such metrics requires closer consideration of the very reasoning behind DevOps adoption	10
9	Environment has to be created where the defined goals can be achieved. Goals should be defined and concrete qualitative and quantitative data be gathered and evaluated	10
10	Definition of metrics (using a dashboard) to get a better understanding of what changed and how it changed	10
11	Measurement quantifies the benefit and cost of known TD in a software system through estimation techniques or estimates the level of the overall TD in a system	12
12	Measurement practices used either data available in project management tools (JIRA, Wiki), or a specific tool to measure TD (SonarQube)	12
13	Measuring customer satisfaction and quality product	31

4.2.4 Sharing

DevOps ecosystem depends on the attitude of the individuals and their relationships with their colleagues or members of other teams. Sharing, as DevOps value, focus on two main factors: sharing personal learning and sharing project information. All the collected data need to be disseminated to learn to address the failure to avoid it in future projects. Sharing is not just telling facts, it is transferring ideas across teams. Table 11 lists the practices gathered on the systematic literature review and the papers where it was found.

Table 11 – Sharing practices reported in the literature

Nº	Practices Related values	SLR Paper
1	Collaborative services and responsible teams. Each service considered a standalone process and all the services communicate with HTTP/REST API	1
2	Logs can bridge the gap and share critical information between dev and ops teams. Log statements and other runtime information, data visualizations	8
3	Sharing personal learning and know-how	10
4	Sharing personal information	10
5	DevOps is more about the agile culture and knowledge-sharing than about specific tools	12
6	Open-source communities affiliated with tools are publicly sharing reusable artifacts to package, deploy, and operate middleware and application components	17
7	Participants provide feedback	18
8	The foreseen benefits for the users relate to knowledge sharing, learning by example, finding resources related to one's interests through the concepts of similar models and model contributors, as well as network building with field experts	18
9	Exhibition the same driving goals and values (i.e. small and quick changes with a focus on the end customer) but from differing perspectives	19
10	TOSCA and IasC are infrastructures that support sharing	21
11	Sharing status of projects, services between teams will reduce configuration time	26
12	Direction of communication: vertical vs horizontal	30
13	Sharing KSA: Knowledge, Skills and Abilities	32
14	Sharing resources to reduce costs	32

5 Conclusions

DevOps is a new phenomenon that gained huge popularity in the past few years. Several major companies like Facebook, Netflix and Yahoo are implementing DevOps. In general, the literature about DevOps is particularly lacking any clear standard to define the practices and the tools, even the definition of DevOps vary from one journal to another.

This research is thus focused on contributing towards a better understanding of how DevOps works in an organization concerning its four main pillars.

5.1 Contributions

Software practitioners and researchers seeking to explore the DevOps phenomenon can learn from this thesis about the meaning of the DevOps concept, its benefits and challenges, and principles which are explained in detail. The main contributions are as follows.

5.1.1 First contribution: Definition of DevOps

The first contribution of the thesis is the definition of DevOps and its main characterizing elements. The thesis explores the definition of DevOps from the existing literature and software practitioners. Based on the findings, the definition (Dyck, Penners, & Lichter, 2015) is enhanced by incorporating the different views of software practitioners' understanding of the concept. As such, DevOps in the literature and amongst software practitioners can best be understood as: 'a mindset change substantiated with a set of practices to encourage cross-functional collaboration between teams – especially development and IT operations – within a software development organization, in order to operate resilient systems and accelerate the delivery of changes'.

5.1.2 Second contribution: Benefits and Challenges

The second contribution of the thesis is about the benefits and challenges of DevOps adoption and implementation. Through DevOps, software practitioners perceive benefits of improvements in the speed of delivering software changes, the quality of the system, collaboration and productivity amongst stakeholders in software development and operations teams. However, it was observed that the benefits are not just as a result of DevOps, but by also applying supporting practices and principles of agile and lean software development, code review, test automation in quality assurance, etc.

5.1.3 Third contribution: Set of Practices

The third contribution of the thesis is a set of practices gathering during the adoption and implementation of DevOps in different contexts. Data extracted in SLR listed in a way to makes it easier to understand, by demonstrating the different core values of DevOps, and how these values can be enhanced by following a set of practices. DevOps practices are grouped into four categories: culture, automation, measurement and sharing.

5.2 Future research

As far as future work is concerned, it has been observed that there are very few high-quality studies related to DevOps, as it is a relatively new concept. There is a lot of scope for empirical work in this domain.

In this study, the benefits of adopting DevOps, challenges, solutions, practices, and problems were studied in general DevOps adoption scenarios. As the adoption of DevOps practices in organizations is dependent on factors like organizational structure, capabilities, project needs, requirements, and project types, some of the identified results could be contextual.

For example, the challenges faced in a small organization adopting DevOps might vary from the challenges faced in a large organization. Most of the DevOps initiatives are trialed in small teams or in few numbers of software products and services. It is not yet clear how hard it will be to extend the practices across teams as well as in aligning DevOps with existing software development practices that are used at larger scale? In addition, how such practices will be used in implementing enterprise applications such as ERP systems.

A case study can be performed to validate the results. The cause-effect factors that may impact the results identified can be investigated by conducting an empirical study. For example, the impact of culture on DevOps adoption can be investigated. Also, no studies address the solutions to overcome DevOps adoption challenges and problems faced by DevOps teams during continuous integration, deployment, and testing. It has been covered this gap, but more research could be performed in this aspect.

Further, it is learned that for transitioning to DevOps, a lot of changes must be made in an organization. It would be interesting to investigate what changes must be made in an organization for transitioning into a full DevOps state. There is currently no proper research on the tools needed for supporting DevOps. The availability of DevOps tools and their functioning can be studied.

6 References

Abdelkebir, S., Maleh, Y., & Belaissaoui, M. (2017). An Agile Framework for ITS Management In Organizations: A Case Study Based on DevOps. In *Proceedings of the 2Nd International Conference on Computing and Wireless Communication Systems* (p. 67:1–67:8). New York, NY, USA: ACM. https://doi.org/10.1145/3167486.3167556

Austel, P., Chen, H., Dube, P., Mikalsen, T., Rouvellou, I., Sharma, U., … Wang, Y. (2015). A PaaS for Composite Analytics Solutions. In *Proceedings of the 12th ACM International Conference on Computing Frontiers* (p. 51:1–51:8). New York, NY, USA: ACM. https://doi.org/10.1145/2742854.2747281

Bang, S. K., Chung, S., Choh, Y., & Dupuis, M. (2013). A grounded theory analysis of modern web applications. *The 2Nd Annual Conference*, 61–62. https://doi.org/10.1145/2512209.2512229

Bass, L., Ravichandran, A., Taylor, K., & Waterhouse, P. (2017). DevOps for Digital Leaders Reignite Business with a Modern DevOps-Enabled Software Factory. *IEEE Software*, *35*(1), 8–10. https://doi.org/10.1109/MS.2017.4541051

Callanan, M., & Spillane, A. (2016). DevOps: Making It Easy to Do the Right Thing. *IEEE Software*, *33*(3), 53–59. https://doi.org/10.1109/MS.2016.66

Cito, J., Oliveira, F., Leitner, P., Nagpurkar, P., & Gall, H. C. (2017). Context-based Analytics: Establishing Explicit Links Between Runtime Traces and Source Code. In *Proceedings of the 39th International Conference on Software Engineering: Software Engineering in Practice Track* (pp. 193–202). Piscataway, NJ, USA: IEEE Press. https://doi.org/10.1109/ICSE-SEIP.2017.1

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*. https://doi.org/10.1016/j.ijinfomgt.2017.11.005

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

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

Di Nitto, E., Jamshidi, P., Guerriero, M., Spais, I., & Tamburri, D. A. (2016). A Software Architecture Framework for Quality-aware DevOps. In *Proceedings of the 2Nd International Workshop on Quality-Aware DevOps* (pp. 12–17). New York, NY, USA: ACM. https://doi.org/10.1145/2945408.2945411

Dittrich, Y. (2016). What does it mean to use a method? Towards a practice theory for software engineering. *Information and Software Technology*. https://doi.org/10.1016/j.infsof.2015.07.001

Dyck, A., Penners, R., & Lichter, H. (2015). Towards Definitions for Release Engineering and DevOps. In *Proceedings of the Third International Workshop on Release Engineering* (p. 3). Piscataway, NJ, USA: IEEE Press. Retrieved from http://dl.acm.org/citation.cfm?id=2820690.2820694

Edwards, D. (2010). What is DevOps? Retrieved from http://dev2ops.org/2010/02/what-is-devops/

Elberzhager, F., & Arif, T. (2017). Software Quality. Complexity and Challenges of Software Engineering in Emerging Technologies, *269*, 33–44. https://doi.org/10.1007/978-3-319-49421-0

Familiar, B., & Barnes, J. (2017). *Business in real-time using Azure IoT and Cortana intelligence suite : driving your digital transformation*. https://doi.org/10.1007/978-1-4842-2650-6

Ferry, N., Chauvel, F., Song, H., & Solberg, A. (2015). Continous Deployment of Multi-cloud Systems. In *Proceedings of the 1st International Workshop on Quality-Aware DevOps* (pp. 27–28). New York, NY, USA: ACM. https://doi.org/10.1145/2804371.2804377

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

Furfaro, A., Gallo, T., Garro, A., Saccà, D., & Tundis, A. (2016). ResDevOps: A Software Engineering Framework for Achieving Long-Lasting Complex Systems. *Proceedings - 2016 IEEE 24th International Requirements Engineering Conference, RE 2016*, 246–255. https://doi.org/10.1109/RE.2016.15

Gottesheim, W. (2015). Challenges, Benefits and Best Practices of Performance Focused DevOps. In *Proceedings of the 4th International Workshop on Large-Scale Testing* (p. 3). New York, NY, USA: ACM. https://doi.org/10.1145/2693182.2693187

Janes, A., Lenarduzzi, V., & Stan, A. C. (2017). A Continuous Software Quality Monitoring Approach for Small and Medium Enterprises. In *Proceedings of the 8th ACM/SPEC on International Conference on Performance Engineering Companion* (pp. 97–100). New York, NY, USA: ACM. https://doi.org/10.1145/3053600.3053618

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

Kattepur, A., & Nambiar, M. (2017). Service demand modeling and performance prediction with single-user tests. *Performance Evaluation*. https://doi.org/10.1016/j.peva.2017.02.003

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

Krusche, S., & Alperowitz, L. (2014). Introduction of continuous delivery in multi-customer project courses. *Companion Proceedings of the 36th International Conference on Software Engineering - ICSE Companion 2014*, 335–343. https://doi.org/10.1145/2591062.2591163

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*, 212–217. https://doi.org/10.1007/978-3-319-18612-2

Lwakatare, L. E., Karvonen, T., Sauvola, T., Kuvaja, P., Olsson, H. H., Bosch, J., & Oivo, M. (2016). Towards DevOps in the embedded systems domain: Why is it so hard? In *Proceedings of the Annual Hawaii International Conference on System Sciences* (Vol. 2016–March, pp. 5437–5446). https://doi.org/10.1109/HICSS.2016.671

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

Magoutis, K., Papoulas, C., Papaioannou, A., Karniavoura, F., Akestoridis, D. G., Parotsidis,

N., … Stephanidis, C. (2015). Design and implementation of a social networking platform for cloud deployment specialists. *Journal of Internet Services and Applications*, *6*(1). https://doi.org/10.1186/s13174-015-0033-5

Olszewska, M., & Waldén, M. (2015). DevOps Meets Formal Modelling in High-criticality Complex Systems. In *Proceedings of the 1st International Workshop on Quality-Aware DevOps* (pp. 7–12). New York, NY, USA: ACM. https://doi.org/10.1145/2804371.2804373

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*, 281–287. https://doi.org/10.1109/ICTER.2016.7829932

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

Ramamurthy, R. (2016). THE DEVOPS CULTURE CHANGE GOES WAY BEYOND IT. *Siliconindia*, *19*(10), 30–31. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&db=bth&AN=128463216&site=ehost-live

Shafer, A., Debois, P., Gat, I., & Debois, P. (2011). Devops Metrics, (June).

Shahin, M. (2015). Architecting for DevOps and Continuous Deployment. *Proceedings of the ASWEC 2015 24th Australasian Software Engineering Conference on - ASWEC ' 15 Vol. II*, 147–148. https://doi.org/10.1145/2811681.2824996

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

Stahl, D., Martensson, T., & 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)*, 440–448. https://doi.org/10.1109/SEAA.2017.8114695

Stillwell, M., & Coutinho, J. G. F. (2015). A DevOps Approach to Integration of Software Components in an EU Research Project. In *Proceedings of the 1st International Workshop on Quality-Aware DevOps* (pp. 1–6). New York, NY, USA: ACM. https://doi.org/10.1145/2804371.2804372

Vallon, R., da Silva Estácio, B. J., Prikladnicki, R., & Grechenig, T. (2018). Systematic literature review on agile practices in global software development. *Information and Software Technology*. https://doi.org/10.1016/j.infsof.2017.12.004

Van Heesch, U., Theunissen, T., Zimmermann, O., & Zdun, U. (2017). Software Specification and Documentation in Continuous Software Development: A Focus Group Report. In *Proceedings of the 22Nd European Conference on Pattern Languages of Programs* (p. 35:1–35:13). New York, NY, USA: ACM. https://doi.org/10.1145/3147704.3147742

Weber, I., Nepal, S., & Zhu, L. (2016). Developing Dependable and Secure Cloud Applications. *IEEE Internet Computing*, *20*(3), 74–79. https://doi.org/10.1109/MIC.2016.67

Wettinger, J., Andrikopoulos, V., & Leymann, F. (2015). Automated capturing and systematic usage of DevOps knowledge for cloud applications. *Proceedings - 2015 IEEE International Conference on Cloud Engineering, IC2E 2015*, 60–65. https://doi.org/10.1109/IC2E.2015.23

Yli-Huumo, J., Maglyas, A., & Smolander, K. (2016). How do software development teams manage technical debt? – An empirical study. *Journal of Systems and Software*. https://doi.org/10.1016/j.jss.2016.05.018

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