DevOps Capabilities and Metrics

Ricardo Manuel Duarte Amaro

Thesis to obtain the Master of Science Degree in

Information and Enterprise Systems

Supervisors: Prof. Rúben Filipe de Sousa Pereira
Prof. Miguel Leitão Bignolas Mira da Silva

Examination Committee

Chairperson: Prof. Daniel Jorge Viegas Gonçalves
Supervisor: Prof. Rúben Filipe de Sousa Pereira
Member of the Committee: Prof. António Manuel Ferreira Rito da Silva

October 2021

Acknowledgments

A special thank you to Prof. Rúben Pereira and Prof. Miguel Mira da Silva, this thesis supervisors, since they deserve my special gratitude for their insights, encouragement, and knowledge sharing, which made this research possible.

I also want to thank my lovely wife, Dália, my son Vicente and my daughter Maria Rita, for their unconditional support throughout my life. It was crucial to have their assistance and understanding, to complete this work.

Last but not least, I'd want to thank all the participants in the interviews, my friends and coworkers who have helped me grow as a person and have always been there for me through good and bad times.

To each and every one of you – Thank you.

Abstract

Nowadays, IT organizations face ever-changing consumer demands, competitiveness, regulatory environments, and sophisticated external threats. As a result, they seek a competitive advantage by using DevOps and its capabilities, such as improving user experience, increasing productivity, and team collaboration.

However, DevOps adoption remains inconsistent, emphasizing the need to provide relevant data and insights to management in their decision-making process to enhance efficiency while applying DevOps capabilities. Unfortunately, there is a lack of systematization between the effective use of these capabilities and the ideal metrics for each one.

Therefore, Design Science Research is done with two Multivocal Literature Reviews to elicit the main DevOps metrics and capabilities and semi-structured interviews to build an outcome-based capability evaluation matrix, focusing on metrics (KPIs) for promoting DevOps adoption. A definition of DevOps capability and another for metrics is identified, along with 37 DevOps capabilities and 24 main metrics are defined and categorized.

It is concluded that cultural capabilities have the highest overall impact. Empowering teams to make decisions and organizational culture are top categories. Capabilities are, dynamic and have been growing and changing over the years, being defined by the ability of an organization to perform practices. The five top metrics that an organization should start by measuring are, in order, M03-Deployment Frequency (DF), M01-Mean Time To Recover/Restore (MTTR), M42-Team happiness, M02-Mean Lead-time for Changes (MLT) and M04-Change Failure Rate (CFR).

Keywords

DevOps; Capabilities; Competencies; Metrics; Performance; Adoption.

Resumo

Atualmente, as organizações de TI, enfrentam exigências dos consumidores, competitividade, ambientes regulamentares e ameaças externas sofisticadas em constante mudança. Como resultado, procuram uma vantagem competitiva utilizando DevOps e as suas capacidades, tais como melhorar a experiência do utilizador, aumentar a produtividade, e a colaboração em equipa.

No entanto, a adoção do DevOps continua a ser inconsistente, enfatizando a necessidade de fornecer dados e conhecimentos relevantes à gestão no seu processo de tomada de decisões para melhorar a eficiência enquanto se aplicam as capacidades do DevOps. Infelizmente, existe uma falta de sistematização entre a utilização eficaz destas capacidades e as métricas ideais para cada uma delas.

Assim, Design Science Research é realizada com duas Multivocal Literature Reviews para obter as principais métricas e capacidades DevOps e entrevistas semi-estruturadas para construir uma matriz de avaliação de capacidades baseada em resultados, centrada em métricas (KPIs) para promover a adopção de DevOps. É identificada uma definição de capacidade DevOps e outra para métricas, com 37 capacidades DevOps e 24 métricas principais sendo definidas e categorizadas.

Conclui-se que as capacidades culturais têm o maior impacto global. Equipas capacitadoras para tomar decisões e cultura organizacional são categorias de topo. As capacidades são, dinâmicas e têm vindo a crescer e a mudar ao longo dos anos, definidas pela capacidade de uma organização para realizar práticas. As cinco principais métricas que uma organização deve começar por medir são, por ordem, M03-DF, M01-MTTR, M42-Team happiness, M02-MLT e M04-CFR.

Palavras Chave

DevOps; Capacidades; Competências; Métricas; Desempenho; Adoção.

Contents

List of Figures

List of Tables

Listings

Acronyms

CALMS Culture, Automation, Lean principles, Measuring and Sharing

CAMS Culture, Automation, Measuring and Sharing

CFR Change Failure Rate

CI *Continuous Integration*

CTV Cycle Time Value

DevOps Developer(Dev) and Operations(Ops)

DevSecOps Developer(Dev), Security(Sec) and Operations(Ops)

DF Deployment Frequency

DSR Design Science Research

KPI Key Performance Indicator

MLT Mean Lead-time for Changes

MLR Multivocal Literature Review

MTTD Mean Time To Detection

MTTF Mean time to failure

MTTR Mean Time To Recover/Restore

NIST US National Institute of Standards and Technologies

SDLC Software Development Life Cycle

SLA Service Level Agreement

SLI Service level Indicator

SLO Service level Objective

SLR Systematic Literature Review

SCM Software Configuration Management

SE Software Engineering

UWR Unplanned Work Rate

WIP Work in Progress

CD *Continuous Delivery or Deployment*

1

Introduction

Contents

This chapter aims to contextualize the research as well as present the motivation for performing it, as well as its objectives, deliverables, scope and relevance. This chapter also includes the outline of the remaining thesis in Section 1.4.

1.1 Motivation

In today's world, IT organizations are increasingly challenged with ever-changing customer requirements, competition, regulatory environments and sophisticated outside threats [10].

Therefore, in organizations where software development is part of the core business, having a competitive advantage by doing things better than competitors [11], like delivering and supporting this software quickly, with reliability and in a predictable form has become increasingly important [12].

However, this process generates a complexity and inefficiency, associated with the silos between development and operations [13]. The need for frequent software delivery, without sustained builds, proper testing and release automation, generates burnout and pain in the engineers doing operations, decaying software delivery performance [14] and reliability.

In reaction to this broken process we witness the emergence of Developer(Dev) and Operations(Ops) (DevOps), an organizational approach that stresses empathy and encourages greater collaboration between engineering teams involved in the software delivery [15], in order to reduce development time, enhance deployment rates, increase stability, optimize Mean Time to Recover and reduce cost of deployment and implementation [16].

Management needs to have a clear vision of the steps to take ahead based on information and metrics in order to increase efficiency [11], therefore the success of applying DevOps capabilities, mentioned in Section 2.2, can be exposed if we find a strong relation to existing DevOps metrics. Furthermore, the successful implementation of the complex [17] process of DevOps capabilities demands control by a rigorous systematization for self-assessment, which in turn should result in growth of the maturity levels that will lead to improved levels of performance [18].

An effective way to control and assess these levels of maturity and adoption would be to use very specific metrics that assess existing capabilities, informing, for example, whether a given software delivery process within the pipeline is performing optimally or could be improved.

This would support management decision-making process by supplying relevant information that conveys whether the adoption of DevOps is improving within the company, so that management can take appropriate action if needed.

1.2 Problem

To the best of the author's knowledge, no study has been conducted that compares various DevOps capabilities or practices with existing DevOps metrics [19], so the problem of this thesis is identified as follows:

Problem. *The assessment of the desired success of each DevOps capability or practice is irregular due to a lack of systematization between the successful application of these capabilities or practices and the ideal metrics for each one.*

1.3 Objectives and Deliverables

While DevOps adoption success is irregular, as a few impediments exist, shown by Smeds et al. [20] and measuring its maturity can be hard, relating different DevOps capabilities with existing DevOps metrics will support management decision process towards increasing performance in the Software Development Life Cycle (SDLC) within the organization. The main objective of maturity models is to evaluate and improve the organization's practices by creating an improvement roadmap [21] and finding the relation between DevOps metrics and DevOps capabilities or practices will facilitate adopting DevOps successfully.

However, since there exists a lack of systematization of the different DevOps capabilities or practices in existing DevOps metrics, the intent of this research is to explore a relationship between the metrics and the capabilities/practices, in order to elicit the main DevOps metrics for each DevOps capability/practice, align the relations impacting positively each found metric and create an evaluation matrix of the DevOps capabilities/practices. For that purpose, the following research questions are used towards the mentioned goal:

RQ1. What are the main DevOps capabilities or practices?

RQ2. Where are capabilities and practices mentioned?

RQ3. How authors distinguish capabilities from practices?

RQ4. What are the main DevOps metrics?

RQ5. What is the purpose of each metric?

RQ6. Why is each metric important?

RQ7. How are DevOps capabilities categorized?

RQ8. How are the main metrics categorized?

RQ9. What DevOps capabilities have a positive impact in which main metrics?

1.4 Thesis Outline

In Chapter 1 the motivation, research problem, the objectives and deliverables of this research are exposed.

The remainder of the document is structured as follows. In Chapter 2 a theoretical background is given about DevOps, DevOps Capabilities and DevOps Metrics. In Chapter 3 a literature review done prior to this research is summarized giving evidence of no prior work done targeting the same goals and assessing the need for this research. In Chapter 4 the research methodology is defined to be Design Science Research (DSR), the build process is explained and methods of research are determined to be Multivocal Literature Review (MLR) and Semi-structured Interviews. In Chapter 5 details the investigation done using MLR and leading to 93 documents utilized in the discovery and definition of 37 DevOps capabilities. In Chapter 6 a DevOps metrics MLR is performed returning 114 documents from several sources that were analyzed, and 58 metrics were extracted leading to a filtered total of 22 main metrics in which 4 are the most relevant. In Chapter 7 a capability matrix is proposed based on the results from the two previous chapters and by conduction 21 semi-structured interviews. In Chapter 8 the capability matrix is evaluated over ten iterations of semi-structured interviews and a final validated version of the artifact is achieved. In Chapter 9 contains the conclusion of the research with limitations and future work.

2

Theoretical Background

Contents

This section provides a theoretical background for the topics discussed in this research, namely *DevOps*, *DevOps capabilities* and *DevOps metrics*.

2.1 DevOps

DevOps is an acronym for the Developer (Dev) and Operations teams (Ops), these engineering teams work collaboratively to eliminate so-called “information silos” [9]. No standard definition exists for DevOps. Blog posts on the topic are common, but they mostly differ on a concrete concept of the term. Jabbari et al. [22] proposed that DevOps is a development methodology aimed at bridging the gap between Development and Operations, emphasizing communication and collaboration, continuous integration, quality assurance and delivery with automated deployment utilizing a set of development practices. It can also be seen as a conceptual framework that is based on the capabilities, mentioned in page 10, focused on the acronym Culture, Automation, Measuring and Sharing (CAMS) [23]. Later, Jez Humble added to these four pillars, the Lean (L) pillar, becoming the acronym Culture, Automation, Lean principles, Measuring and Sharing (CALMS) [24].

Sousa et al. [25] in his paper about DevOps foundations and perspectives, emphasizes the new approach to software delivery that occurs through collaboration between development teams and operations, illustrated in Figure 2.1, as opposed to the traditional approach that is separated in organizational silos. This characteristic of good cooperation between IT Development and IT Operation teams is crucial in order to ensure successful deployment and operations of IT systems [26].

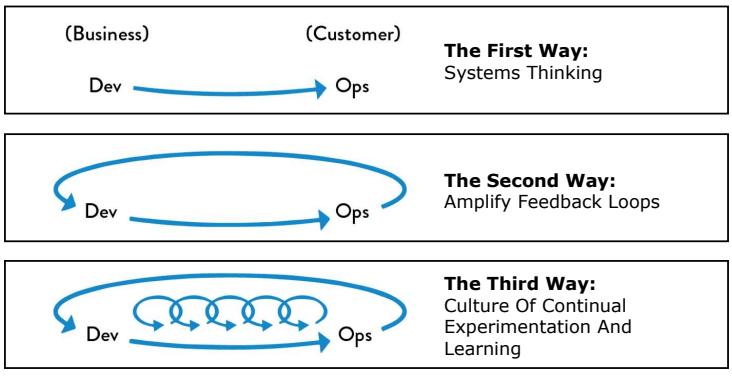

Figure 2.1: The Three Ways: The principles underpinning DevOps (adapted) [1]

DevOps is also a culture, movement, or practice emphasized on collaboration and communication, focused on improving the software release cycle speed to production and build automation of new software components, while keeping high quality, mentioned in Lwakatare et al. [27], where a literature review on the term DevOps concludes that DevOps is a change of mindset substantiated with a set of automated practices to encourage cross-functional team collaboration.

Lastly and according to Riungu-Kalliosaari et al. [28], DevOps is a set of practices aimed to reduce the time that a change made to a system takes to go into normal production, while ensuring high quality and the least friction and blame between teams as opposed to trust and empathy.

2.2 DevOps Capabilities

Wu et al. [29] defines operational capabilities as being "sets of skills, processes, and routines, developed within the operations management system, that are regularly used in solving its problems through configuring its operational resources". These can be strongly impacted by dynamic capabilities [30–33] as the processes that transform a firm's operational routines [31].

As seen in the research performed by Erich et al. [34–36] focused interviews on DevOps leading to principles and practices were done in six organizations [36]. The key extracted areas related to DevOps where: "culture of collaboration, automation, measurement, sharing, services, quality assurance, and governance". At that time, it was concluded that there existed very few academic studies evidencing DevOps principles and practices effectiveness. Notably, the term capability was already mentioned interchangeably with the meaning of DevOps ability to solve problems — "increased problem-solving capabilities" or ability to do automated testing — "automated testing capabilities" [36]. Thus leading to an ability or being capable of practicing something towards enabling DevOps.

2.3 DevOps Metrics

In literature, the first important questions leading to DevOps metrics being discussed are raised in 2010 [15]. How long would it take for an organization to deliver a single-line-of-code modification, and if that process is consistent and reproducible. Metrics are still hard to quantify since they encompass many aspects of the software delivery process, from analysis to development to release. Therefore, improving measuring the software delivery process is relevant and pursued.

In a study about Developer(Dev), Security(Sec) and Operations(Ops) (DevSecOps) by Prates et al. [37] goals include increasing the *Number of Continuous Delivery Cycles Per Month* related deployment frequency, lower Defect *Density Defect* or *Defect Burn Rate* indicating how quickly a team is addressing defects.

According to Forsgren et al. [19], organizations ideally, should start by collecting a baseline using surveys while continuing to build out system-based metrics, which should usually use data from different systems of record in the software delivery value stream. This way, businesses may obtain a more comprehensive perspective of their software delivery value chain and DevOps transformation efforts.

3

Literature Review

Contents

The main research objectives of this thesis focuses on relating DevOps capabilities and metrics. Before acknowledging the importance and uniqueness of this study, an initial search for related work, based on the Systematic Literature Review (SLR) protocol [38], was done in order to identify any existing studies that related these same topics.

3.1 DevOps Capabilities and Metrics SLR

This SLR targeted exposing sustained evidence of relevant literature for a set of research questions. In November 2020, a report was done using relevant keywords in a search string against a few datasets in attempt to discover earlier studies connected to this project that may already find answers to the suggested research topics.

- **Search String**: `(devops AND (metrics OR measures OR kpi OR indicators) AND (practices OR capabilities))`.
- **Datasets**: The search engines used were two brokers: Scopus[1] and Web of Science[2], in conjunction with IEEE[3], ACM[4] and EBSCO[5].

The *review protocol* used starts with searching datasets with the string, applying inclusion and exclusion criteria, screening abstracts, assess eligibility of full-text document and reaching a final document set. The inclusion and exclusion criteria, filters documents written in English, peer-reviewed and scientific papers, that explicit discusses DevOps, mentions DevOps metrics and Mentions DevOps capabilities or practices.

In the initial search step, *filter 1* (All fields; All documents) was used together with the search string, both present in Table 3.1. On a second pass, *filter 2* (Abstract; All documents) was used over the existing search results, therefore reducing the number of documents that have an abstract mentioning the keywords, narrowing down to a total of 74 papers. Applying inclusion & exclusion criteria *filter 3*, 59 articles remain. This leads to *filter 4*, which is defined to remove the duplicates from the list of results in order to obtain the set of documents to have abstracts screened.

After the selected set of abstracts were screened, a full-text document assess was done and finding 17 documents that were eligible to analyze and extract any information relevant for this research. The relation of final document set by database show that the most results remaining (5) came from ACM with 29.4%. For the cases of Scopus, Web Of Science, IEEE they all contributed with 4 relevant research

[1] https://www.scopus.com
[2] https://apps.webofknowledge.com
[3] https://ieeexplore.ieee.org
[4] https://dl.acm.org
[5] https://search.ebscohost.com

Table 3.1: Selection filters to narrow related work, as part of the SLR protocol

Database	Search String	Filter 1	Filter 2	Filter 3	Filter 4	Filter 5	Filter 6
Scopus	`(devops AND (metrics OR measures OR kpi OR indicators) AND (practices OR capabilities))`	443	31	31	25	14	4
Web Of Science	`(devops AND (metrics OR measures OR kpi OR indicators) AND (practices OR capabilities))`	18	12	12	8	6	4
IEEE	`(devops AND (metrics OR measures OR kpi OR indicators) AND (practices OR capabilities))`	11	7	7	7	7	4
ACM	`(devops AND (metrics OR measures OR kpi OR indicators) AND (practices OR capabilities))`	667	7	7	7	7	5
EBSCO	`(devops AND (metrics OR measures OR kpi OR indicators) AND (practices OR capabilities))`	28	17	2	2	2	0
	Total	1167	74	59	49	36	17

Filter 1 = All fields, All documents
Filter 2 = Abstract, All documents
Filter 3 = Peer-reviewed & relevant (inclusion/exclusion criteria)
Filter 4 = Remove duplicates
Filter 5 = After Abstracts Screened
Filter 6 = Full-text Document Assess

documents each, however EBSCO's results were also found in the other databases, thus showing no extra addition.

Important to note the distribution and growth of the selected papers shown over the years in relation to the publication seen in Figure 3.1. This reveals a growing interest in the last three years with an increase in volume of scientific research work related to DevOps metrics and capabilities. Thus, confirming the research objectives, potential usefulness of this thesis might have in this topic.

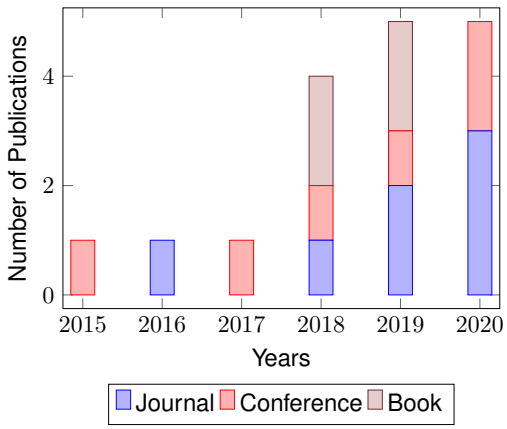

Figure 3.1: Relation of final documents publication year per type.

The first publication [39] from this set was a conference proceeding, that was followed by a journal article and another proceeding in the next two years, until that in 2018, the interest for this subject as grown considerably including in books and in the last year of 2020, seen in two conference proceedings

and three journal articles, with a total of five publications in each of the last two years.

3.2 Related Work

At this last stage of the SLR, the research question is analyzed, aligned with the objective of this review in face of the related work.

For assessing the full eligibility of all the related work and refine the level of relevance that could be attributed to each of the papers the following it was searched for any metrics[1] or DevOps metrics[2], any capabilities or practices[3], and a relation of DevOps metrics with capabilities[4]. The results seen in Table 3.2 go in line with the proposed objective in order to find what studies relate DevOps metrics to DevOps capabilities, but returns zero results for the research question.

As seen in Figure 3.2, from the initial 36 publications that were read, only 20 mention or enumerate DevOps metrics and 22 mention capabilities or practices. From those, only 17 match together to conform to the acceptance criteria of at least mentioning DevOps metrics and capabilities or practices.

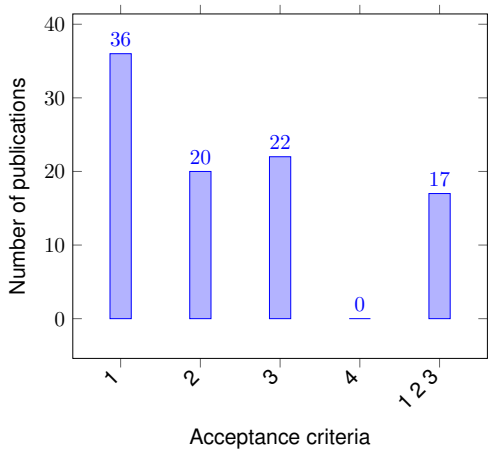

Figure 3.2: Relation of final documents with criteria

Interestingly, only one of these papers refers to the work of Senapathi et al. in *DevOps Capabilities, Practices, and Challenges* [4], *managing quality assurance challenges of DevOps through analytics*, only mentions five of the capabilities with a strong focus on quality assurance in the DevOps competency.

From the full reading performed, summarized in the previous list, interesting material was identified, but none of the articles was found to be relating directly DevOps metrics to DevOps capabilities.

Table 3.2: Relation of DevOps metrics and capabilities from full-text review

Paper Title	Mentions any metrics[1]	Mentions or enumerates DevOps Metrics[2]	Mentions any capabilities or practices[3]	Relates DevOps metrics with capabilities[4]
1. A DevOps Implementation Framework for Large Agile-Based Financial Organizations. [40]	yes	yes	yes	no
2. A Taxonomy of Software Delivery Performance Profiles: Investigating the Effects of DevOps Practices. [41]	yes	yes	yes	no
3. An agile framework for ITS management in organizations. [42]	yes	yes	yes	no
4. DevOpRET: Continuous reliability testing in DevOps. [43]	yes	yes	yes	no
5. DevOps and software quality: A systematic mapping. [44]	yes	yes	yes	no
6. Devops enhancement with continuous test optimization. [45]	yes	yes	yes	no
7. DevOps with continuous testing architecture and its metrics model. [46]	yes	yes	yes	no
8. DevSecOps metrics. [37]	yes	yes	yes	no
9. Dogfooding: Using IBM cloud services to monitor IBM cloud infrastructure. [47]	yes	yes	yes	no
10. Managing quality assurance challenges of Devops through analytics. [48]	yes	yes	yes	no
11. Non-Intrusive Anomaly Detection with Streaming Performance Metrics and Logs for DevOps in Public Clouds. [49]	yes	yes	yes	no
12. Omniscient devops analytics. [50]	yes	yes	yes	no
13. Self-Service Cybersecurity Monitoring as Enabler for DevSecOps. [51]	yes	yes	yes	no
14. Test Automation Process Improvement in a DevOps Team. [52]	yes	yes	yes	no
15. Towards a DevOps approach for software quality engineering. [39]	yes	yes	yes	no
16. Towards the use of the readily available tests from the release pipeline as performance tests. [53]	yes	yes	yes	no
17. Using Analytics to Guide Improvement during an Agile–DevOps Transformation. [54]	yes	yes	yes	no

4

Research Methodology

Contents

In an initial report, a Literature Review has been conducted to identify the problem of this research, reflected in Chapter 3. Here it is used the activities of Design Science Research (DSR) proposed by Hevner et al. [2] as the main research methodology.

For the build phase [55] two MLR [3] are conducted as mentioned in Section 4.2 to help identify the artifact. After having the initial artifact, it is planned to conduct the evaluation phase [56] using Semi-structured Interviews mentioned in Section 4.3, in order to support the findings of the initial research and improve the artifact.

4.1 Design Science Research

For the development of the proposed evaluation model for DevOps capabilities, Design Science fundamentals are applied for conducting this research in the field of Information Systems and Technologies, as initially suggested by March and Smith in 1995 [55] and nowadays by Hevner et al. [2], as the standard reference for the DSR methodology and in particularly the three cycles of Hevner [57].

This DSR is divided in two processes, build (process of construction of an artifact) supported by two MLR in Chapter 5, plus Chapter 6, resulting in a proposal in Chapter 7 and evaluate (to determine how well the artifact behaves) in Chapter 8. A DSR includes a set of synthetic and analytical techniques and perspectives for conducting research, with applicability in information systems. The design is characterized by specialized activities that produce an innovation [2], the design artifact or design theory as a means to improve the current state of practice, as well as existing research knowledge.

Hevner et al. [2] proposes a series of guidelines in DSR in Information System:

1. **Design as an Artifact**: DSR must produce a viable artifact that could come in various types [55] including constructs (vocabulary and symbols), models (abstractions and representations), methods (process stages to be followed to solve problems using IT), instantiations (implementations of constructs and models) [58].

2. **The Relevance of the Problem**: The essential target of DSR is to create innovation-based answers for significant and pertinent business issues.

3. **Project Evaluation**: The value, quality and viability of the design artifact should be thoroughly shown through proper evaluation methods.

4. **Research Contribution**: Effective research in design science ought to give clear and evident contributions in the territories in which the design artifact is applied, design fundamentals and/or design methodologies.

5. **Research Rigor**: DSR relies upon the use of rigorous strategies in both assessing and building the design artifact.

6. **Design as a Search Process**: The quest for a successful artifact relies upon the utilization of accessible means to accomplish the ideal outcomes while the laws in the environment of the problem are as yet fulfilled.

7. **Research Communication**: The presentation of research in design and science must be effective in both technology-oriented and managerial consultancy.

Baskerville et al. [59] underlines the dual mandate of the DSR, which is to use acquired knowledge to solve problems, create changes, or improve existing solutions; and at the same time to generate knowledge, perceptions and theoretical explanations. Hevner [57] calls the attention to the pragmatism of DSR in cycles of relevance and rigor, in the creation of artifacts. The DSR methodology process in the field of Information Systems, shown in Figure 4.1, is based on Develop/build and justify/evaluate [2].

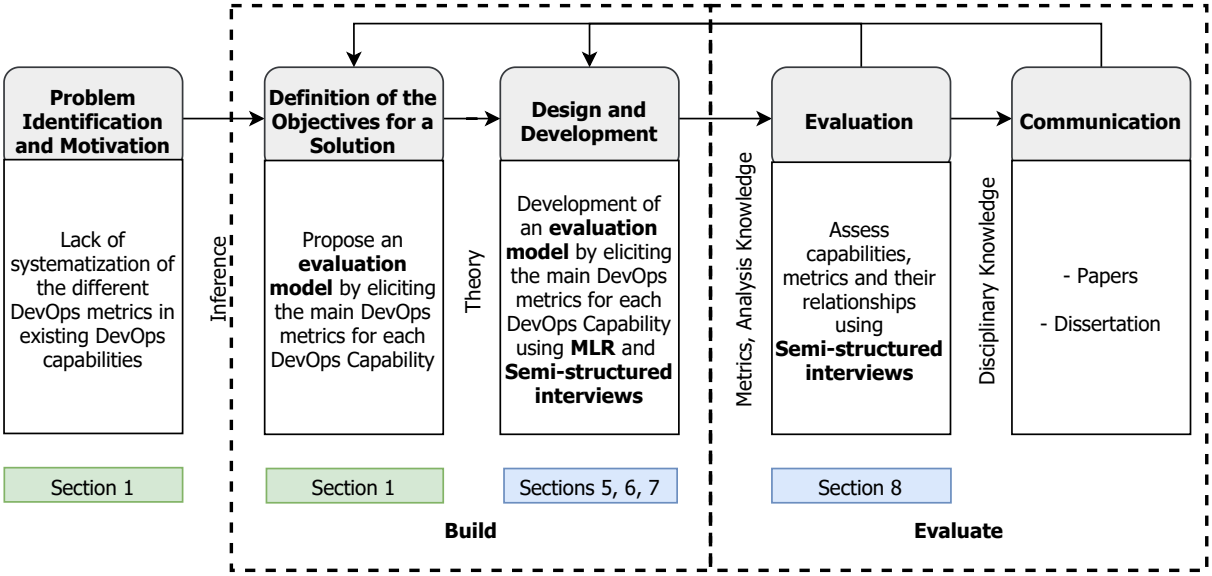

Figure 4.1: Adapted phases of the DSR process model [2]

1. **Identification of the problem and motivation**, defining the research problem and justifying the value of a solution. In the Introduction its identified the problem scope by being the Lack of systematization of the different DevOps metrics in existing DevOps capabilities and a purpose was defined.

2. **Definition of objectives for a solution**, inferring the objectives of a solution from the problem definition and knowledge of the state of the problem and possible solutions. The objectives can be quantitative or qualitative. For this work is proposed an evaluation model for DevOps capabilities.

3. **Design and development** of the artifact's desired functionality and its architecture followed by its creation. Such artifacts are potentially, with each defined broadly, constructs, models, methods,

or instantiations [2]. This research targets the development of an evaluation model of Devops capabilities. For this stage MLR is used, as explained in Section 4.2.

4. **Evaluation** of the solution, comparing the objectives and results. Observing and measuring how well the artifact supports a solution to the problem. At the end of this activity, the researchers can decide whether to iterate back to step 3 to try to improve the effectiveness of the artifact or to continue on to communication and leave further improvement to subsequent projects [60]. This stage uses the semi-structure interviews explained in Section 4.3.

5. **Communication** of the problem, the artifact, its utility, novelty and effectiveness, as well as the rigor of its design to researchers and other relevant audiences. In this case, communicating in papers and a dissertation.

4.2 Multivocal Literature Review

A MLR is a type of Systematic Literature Review (SLR), which aims to incorporate gray literature like blogs, videos, web-pages and white papers, which are constantly produced by Software Engineering (SE) practitioners outside academic forums, notwithstanding the published (peer-reviewed) writing like journal articles and conference papers. Therefore, MLR is important for the expansion of the research by including literature that normally wouldn't be taken due to its "gray" nature [3], as show in the Figure 4.2 on page 21.

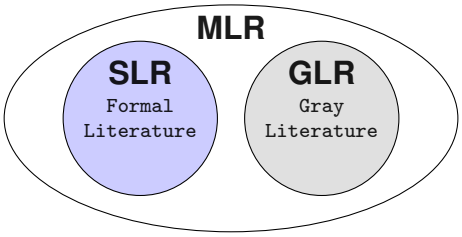

Figure 4.2: The relationship of SLR, GLR and MLR Studies [3]

While considering conducting a Literature Review from formal literature in the specific topic of DevOps, a few researchers already realized that "broadening" the scope and including Gray Literature (GL) would add value and benefits to the review study. Some examples of successful DevOps research, in the same area, using MLR already exist [37, 61, 62], thus corroborating the practical usefulness of this method for the proposed research, expanding the diversity of sources that are available in a variety of forms, reflecting different purposes and perspectives [63].

There are 3 objectives to be pursued with MLR for this research:

1. To map out the DevOps capabilities in detail from literature.

2. To gather the DevOps metrics from literature in a rigorous manner.

3. Construct a base for achieving the main objective, of finding the relationships between the main DevOps metrics and capabilities.

For the proposed research is identified the need to expand the research outside the boundaries of scientific knowledge and therefore MLR gives us that opportunity, while still maintaining a rigorous qualitative analysis procedure [63] for reviewing that literature. The separation of several types of literature is seen in Table 4.1 on page 22, where is listed 'White' and 'Gray' literature sources into 1st tier, with high credibility, and 2nd tier with moderate credibility. For DevOps, it is preferable to include 2nd tier, given that there is valuable expertise and knowledge on those sources. However, it is also necessary to exclude literature that corresponds to ideas, concepts and thoughts, like tweets, social networks or emails from the 3rd tier.

Table 4.1: Spectrum of the 'white', 'gray' and excluded literature (adapted) [3].

'White' literature	'Gray' literature	Excluded literature
Published journal papers	Preprints	Ideas
Conference proceedings	e-Prints	Concepts
Books	Technical reports	Thoughts
	Lectures	
	Data sets	
	Audio-Video (AV) media	
	Blogs	

Multiple guidelines exist in literature to conduct SLR studies in SE. However, several phases of MLR differ from those of traditional SLR. In particular, the process of researching and assessing the quality of the source. Therefore, SLR guidelines are only partially useful for conducting MLR studies as seen in 5.1 and Figure 6.1. This process shows the planning, conducting and reporting as proposed by Garousi et al. [3].

In following this process, it is expected that the gray literature will return substantial knowledge in certain areas of this DevOps research, but of course, the inclusion of such literature brings certain challenges as the evidence provided is often based on experience and opinion. For that reason, for this research process **systematic guidelines** are used for performing MLR in software engineering (SE) [64], to approach a structured search, similarly to SLR, collecting the materials by applying the inclusion and exclusion criteria in the search results obtained from well-known search engines like Google, Google Scholar and others.

1. The MLR **planning phase** consists of the following two phases.

    - Establishing the need for an MLR in a given topic.

- Defining the MLR's goal and raising its research questions.

2. Once the MLR is planned, we proceed to **conducting the review** in five phases.
    - **Search process** for formal or GL is typically done via means of using defined search strings.
    - **Source selection** normally includes determining the selection criteria and performing the selection process.
    - **Study quality assessment of sources** in order to determine the extent to which a source is valid and free of bias.
    - **Data extraction** design forms, procedures and logistics, with possibility of automated data extraction and synthesis.
    - **Data synthesis** with chosen qualitative and quantitative techniques.
3. Finally, **reporting the review** is the last phase.
    - The reporting phase of an MLR is similar to the SLR guidelines of Kitchenham and Charters [38], summarizing the extracted data from the selected literature and report findings.

4.3 Semi-structured Interviews

Here the capabilities, the metrics and their relations is evaluated, by conducting semi-structured interviews within different organizations. During these interviews, an interview protocol is used and participants are informed about the research goals, while asking for their consent to be interviewed.

Interviews with a semi-structured format are a common approach in development research. Often, it is a good way to learn about the reasons behind people's decisions and behaviors. It is also common for them to give vital information that the researcher had not before [65].

In this process, questions are asked regarding the organization and DevOps capabilities and metrics used, since the objective is to first have an insight into the organization's DevOps practices. Thereafter, the interview focuses on examining the artifact from the viewpoint of the listed capabilities and their relation to the respective metrics.

Semi-structured interviews are specially indicated in the following situations [66]:

- To ask probing, open-ended questions and to know the independent thoughts of each individual in a group.
- To conduct an evaluation on one-on-one interviews with key program managers, staff, and front-line service providers.

- To examine uncharted territory with unknown but potential momentous issues, and interviewers need maximum latitude to spot useful leads and pursue them.

These kinds of interviews are a common method for qualitative research [66]. The researchers and participants engage in a formal interview, using a developed “interview guide”. A list of questions and topics, usually in a particular order, that are to be covered during the conversation. However, the interviewer can also follow other topics in the conversation that can guide when he or she feels this is appropriate [67].

The DevOps capabilities confirmed in the literature, together with the related metrics to be found, is used to form an interview protocol in which the targeted evaluation model acts as a guide to elicit metrics from capabilities.

Semi-structured interviews allow interviewers to go into further detail on subjects that come up in talks with specific participants [68].

5

DevOps Capabilities Multivocal Literature Review

Contents

In the build phase of DSR, the process of construction of the artifact is undertaken. In the case for this research, two MLRs are used, as explained next.

5.1 Planning the MLR

This section corresponds to the first phase of the mentioned MLR process. It begins by explaining the motivation for this work, followed by the objectives and the corresponding research question that is intended to be answered throughout the research. Thereafter, a review protocol is presented.

The full process is shown in 5.1, which exposes the planning, conducting, and reporting as proposed by Garousi et al. [3].

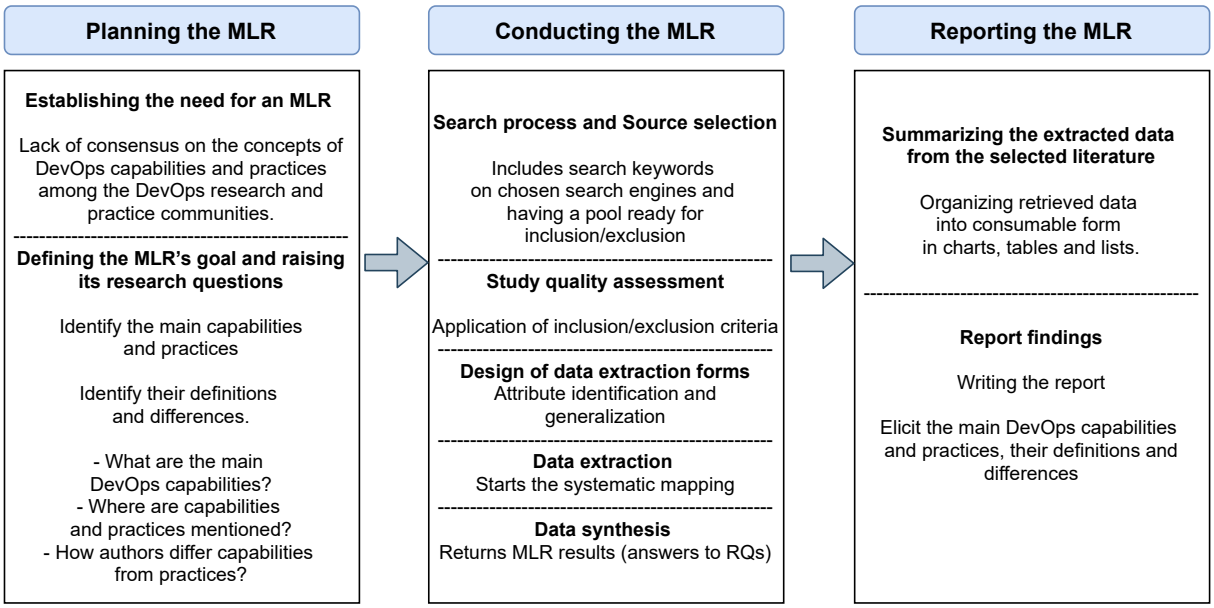

Figure 5.1: DevOps capabilities Multivocal Literature Review (MLR) Steps [3]

5.1.1 Motivation

In software development organizations, that want to implement DevOps internally, management needs to have relevant supporting information about this technological transformation, to assess the success and increase efficiency [11] of applying capabilities [8, 20]. However, the concept of capabilities and practices is still not well-defined within DevOps practitioners community.

As this topic has been further explored and analyzed from the industry side than from the scientific side, with leading technology companies regularly publishing reports [69], a Multivocal Literature Review expands the diversity of sources to identify the main capabilities and practices, their definitions and

differences, in order to map the capabilities mentioned by DevOps practitioners, researchers and how they distinguish between competencies and practices.

5.1.2 Research Questions

Based on the main purpose of this research, an investigation for scientific and 'gray' related work was done that addresses or explores the area of capabilities and practices, which can be translated into the previously defined RQ1, RQ2, RQ3.

5.1.3 Review Protocol

In order to find other studies related to this work, that may achieve answers to the proposed research questions, a search was conducted in April 2021 using various keywords. The search string used to perform the search to retrieve the maximum number of studies and the chosen datasets are listed in this section.

- **Search String**: `(devops AND (practices OR capabilities))` .
- **Datasets**: The search engines used were, Google search[1], Scopus[2], Web of Science[3], IEEE[4], ACM[5] and EBSCO[6].

The review protocol used the workflow shown in Figure 5.2. The first set of papers is obtained. In a first phase, after the search is complete and snowballing is done, inclusion and exclusion criteria is applied for refining the search results.

In order to facilitate searching and collecting high volumes of gray literature some code was developed as seen in Listing B.1 (Python code for consistent fetching of large number of Google search results) to parse the data into two CSV files [70]. This way we can ensure obtaining clean results that are not specific for the user, but general, this solving the problem of consistency in the returned results because Google search returns customized results that are tailored differently for different users based on their previous search history and preferences. Lastly, it facilitates the work of fetching the results into spreadsheet files that are easily consumable in the MLR process.

The inclusion and exclusion criteria for this MLR is shown in Table 5.1. After that step, the abstracts must be screened in order to evaluate the relevance they have to the research. Finally, the relevant papers are to obtain the final selection of studies to perform the review.

[1] `https://www.google.com`
[2] `https://www.scopus.com`
[3] `https://apps.webofknowledge.com`
[4] `https://ieeexplore.ieee.org`
[5] `https://dl.acm.org`
[6] `https://search.ebscohost.com`

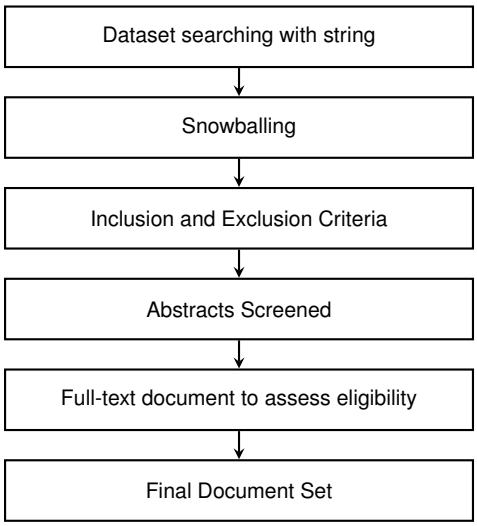

Figure 5.2: Review protocol performed in this research.

Table 5.1: Inclusion and exclusion criteria applied in this research.

Inclusion Criteria	Exclusion Criteria
Written in English	Unidentified author
Published in and after 2013	No publication date
Full-text accessible	Advertisement or Job Post
Mentions DevOps capabilities or practices	

5.2 Conducting the MLR

In this section, it is described how the review is conducted, which is the second phase of the SLR. At this moment, the search is performed using the search query over the selected databases and an analysis is carried on top of the extracted data.

5.2.1 Selection of Studies

For reference, the complete summary of the review process is shown in the diagram in Figure 5.3 with a visual representation of the applied MLR selection process. This reflects all the selection work done through the methodical process of MLR.

In the initial search step *filter 1* (All fields; All documents) was used together with the search string, both present in Table 5.2. This is shown in Table 5.2, as part of the MLR protocol to find the final set of article, which gives us a relation of the articles found in conjunction with the filters used.

The discrepancy from *filter 1* to *filter 2* is justified by the fact that initially the keywords could be found anywhere within the returned item and some search engines return more literature than just academic papers, like newspapers or reports. While in the case of Google search engine, this does not apply. Thus, the results remaining the same.

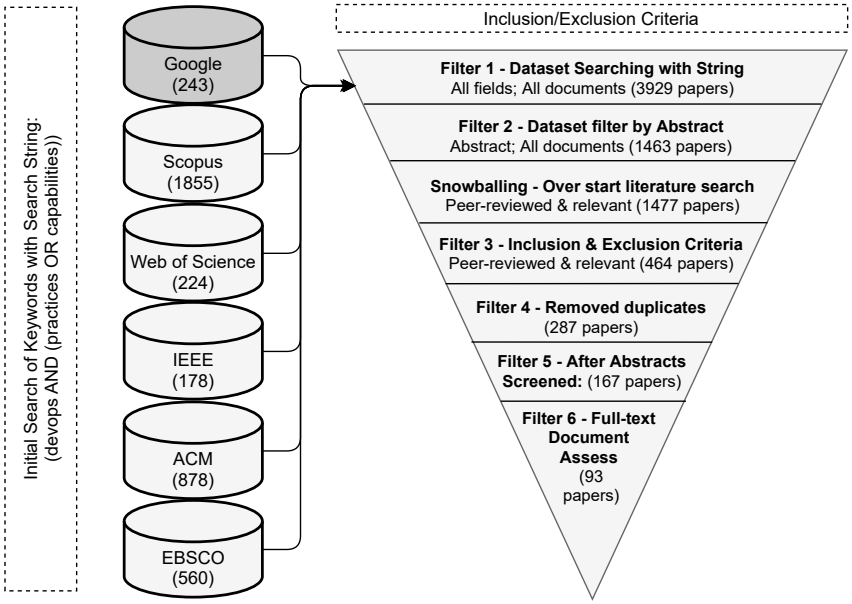

Figure 5.3: Followed Multivocal Literature Review process (adapted) [3].

On a second pass, *filter 2* (Abstracts; All documents) was used over the existing search results, therefore reducing the number of documents that have an abstract mentioning the keywords, narrowing down to a total of 1463 publications.

In the next phase a *snowballing* [71] is conducted leading to extra 14 relevant publications found, which increased the total amount of papers.

Applying inclusion & exclusion criteria *filter 3*, present in Table 5.1, 464 publications remain. This leads to *filter 4*, which is defined to remove the duplicates from the list of results in order to obtain the set of documents to have abstracts screened. For the cases belonging to gray literature, there is no abstract. Therefore, all text was skimmed, making it possible to better assert an inclusion or exclusion of that publication.

In the end, after all abstracts are screened, 93 publications remain for full-text document assess.

5.2.2 Data Extraction Analysis

After selecting the final set of publications, an analysis of the different components of the results is presented here, in a relationship of the final set of documents based on the source data. This analysis arises from the evaluation of the full text of the 93 publications eligible for extraction of any relevant information for this research. An overview is also given of which years and categories of publications were selected for full reading.

The relation of Gray and white literature final document set by database reflected in Figure 5.4 show that 75 results came from Google search with 80,65% of gray literature. For the cases of Scopus, Web

Table 5.2: Filters used in the MLR protocol.

Database	Search String	Filter 1	Filter 2	Sowballing	Filter 3	Filter 4	Filter 5	Filter 6
Google	(devops AND (practices OR capabilities))	243	243		89	89	77	75
Scopus	(devops AND (practices OR capabilities))	1855	342		157	42	40	2
Web Of Science	(devops AND (practices OR capabilities))	224	174		91	29	24	1
IEEE	(devops AND (practices OR capabilities))	178	146		67	67	14	8
ACM	(devops AND (practices OR capabilities))	878	92	+14	22	22	6	4
EBSCO	(devops AND (practices OR capabilities))	560	475		38	38	6	3
	Total	3929	1463	1477	464	287	167	93

Filter 1 = Query All fields, All documents
Filter 2 = Query Abstracts, All documents
Snowballing = Applied over starting literature search [3]
Filter 3 = Relevant (inclusion/exclusion criteria)
Filter 4 = Remove duplicates
Filter 5 = After Abstracts Screened
Filter 6 = Full-text Document Assess

Of Science, IEEE, ACM and EBSCO they all contributed with the total sum of 18 (19,35%) of relevant research documents.

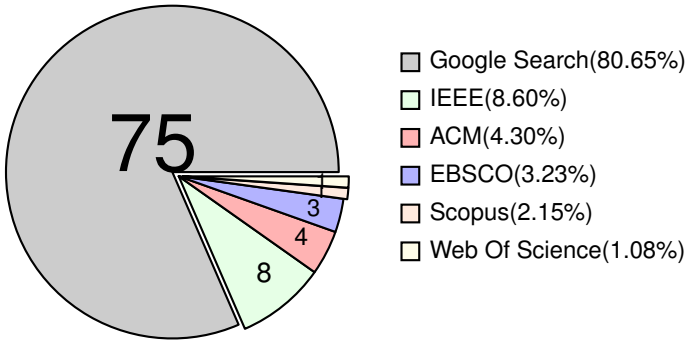

Figure 5.4: Distribution of the final set of documents per database.

Important to note the distribution and growth of the selected papers shown over the years in relation to the publication seen in Figure 5.5. This shows a growing interest in the last three years with an increase in volume of research work related to researching capabilities, confirming the potential interest and usefulness of this research might have in the area.

The first publication from this set was Puppet Labs' 2013 State of DevOps Report [72], that was followed by the next 2014 State of DevOps Report [73] and a webpage on "Six Core Capabilities of a DevOps Practice" from the New Stack website [74]. In 2015, 2016 and 2017, there is a continued growth in webpages and conference papers in 2017.

The interest in gray literature has risen significantly in 2018, as evidenced by the enormous increase

of webpages in that year, indicating that practitioner publications have evolved far quicker than scientific research. Despite a brief reduction in publications in 2019, practitioners' publications still continued to grow in 2020 as we can see in Figure 5.5.

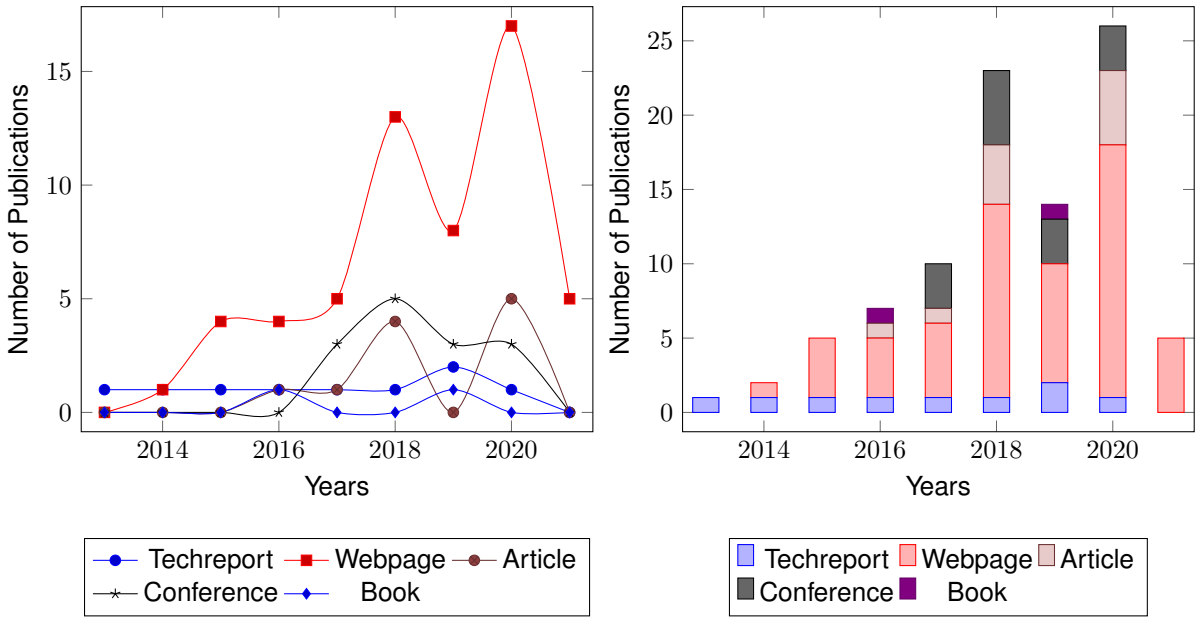

Figure 5.5: Distribution of publications per type over the years.

It becomes important to note that the values for 2021 are lower because the search that created the database for this paper occurred in March 2021, and therefore it is an incomplete year. Also, the drop in publications in 2019 can be justified partially due to the global pandemic of COVID-19, where much of the industry took a big hit. This change deeply impacted typical working routines, affecting both well-being and productivity [75].

In order to get an overview of how the various capabilities have grown in the literature over the years, Figure 5.6 can be observed.

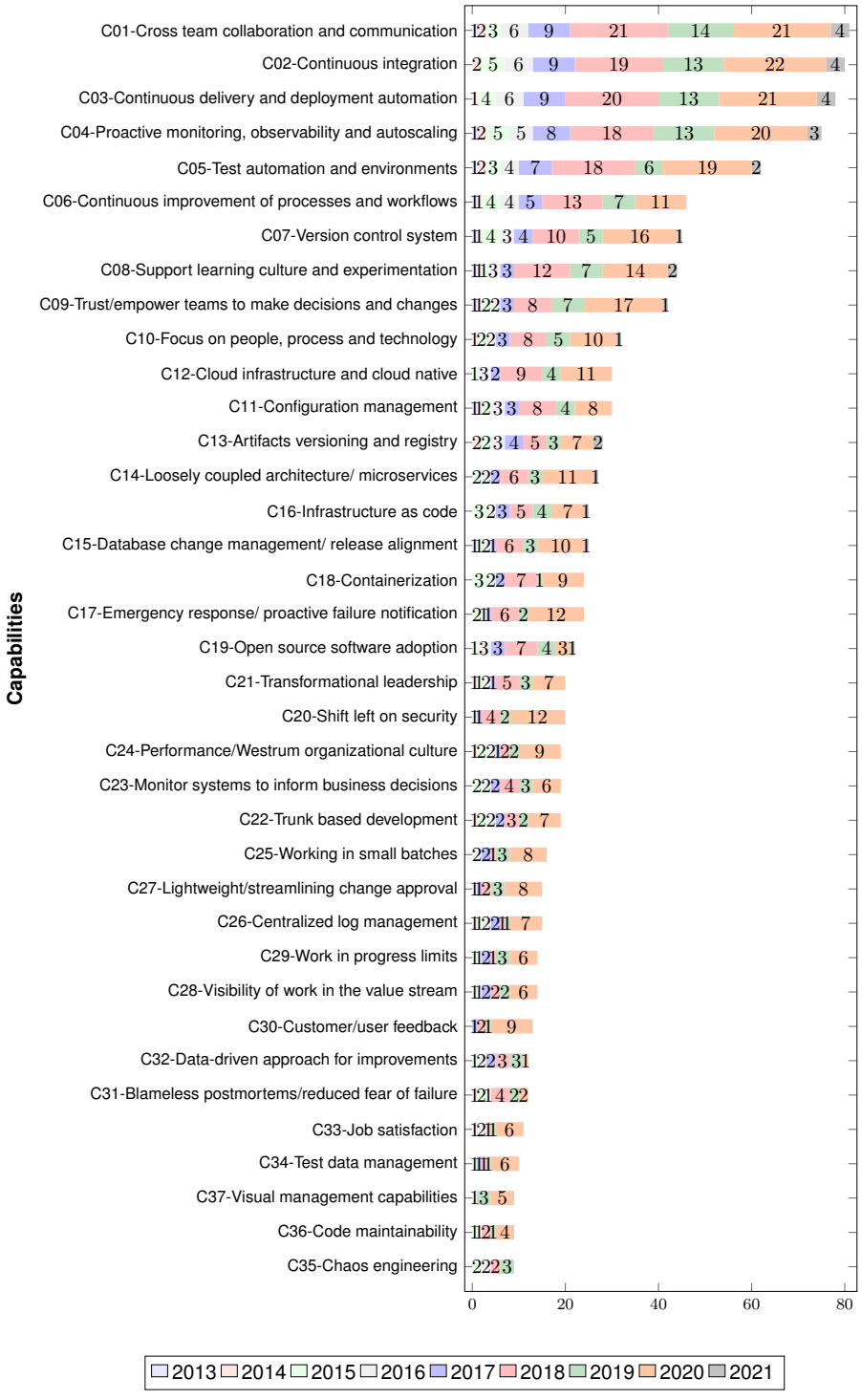

Figure 5.6: List of capabilities identified by number of publications over the years.

It is noticed a big leap in 2020, namely in cross team collaboration, continuous integration, continuous delivery, monitoring and test automation with around 20 mentions each.

Regarding collaboration, as mentioned by Kim et al. [76], this denotes an antithesis to a culture of fear, and instead organizations embracing DevOps strive to have a high-trust, collaborative culture, where people are rewarded for taking risks.

A trusting organizational culture that encourages information flow predicts software delivery performance and technical organizational success [8]. The concept that a healthy culture with increased information flow predicts exceptional achievements is not new; it is based on research by sociologist Dr. Ron Westrum [77].

Also, continuous integration (CI) and continuous delivery (CD) seen in the same Figure 5.6 are the corner stone of software delivery, denoting the huge importance of a CI/CD pipeline [4, 48, 78–83]. A pipeline targets being a repeatable system composed of phases that code must pass through before it can be deployed to production: First, programmers have to write the code, then, the team compiles the code into a build to check for errors, next teams run tests to ensure that the new code will behave as intended in the production environment and finally once the new code has passed the testing phase, it is deployed to the end user [84, 85]. Therefore, continuous integration is the capability of multiple developers to commit and merge their code [86,87]. Continuous delivery entails deploying code updates to production as frequently as feasible.

Monitoring and test automation are within the top five most mentioned capabilities. Monitoring the service the DevOps pipeline, the infrastructure or any other component of the SDLC [7, 83, 88, 89] is as important as developing the software itself. Test automation validates code correctness in production-like environments, in order to have that code deployed into production quickly, safely and securely.

Finally, from the relation of these and the other capabilities over the years, it is shown that practitioners are championing these principles, practices, and tools to minimize waste [83, 90], enabling faster feedback cycles [7, 91], exposing invisible technical debt [76], improving value in delivery, maintenance and operational functions [83].

5.3 Reporting the MLR

At this step of the MLR, the DevOps capabilities are reported, and all three research questions are assessed in light of the publications obtained using the research protocol.

5.3.1 RQ1 - What are the main DevOps capabilities or practices?

In this section, the MLR provides an answer to the first research question, by revealing a list of 37 capabilities. As seen in Table 5.4 this list is based on the literature review of the 93 publications, also

considering the fact that some practitioners mention them as practices and others as capabilities as shown in Figure 5.7. The list includes their name, the number of publications in which they have been mentioned, and, most importantly, their definitions.

C01. *Cross team collaboration and communication*, mentioned in 81 publications. In order to enable cross-functional collaboration between application teams, operations and security teams [4, 48, 82, 84, 90, 92–98] the organization has to identify the stakeholders, including customers [99], of every project so that they join, have insights about various project phases and processes, and start making valuable contributions.

C02. *Continuous integration (CI)*, mentioned in 80 publications. Continuous integration takes tasks like testing and building, and automates them [7, 100], driving teams to produce high quality software, to reduce the cost of ongoing software development and maintenance [6, 88, 91, 93, 101–104], and to increase the productivity of the teams. [4, 91]. The CI process creates canonical builds and packages that are ultimately deployed and released [5].

C03. *Continuous delivery and deployment automation*, mentioned in 78 publications. While continuous delivery automates the entire software release process with a manual step, continuous deployment automates that step, deploying smaller changes [105] to production as soon as they are released from acceptance testing [83, 106], without manual intervention [5], releasing faster and more frequent, reducing the risk of production deployments and providing faster feedback to the teams [8, 90].

C04. *Proactive monitoring, observability and autoscaling*, mentioned in 74 publications. It is critical to monitor the infrastructure [44, 76, 83, 105, 107], whether it is in the cloud or in a local data center. Combining proactive monitoring with autoscaling can automatically solve capacity issues [73, 87, 96, 108] and reduces the need to scale the system manually.

C05. *Test automation and environments*, mentioned in 62 publications. Getting quick feedback on the impact of changes across the SDLC is fundamental to integrate quality into software [72, 76]. It is important to have automated and correctly provisioned test environments along the pipeline [4, 87, 109, 110], reducing long lead times [100, 106, 111–113].

C06. *Continuous improvement of processes and workflows*, mentioned in 46 publications. Continuous improvement is enabled through a combination of continuous integration, deployment, testing, workflows and monitoring [48], like, implementing branch-naming consistency, where all work originates from the same source while developing on a branch referencing a ticket [114], or applying consistent patterns across multiple applications [113].

C07. *Version control system*, mentioned in 45 publications. Version control and automation are tightly intertwined [6, 7, 86, 89] enabling efficiency and productivity [72, 115, 116]. Version control extends versioning to all production artifacts [76], such as application code, configurations, system settings, and scripts for automating build and environment setup [5, 112].

C08. *Support learning culture and experimentation*, mentioned in 44 publications. Organizations that develop a learning culture [117–119] and comprehend its impact on organizational performance encourage engineers to have the ability to work alone and experiment [6, 109] to test business concepts and new ideas, to write and update requirements during development [111, 113].

C09. *Trust/empower teams to make decisions and changes*, mentioned in 42 publications. Trust is essential in every relationship, but it is especially critical for DevOps [114] to improve software delivery performance and job satisfaction empowering them with the ability to make educated decisions about the tools and technologies they employ [44, 73, 98, 103, 111, 117]. This helps to create greater outcomes [4, 87, 120].

C10. *Focus on people, process and technology*, mentioned in 32 publications. People, process and technology are the three pillars of a software development project. There must be a feeling of community, sharing a common goal, and contributing to the common cause [103]. Improving the culture is an ongoing journey [121, 122]. DevOps unifies people, processes, and technology: when all three are aligned toward the same business goals, innovation can be implemented more quickly [80, 113, 123, 124].

C11. *Configuration management*, mentioned in 30 publications. Is practiced in one form or another as part of any software engineering project. A Software Configuration Management (SCM) is a system for managing the evolution of software products [125, 126], automating the configuration, monitoring, managing, and maintenance of all entities of infrastructure and systems like servers, applications, storage, networks, and all managed services [88].

C12. *Cloud infrastructure and cloud native*, mentioned in 30 publications. The US National Institute of Standards and Technologies (NIST) defines five essential characteristics of cloud computing [127]: On demand self-service, broad network access, resource pooling, rapid elasticity and measured service [87, 128]. Each service may be deployed individually [129], with flexibility, tool sets, and scalability for applications. Serverless architectures on clouds can dramatically reduce DevOps effort [86]. In a pipeline—for example for worker nodes, deploying artifacts to test or even production environments [78].

C13. *Artifacts versioning and registry*, mentioned in 28 publications. Enable organizations to centrally store artifacts and build dependencies as part of the software delivery process [7, 98, 130]. It is important to version these artifacts in a repository manager [131–133], either they are promoted containers along the pipeline, bundles, charts, packages or any other kind to make the changes visible, reliable and repeatable [15, 133] for all production artifacts [16, 97, 119].

C14. *Loosely coupled architecture/ microservices*, mentioned in 27 publications. Microservices are an architecture design for building a single application with smaller services that run independently, usually communicating via API calls [134]. Improves agility and helps organizations to easily grow their product at a cheaper cost and in a shorter time [94]. Each service may be deployed separately and decentralized. Produced and delivered using automated tools and automated procedures [90].

C15. *Database change management/ release alignment*, mentioned in 25 publications. Database change management [7, 135] and release alignment change management processes [102, 136, 137], when well implemented, help developers and IT professionals to easily manage database updates, system configurations, deploy new code quickly and fix incidents faster.

C16. *Infrastructure as code*, mentioned in 25 publications. With infrastructure as code [105, 120, 138–140], it is possible to express procedures in code [141] rather than setup infrastructure or software manually. Using technologies like Chef, Puppet, Ansible, or Salt [142, 143]. That way it is possible to use version control to keep track of all infrastructure modifications in a repeatable and more efficient manner [144].

C17. *Emergency response/ proactive failure notification*, mentioned in 24 publications. Proactive failure notification [7] focus on actionable notifications based on the values being monitored and that have known failure thresholds, instead of a reactive system to alert when it has already failed [4, 95], improving emergency response efficiency [145] and reducing risk of customer impact [44].

C18. *Containerization*, mentioned in 24 publications. Containers are efficient for app development and hosting [86]. They allow DevOps, developers, and system administrators to swiftly, securely, and effectively test, build, deploy, and manage applications [83]. This capability has become a new standard in DevOps pipelines, clusters, and applications [115, 134]. C19. *Open source software adoption*, mentioned in 22 publications. Open source adoption correlates with DevOps success [81, 87, 100, 146], and the knowledge of open source solutions for testing and deployment is a must for a DevOps engineer [86]. This model is well represented in the DevOps tool set [116, 147] with impact in early DevOps emergence [117]. Organizations assemble and contribute open source parts, which has become a reliant software supply chain [76, 111].

C20. *Shift left on security*, mentioned in 20 publications. Integrating security into the design and testing phases of the software development process is key to driving IT performance. Including security reviews of applications, including the infosec team [5, 6, 109]. Shift left on security is related to DevSecOps [98] concept and emphasizes automating as much as possible security policies in order to accelerate processes, decrease human error and aiding in quality improvement [148] and audits [113, 149].

C21. *Transformational leadership*, mentioned in 20 publications. Is a style in which leaders inspire and encourage teams to attain better levels of performance [90, 97]. Transformational leaders focus on the growth and performance of their followers and organization [76, 112]. Effective leaders [113] impact software delivery performance by pushing the use of technical and product management capabilities [150].

C22. *Trunk based development*, mentioned in 19 publications. In trunk based development, each developer works in small batches, merges that work into trunk at least once (and potentially several times) a day [15, 137], consistent with commonly accepted continuous integration practices [27, 97, 151].

C23. *Monitor systems to inform business decisions*, mentioned in 19 publications. Inform business decisions using visual dashboards [4, 89, 107, 113, 152] allows organization to track configuration changes made to servers along with databases and deployments [123] that have taken place, along with various metrics, logs, and graphs [153, 154] to give a holistic view of changes happening in the system.

C24. *Performance/Westrum organizational culture*, mentioned in 19 publications. Ron Westrum developed a typology of organizational cultures that includes three types of organizations [77]. *Pathological organizations* are characterized by low cooperation across groups and a culture of blame. *Bureaucratic cultures* are preoccupied with rules and positions, and responsibilities are compartmentalized by department. *Generative organizations* are performance oriented [109], with good information flow, high cooperation and trust, bridging between teams [76, 155].

C25. *Working in small batches*, mentioned in 16 publications. Working in small batches with a lightweight approval process helps ensure work can get through the system quickly [107] with shorter lead times [7]. It enables fast flow through the development pipeline, fixing errors as they are discovered vs. at the end [156] also allowing to deliver of MVPs, features, and bug fixes sooner, which also helps enable the customer feedback loop above [98].

C26. *Centralized log management*, mentioned in 15 publications. Centralized logs for applications with multiple servers facilitate debugging [87, 157] when sent to a common service that enables easy centralization, rotation, and deletion [76, 84, 158].

C27. *Lightweight/streamlining change approval*, mentioned in 15 publications. Replace heavyweight change-approval systems with peer review [150]. Lead times and release frequency improve considerably [97] with negligible impact on system stability [73, 150].

C28. *Visibility of work in the value stream*, mentioned in 14 publications. Understand and visualize the flow of work [97, 100] from idea to customer outcome in order to drive higher performance. Make the value flow visible for everyone to understand where their piece fits into the whole flow [107, 150] from the business all the way through to customers [6, 155].

C29. *Work in progress limits* or Work in process limits, mentioned in 14 publications. Prioritize work, limit the number of things that people are working on [10, 112], and focus on getting a few high-priority tasks done [150]. Work in Progress (WIP) limits [107] are identified and enforced. Flow is defined [98]. Overload is limited [90].

C30. *Customer/user feedback*, mentioned in 13 publications. Drive better organizational outcomes by gathering customer feedback [155, 159] and incorporating it into product and feature design [107].

C31. *Blameless postmortems/reduced fear of failure*, mentioned in 12 publications. By removing blame, fear is reduced, and by reducing fear, teams are empowered to surface and solve problems more efficiently. Mistakes occur. Holding blameless postmortems [16, 73, 76, 87] is an effective technique of learning from mistakes [150, 160].

C32. *Data-driven approach for improvements*, mentioned in 12 publications. Analyzing factual data [140] can help an organization achieve performance. Sharing application graphs [157, 161], usage patterns with team members to get everyone aligned. Include scalability, testing, and deployment to simplify the entire process [84].

C33. *Job satisfaction*, mentioned in 11 publications. Job satisfaction is the top predictor of organizational performance [73, 92]. DevOps adoption also help avert burnout [112], a common reason why technical people leave jobs [162].

C34. *Test data management*, mentioned in 10 publications. Managing test data can be challenging [107]. Define the right strategies for managing test data effectively along with approaches to provide fast, secure data access for testing [148] like adequate data to run a test suite, acquiring data on demand, conditioning and limiting the amount of test data needed in the pipeline [5].

C35. *Chaos engineering*, mentioned in 9 publications. Contemporary and distributed software programs must be capable of dealing with unexpectedly tumultuous environments [76,78,94,107,124]. As a result, such systems must be built from the start to withstand unanticipated problems and shortcomings in production contexts [163].

C36. *Code maintainability*, mentioned in 9 publications. Code maintainability [7, 111] is essential for making it simple for developers to identify, reuse, and alter code, as well as keep dependencies up to date [164].

C37. *Visual management capabilities*, mentioned in 9 publications. Improves a company's capacity to assess progress toward goals, manage change and improvements [18]. Visual management of work and pulling it through the system [17] is a critical part of the First Way of DevOps [76].

5.3.2 RQ2 - Where are capabilities and practices mentioned?

Based on the extended research enabled by this MLR, it is seen that capabilities have been mentioned interchangeably as practices in 66 publications Table 5.3 and eight publications even distinguish practices from capabilities [4, 82, 95, 102, 112, 122, 123, 165]. The two terms are actually described across several types of white and gray literature as seen in Figure 5.7.

It can be observed in the figure, that the webpages, overwhelmingly created by practitioners, have the most mentions as practices, but also include a substantial number of mentions as capabilities. The same happens in Techreport, Conference and Book, which are closer to the gray literature.

On the other hand, the scientific articles make more mentions as capabilities, which becomes a very interesting finding of this research, as it reveals that practitioners are more focused on DevOps practices, while the scientific community tries to organize capabilities in a way that abstracts more generic concepts applicable to building skills and enablers. Nevertheless, the concepts are the same, only at different stages of the process.

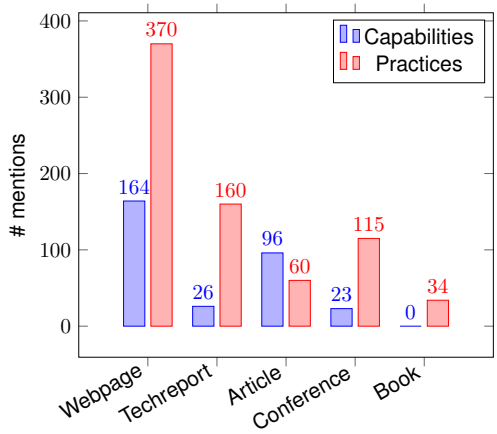

Figure 5.7: Number of publications mentioning capabilities and practices among sources.

A different example of this same interchangeability can be observed on a seminal book, “Continuous Delivery” [15] from 2010, that starts on mentioning the term capabilities. Despite not explicitly mentioning the word “DevOps” it describes, however, in detail the deployment pipeline pattern, which is usually central to DevOps capabilities. In page 109 of the book, the capability of deployment and production release is described in detail, explaining how the process is automated, with speed, repeatability and reliability in mind. Jezz Humble mentions that when the “capability” of automating the process as normal events is available, releases are essentially without risk. Some other books also talk about capabilities and are frequently cited in gray literature by researchers and practitioners [6,85,87,98,102,117,120,157] like "The Phoenix Project" [17], "Accelerate: The Science of Lean Software and DevOps" [5], "Lean enterprise : adopting continuous delivery, DevOps, and lean startup at scale" [150], and “The DevOps Handbook” [76].

In Table 5.3 it is mentioned that one publication indicates a DevOps practice definition [123] to be a subset implementation of a capability.

Table 5.3: Six publication properties identified from the MLR.

Property	Publications	Total
Interchangeably mentions capabilities and practices	[8, 16, 44, 48, 69, 72, 73, 76, 78–81, 83, 84, 86–90, 92–94, 97, 99, 100, 103, 104, 106, 110, 113–119, 121, 124, 128, 141, 152, 155, 160, 166–188]	66
Mentions capabilities directly	[4, 6, 8, 82, 85, 95, 98, 100, 102, 104, 105, 109, 111, 112, 122, 123, 135, 146, 165]	19
Presents different or reorganized capabilities compared to Senapathi et al. [4]	[6, 8, 95, 98, 102, 104, 105, 109, 111, 122, 123, 135, 146, 165]	14
Distinguishes practices from capabilities	[4, 82, 95, 102, 112, 122, 123, 165]	8
Indicates a definition for capability	[4, 95, 97, 105, 112, 123]	6
Indicates a definition for practice	[123]	1

Six publications indicate various capabilities definitions like a higher level categorization of practices [105], are important to enhance software company's profitability, productivity and market share [112], capabilities are the core DevOps aspect which comprises capabilities such as "continuous planning, collaborative and continuous deployment, continuous integration and testing, continuous release and deployment, continuous infrastructure monitoring and optimization, continuous user behavior monitoring and feedback and service failure recovery without delays" [95]. The previously mentioned research done by Senapathi et al. [4] and 2017 State of DevOps report [97] defines them as a combination set that can change over time, which includes categories of capabilities like "continuous delivery as the combination of the capabilities" to be "deployment automation and automated testing, continuous integration and trunk-based development, and version control for all production artifacts". These set of capabilities have been changing over the years.

5.3.3 RQ3 - How authors distinguish capabilities from practices?

The differentiation of why some authors mention capabilities and other practices is discussed here and defined in Section 2.2. It is largely observed that the word "capability" is used when the observation is external or at a high-level overview. It is then a matter of perspective. When talking about a capability, we see a third-party assessment of something that is being looked at from the outside, while observing a group to see what they are capable of doing.

Whereas, a practice is seen from the standpoint of the internal team or group, realizing "I am doing these things". That ability converted to an action is then mentioned with the term "practice". Therefore, authors will speak about capabilities from an evaluation standpoint, and practices from a hands-on approach perspective. The capability definition points to an organization's "ability" to perform or achieve a certain process, whereas a practice is referred to more at the level of DevOps practitioners and thus more observed in the gray literature publications as discussed in Section 5.3.4.

Clear examples of this more formal research concept were presented earlier by Smeds et al. [20], Senapathi et al. [4] and more recently in the book Accelerate [5], in DORA [7, 8, 111] and in several journal articles or proceedings [44, 82, 90, 95, 95, 105, 112, 123].

A capability is also mentioned as a "construct" [5, 146] and that there are "capabilities we are building" [76] in order to enable the organization for a certain practice. Organizations should target developing capabilities and habits in their people [189] as an enabler for continuous improvement and functional skills.

The number of publications mentioning capabilities or practices is organized in Table 5.4.

Table 5.4: Number of publications mentioning capabilities or practices.

Capability	Mentions as practice	#	Mentions as capability	#	Total
Cross team collaboration and communication	[16, 44, 48, 72–74, 76, 79–84, 86–88, 90–94, 96, 97, 99–101, 103, 104, 106, 110, 113, 115–121, 124, 128, 134, 141, 144, 147, 152, 155, 160, 166, 168–170, 172, 174, 176–184, 186, 190, 191]	66	[4, 6, 69, 85, 95, 98, 102, 105, 109, 111, 112, 122, 123, 135, 146]	15	81
Continuous integration	[16, 44, 48, 73, 74, 76, 78, 80–84, 86–94, 96, 97, 99, 100, 103, 104, 106, 107, 113–121, 124, 128, 141, 144, 147, 152, 155, 160, 169, 170, 173–175, 177, 179–188, 190]	63	[4, 6, 8, 85, 95, 98, 102, 104, 105, 109, 111, 112, 122, 123, 135, 146, 165]	17	80
Continuous delivery and deployment automation	[16, 44, 48, 73, 76, 78, 80, 82–84, 86–90, 92–94, 96, 97, 99–101, 103, 104, 106, 107, 110, 113–119, 121, 124, 128, 134, 141, 144, 152, 155, 160, 169, 170, 174, 175, 177, 178, 180–186, 188, 190, 191]	60	[4, 6, 8, 69, 85, 95, 98, 102, 104, 105, 109, 111, 112, 122, 123, 135, 146, 165]	18	78
Proactive monitoring, observability and autoscaling	[16, 44, 48, 72–74, 76, 78, 80, 82–84, 86–92, 96, 97, 101, 103, 106, 107, 113, 115–117, 119–121, 124, 128, 141, 144, 147, 152, 157, 160, 169–172, 174, 176–178, 180, 181, 183, 184, 186–188, 190, 191]	57	[4, 6, 8, 85, 95, 98, 102, 104, 105, 109, 111, 112, 122, 123, 135, 146, 165]	17	74
Test automation and environments	[16, 48, 72–74, 76, 78, 79, 81–83, 86, 87, 89–94, 96, 97, 100, 103, 104, 106, 110, 113–117, 124, 141, 144, 155, 160, 168, 169, 173–179, 182–185]	49	[4, 6, 8, 69, 102, 104, 105, 109, 112, 122, 123, 146, 165]	13	62
Continuous improvement of processes and workflows	[16, 44, 48, 72, 73, 76, 78, 84, 87, 88, 90, 92–94, 97, 99, 100, 103, 110, 113–115, 117, 121, 124, 141, 155, 166, 174, 176, 181, 182, 184, 185, 187]	35	[4, 69, 85, 102, 105, 111, 112, 122, 123, 135, 146]	11	46
Version control system	[16, 44, 48, 72, 73, 76, 78, 79, 82–84, 87–90, 92, 94, 97, 100, 103, 106, 113, 115, 117, 119, 141, 155, 172, 177, 179, 183, 186, 187]	33	[6, 8, 69, 98, 102, 104, 109, 111, 112, 123, 135, 146]	12	45
Support learning culture and experimentation	[16, 44, 72, 73, 76, 80, 81, 83, 84, 90, 92–94, 97, 100, 103, 106, 113, 114, 117–119, 124, 155, 166, 168, 176, 177, 181–183, 186]	32	[4, 6, 8, 98, 102, 104, 109, 111, 112, 123, 135, 165]	12	44
Trust/empower teams to make decisions and changes	[16, 44, 72, 73, 76, 80–84, 88–90, 92–94, 97, 100, 103, 113, 114, 117, 119, 121, 155, 166, 174, 176, 181, 183, 187]	31	[6, 8, 98, 102, 104, 105, 109, 111, 112, 135, 146]	11	42
Focus on people, process and technology	[16, 73, 76, 79, 84, 85, 88, 90, 92, 94, 97, 100, 103, 113, 117, 119, 121, 124, 152, 166, 176, 181, 182, 186, 187]	25	[8, 104, 105, 112, 122, 123, 135]	7	32
Configuration management	[16, 44, 72, 73, 76, 80, 81, 84, 88–90, 92, 97, 100, 103, 113, 155, 168, 170, 174, 176, 178, 181, 182, 184, 187]	26	[102, 112, 123, 146]	4	30

Continued on next column

Table 5.4 – *Continued from previous column*

Capability	Mentions as practice	#	Mentions as capability	#	Total
Cloud infrastructure and cloud native	[44, 48, 76, 78, 81, 82, 84, 87, 94, 97, 99, 100, 103, 113–115, 117, 174, 181, 184, 185]	21	[4, 8, 69, 102, 104, 111, 112, 122, 135]	9	30
Artifacts versioning and registry	[16, 73, 74, 76, 79, 80, 82, 83, 96, 97, 100, 107, 110, 113, 119, 120, 144, 155, 169, 179, 181, 183, 186, 187]	24	[69, 98, 112, 146]	4	28
Loosely coupled architecture/ microservices	[16, 44, 76, 79, 82, 87, 88, 90, 92, 94, 97, 100, 113, 114, 117, 119, 181, 185]	18	[4, 6, 8, 98, 104, 109, 111, 112, 135]	9	27
Database change management/ release alignment	[72, 76, 87, 89, 100, 106, 110, 113, 114, 117, 119, 166, 170, 174, 186]	15	[8, 69, 85, 102, 104, 105, 111, 123, 135, 146]	10	25
Infrastructure as code	[16, 76, 78, 79, 82, 87, 88, 90, 93, 94, 97, 100, 106, 113, 114, 119, 124, 141, 155, 182]	20	[98, 105, 111, 123, 146]	5	25
Emergency response/ proactive failure notification	[76,82,87,92,94,100,103,106,110,113,121,174,181]	13	[6, 8, 95, 104, 109, 111, 112, 123, 135, 165, 187]	11	24
Containerization	[16, 76, 78, 81, 82, 87, 92, 99, 100, 113–115, 117, 141, 181, 183, 185]	17	[8, 69, 104, 111, 112, 122, 135]	7	24
Open source software adoption	[48, 76, 81, 83, 84, 86, 87, 94, 97, 100, 104, 113, 115–117, 120, 134, 146, 147, 181, 185]	21	[111]	1	22
Shift left on security	[44, 76, 81, 82, 84, 97, 100, 103, 113, 181]	10	[4, 6, 8, 98, 104, 109, 112, 123, 135, 146]	10	20
Transformational leadership	[16, 73, 76, 84, 92, 94, 97, 100, 113, 155]	10	[4, 6, 8, 98, 104, 109, 111, 112, 135, 176]	10	20
Trunk based development	[16, 44, 73, 76, 87, 88, 97, 155, 179]	9	[6, 8, 98, 102, 104, 109, 111, 112, 135, 146]	10	19
Monitor systems to inform business decisions	[44, 76, 87, 89, 90, 103, 110, 113, 152, 174, 181, 182, 184, 187]	14	[4, 8, 104, 123, 146]	5	19
Performance/Westrum organizational culture	[16, 44, 73, 76, 84, 87, 94, 97, 103, 113, 184]	11	[6, 8, 98, 104, 109, 112, 123, 135]	8	19
Working in small batches	[76, 90, 93, 97, 103, 107, 117, 155]	8	[6, 8, 98, 104, 109, 111, 112, 135]	8	16
Centralized log management	[73, 76, 82, 87, 100, 113, 115, 121, 152, 157, 183, 184]	12	[8, 104, 112]	3	15
Lightweight/streamlining change approval	[76, 84, 92, 97, 100, 103, 113]	7	[6, 8, 98, 104, 109, 111, 112, 135]	8	15

Continued on next column

Table 5.4 – *Continued from previous column*

Capability	Mentions as practice	#	Mentions as capability	#	Total
Visibility of work in the value stream	[76, 87, 92, 100, 107, 113, 182]	7	[6, 8, 98, 104, 109, 111, 112]	7	14
Work in progress limits	[16, 76, 90, 97, 107, 117]	6	[6, 8, 98, 104, 109, 111, 112, 135]	8	14
Customer/user feedback	[44, 85, 93, 97, 99]	5	[6, 8, 98, 104, 109, 112, 123, 135]	8	13
Blameless postmortems/reduced fear of failure	[16, 73, 76, 87, 92, 93, 113, 160]	8	[102, 111, 135, 146]	4	12
Data-driven approach for improvements	[16, 48, 88, 90, 97, 100, 113, 117, 155, 178, 184]	11	[111]	1	12
Job satisfaction	[73, 76, 92, 100, 155]	5	[6, 8, 98, 104, 109, 112]	6	11
Test data management	[87, 183]	2	[6, 8, 98, 104, 109, 112, 122, 135]	8	10
Chaos engineering	[76, 78, 80, 87, 94, 124, 185, 187]	8	[111]	1	9
Code maintainability	[44, 76, 87, 113, 117]	5	[8, 104, 111, 112]	4	9
Visual management capabilities	[76, 84]	2	[6, 8, 98, 104, 109, 111, 112]	7	9

Concluded

It is now clear that, despite existing still some confusion while interchanging these two words, as seen in Figure 5.7 and Table 5.4, we are, in reality, talking about the same fundamental concepts.

The usage of capabilities or practices is not consensual, and because this study is not about achieving that consensus, it is chosen to use the term **capabilities**; nevertheless, others might also use practices. A consensus between the two should be further researched in the future.

In Table 5.4 we can see the complete set of publications mentioning as capabilities or practices. Notably, there are some that are mentioned more as a capability like "Trunk based development", "Lightweight/streamlining change approval", "Work in progress limits", "Customer/user feedback", "Test data management" and "Visual management capabilities".

5.3.4 DevOps Capabilities Synthesis

A MLR was conducted, in order to identify and sort an updated list of capabilities or practices, their definitions in Section 5.3.1, in which publications they are mentioned in Section 5.3.2 and their differences in Section 5.3.3. In extent, it is seen that the capabilities are, in fact, dynamic and have been changing over the years.

The list of the most studied and approached capabilities was collected as it was proposed. Of which, it is highlighted *cross team collaboration and communication*. In this collaboration, developers have access to a self-service platform that provides a foundation for automation, standardization, and team autonomy to enable the other three most mentioned capabilities: *continuous integration*; *continuous delivery and deployment automation*; *proactive monitoring, observability and autoscaling*.

Table 5.5: Definition of DevOps capability.

A **DevOps capability** is here defined as the ability to do something [192] or by the quality, or state of being capable [193]. On the other hand, technological capabilities are defined as the information and skills - technical, managerial and institutional - that enable productive enterprises [90, 150, 189] to utilize equipment and technology efficiently [194]. It consists of the combined skills accumulated and developed by its members over time.

A **DevOps practice**, differently from the definition of capability given in Table 5.5, is defined as the use of the mentioned capability by an individual or a group such as the engineering team. As a result, authors will discuss capabilities from the perspective of evaluation, but refer to practices from the perspective of a hands-on approach. Therefore, this research will use the term "capability" from now on, since that is applicable to evaluating DevOps adoption.

In 2015, Smeds et al. [20], proposed a set of capabilities that were later referenced and augmented in another research conducted by Senapathi et al. [4] in 2018. In this paper, the author discusses and establishes what the DevOps technology enablers and engineering capabilities are, here presented in Table 5.6.

Interestingly, from all the final publications gathered in this research only two of the scientific literature papers refer to the work of Senapathi et al. [4]. The mentioned papers are “A maturity model for DevOps” [123] and “Managing quality assurance challenges of DevOps through analytics” [48], therefore evidencing the low impact of the first definition proposals had in the long term.

These capabilities bring several types of advantages which can be grouped into technical, cultural and business benefits, Kim et al. [76]. The technical outcomes are mostly in the practices of Continuous Integration, Delivery and Reliability [195]. The cultural benefits reside in the improvement of communication and the creation of stronger feedback and collaboration between different teams [4, 44, 73, 76, 81–83, 95, 96, 100–102, 114, 116, 121, 177, 186, 191] and, in result, to improve employee motivation, which usually contributes to achieving the organization goals.

Table 5.6: List of DevOps capabilities proposed by Senapathi et al. [4].

1. Collaborative and continuous development
2. Continuous integration and testing
3. Continuous release and deployment
4. Continuous infrastructure monitoring and optimization
5. Continuous user behavior monitoring and feedback
6. Service failure recovery without delay
7. Continuous Measurement

Finally, the business benefits are on the customer side and the faster delivery of value, and at the same time ensuring greater business stability and increased innovation [150]. Evolving from the various State of DevOps reports research done over the years [69, 112], 24 Key capabilities have been published in the book Accelerate [5] in 2018, where the five categories were initially proposed can be seen in Table 5.7.

Table 5.7: List of capabilities in five categories proposed by Accelerate [5, 6].

Continuous Delivery	**Architecture**	**Product and Process**	**Lean Management and Monitoring**	**Culture**
Version control	Loosely coupled architecture	Customer feedback	Change approval processes	Westrum organizational culture
Deployment automation	Empowered teams	Value stream	Monitoring	Supporting learning
Continuous integration		Working in small batches	Proactive notification	Collaboration among teams
Test automation		Team experimentation	WIP limits	Job satisfaction
Test data management			Visualizing work	Transformational leadership
Shift left on security				
Continuous delivery (CD)				

More recently the DevOps Research and Assessment (DORA) [8] team has redefined a set of capabilities seen in Table 5.8, that correlate to Accelerate [5], but expands and refines them based on a several year research and data from practitioners worldwide. The correlation can be immediately noticed in the way continuous delivery and architecture categories are merged into a single technical category of capabilities.

The research here presented also identifies how an important capability such as **collaboration among teams** was dropped from Table 5.8 cultural category, when compared to Table 5.7, despite the fact being the most mentioned from all capabilities, 81 times out 93 publications seen in Figure 5.6.

But in general, it is noticed a growth in the number of technical and measurement capabilities, such as database change management, monitoring systems to inform business decisions, cloud infrastructure and code maintainability.

Table 5.8: List of four capability categories proposed by DORA [7, 8].

Technical	Process	Measurement	Cultural
Version control	Team experimentation	Monitoring systems to inform business decisions	Westrum organizational culture
Trunk-based development	Streamlining change approval	Monitoring and observability	Learning culture
Continuous integration	Customer feedback	Proactive failure notification	Job satisfaction
Deployment automation	Visibility of work in the value stream	Work in process limits	Transformational leadership
Continuous testing	Working in small batches	Visual management capabilities	
Continuous delivery			
Architecture			
Empowering teams to choose tools			
Test data management			
Shifting left on security			
Database change management			
Cloud infrastructure			
Code maintainability			

6

DevOps Metrics Multivocal Literature Review

Contents

Following the previous chapter, the process of construction of the artifact continues to the second MLR that gathers the DevOps metrics needed for the research proposal.

6.1 Planning the MLR

This section represents the first phase of the MLR process. It begins with an explanation of why this work is being done, then continues on to the objectives and research questions that are answered throughout the study. The following stage is to create a review protocol.

The full process shown in 6.1 exposes the planning, conducting and reporting as proposed by Garousi et al. [3].

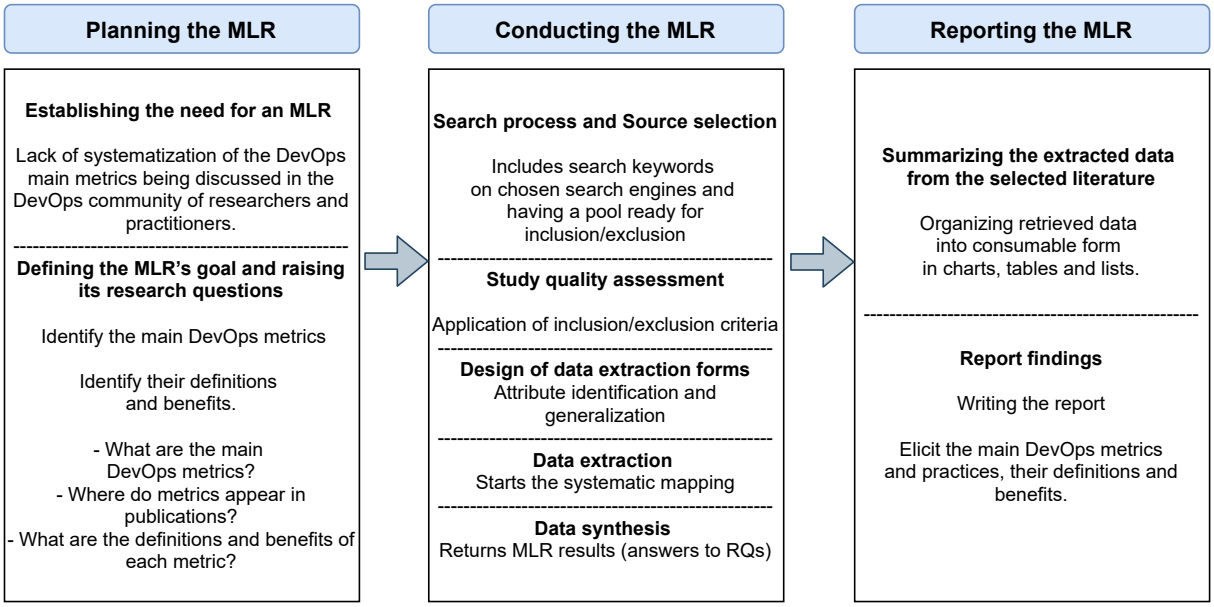

Figure 6.1: DevOps metrics Multivocal Literature Review (MLR) Steps [3]

6.1.1 Motivation

Management in software development firms that want to implement DevOps internally must have access to suitable supporting information systems and metrics surrounding this transformation in order to analyze the success and improve the efficiency [11] of using DevOps metrics. Management in software development businesses that wish to adopt DevOps internally must have access to appropriate supporting information and metrics regarding this technical change to assess success and enhance efficiency [11] of applying DevOps metrics [28, 37, 40, 136, 196–198].

However, there exists a lack of systematization of the key DevOps metrics being debated in the DevOps community of scholars and practitioners.

6.1.2 Research Questions

Based on the primary goal of this study, a MLR is used to search for scientific and 'gray' literature that discusses or investigates DevOps metrics, which can then be captured into RQ4, RQ5, and RQ6 research questions.

6.1.3 Review Protocol

The first batch of papers has been procured as seen in Figure 6.2 following the completion of the search and snowballing, inclusion and exclusion criteria is used to refine the search results in the first phase.

The review protocol used the workflow shown in Figure 6.2.

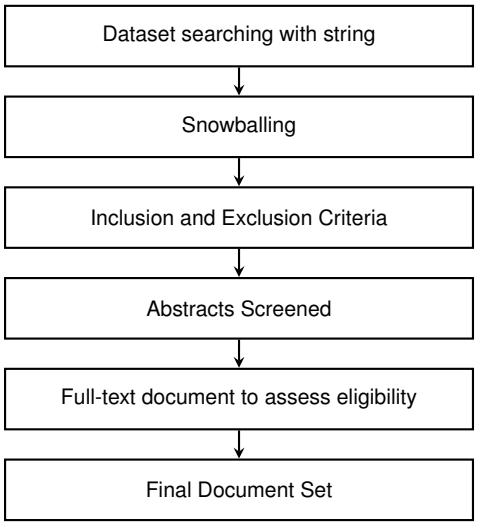

Figure 6.2: Review protocol performed in this research.

A search utilizing various keywords was done in April 2021 in attempt to identify additional studies connected to this study that may reach answers to the specified research topics. This section lists the search string used to get the most studies and datasets.

- **Search String**: `(devops AND (metrics OR measures OR kpi OR indicator))`.
- **Datasets**: The search engines used were, Google search[1], Scopus[2], Web of Science[3], IEEE[4], ACM[5] and EBSCO[6].

[1] https://www.google.com
[2] https://www.scopus.com
[3] https://apps.webofknowledge.com
[4] https://ieeexplore.ieee.org
[5] https://dl.acm.org
[6] https://search.ebscohost.com

To make finding and gathering large amounts of gray literature easier, the same python code from Section 5.1.3 was used, present in Listing B.1 (Python code for consistent fetching of large number of Google search results), to parse the data into two CSV files [70].

The criteria for inclusion and exclusion for this MLR is provided in Table 6.1. Following that, the abstracts must be evaluated to determine their relevance to the investigation. Finally, the relevant articles are reviewed in order to arrive at the final study selection for the review.

Table 6.1: Inclusion and exclusion criteria applied in this research.

Inclusion Criteria	Exclusion Criteria
Written in English	Unidentified author
Published in and after 2010	No publication date
Full-text accessible	Advertisement or Job Post
Mentions DevOps metrics	

6.2 Conducting the MLR

This section describes how the MLR's second phase review is carried out. At this time, the search is carried out over the specified databases using the search query, and an analysis is carried out on top of the retrieved data.

6.2.1 Selection of Studies

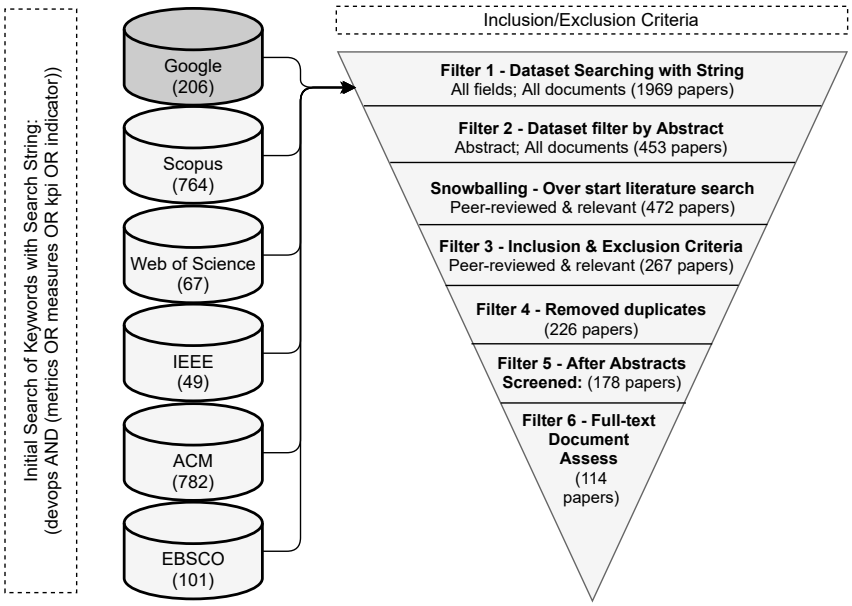

Figure 6.3: Followed Multivocal Literature Review process (adapted) [3].

In the first phase of the search, *filter 1* (All fields; All documents) was combined with the search term, both of which were found in Table 6.2.

The difference between *filter 1* and *filter 2* is warranted since the keywords could previously be located everywhere within the retrieved item, and some search engines return more literature than simply academic papers, such as newspapers or reports. In the case of the Google search engine, however, this is not the case, thus the results stay the same.

In the second iteration of search results, *filter 2* (Abstracts, All papers) was utilized and therefore the number of articles which include an abstract mentioning the keywords was reduced to 453 publications in total.

Table 6.2: Filters used in the MLR protocol.

Database	Search String	Filter 1	Filter 2	Sowballing	Filter 3	Filter 4	Filter 5	Filter 6
Google	`(devops AND (metrics OR measures OR kpi OR indicator))`	206	206		149	149	116	109
Scopus	`(devops AND (metrics OR measures OR kpi OR indicator))`	764	85		49	24	27	1
Web Of Science	`(devops AND (metrics OR measures OR kpi OR indicator))`	67	44		28	12	16	2
IEEE	`(devops AND (metrics OR measures OR kpi OR indicator))`	49	32	+19	18	18	6	0
ACM	`(devops AND (metrics OR measures OR kpi OR indicator))`	782	20		13	13	4	1
EBSCO	`(devops AND (metrics OR measures OR kpi OR indicator))`	101	66		10	10	9	1
	Total	1969	453	472	267	226	178	114

Filter 1 = Query All fields, All documents
Filter 2 = Query Abstracts, All documents
Snowballing = Applied over starting literature search [3]
Filter 3 = Relevant (inclusion/exclusion criteria)
Filter 4 = Remove duplicates
Filter 5 = After Abstracts Screened
Filter 6 = Full-text Document Assess

In the following stage, a *snowballing* [71] is performed, resulting in an increase in the overall quantity of articles to 19 more relevant publications identified.

We remain with 267 publications after applying inclusion & exclusion criteria *filter 3*, present in Table 6.1. This results in *filter 4*, which is defined to eliminate duplicates from the list of results in order to acquire the set of documents to have abstracts examined. Because there is no abstract for instances pertaining to gray literature, the entire material was skimmed, allowing a better assertion of inclusion or exclusion of that publication.

After screening all abstracts, 114 articles are left for full-text document review.

6.2.2 Data Extraction Analysis

Following the selection of the final collection of publications, an analysis of the different components of the findings is provided here, in a relationship of the final set of papers based on the source data. This study is based on an examination of the entire text of the 114 articles suitable for extraction of any relevant information for this research. An overview is also provided of which years and types of articles were chosen for complete reading.

The relationship of final document set by database, as represented in Figure 6.4, shows that the vast majority of 109 results originated from Google search, with 95.61 percent of gray literature. Only two publications are contributed by Web Of Science (1.75%). ACM, EBSCO, and Scopus each provided one item, leading to a total of three publications (2.63%) of relevant research articles.

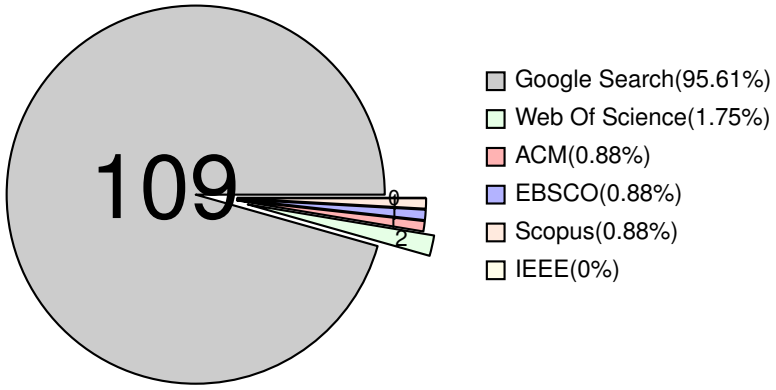

Figure 6.4: Distribution of final set of documents per database.

Finally, there were no matching papers for this study on IEEE. This validates the expectation that the practitioner community will produce a wider range of findings when compared to the scientific literature.

In Figure 6.5 it is reflected how publications have been evolving over the years with the biggest amount of generated literature appearing in webpages in the year 2020. The low number in 2021 reflects the fact that this research results were gathered in yearly March 2021.

Another interesting aspect to note is the early appearance in 2010 of the book “Continuous Delivery” [15] mentioning that metrics help improvements and efficiency in the continuous integration and delivery processes. According to Jezz Humble, a well-implemented deployment pipeline should make determining the cycle time simple. It should also display on how long it takes from check-in to each stage of the procedure. This is an effective method for identifying bottlenecks in operations.

Since 2013, the same author has been consistently contributing to the topic, including in numerous State of DevOps reports [16, 72, 73, 97, 100, 111, 113, 155] and in other relevant books. These important contributions to DevOps metrics are also shared with Joanne Molesky and Barry O’Reilly in “Lean enterprise : continuous delivery, DevOps, and lean startup at scale” [150] Gene Kim, Patrick Debois and John Willis in the “The DevOps Handbook” [76] and congregating efforts with Nicole Forsgren in

"Accelerate: The science of lean software and devops" [5] where important metrics are discussed based on the investigation done in yearly surveys and Tech reports, which have been consistent since 2013.

In Figure 6.5 it is observed an increasing interest in the sector in the last several years, despite a relatively low amount of research work done on studying metrics, demonstrating the potential appeal and utility of this research in the field.

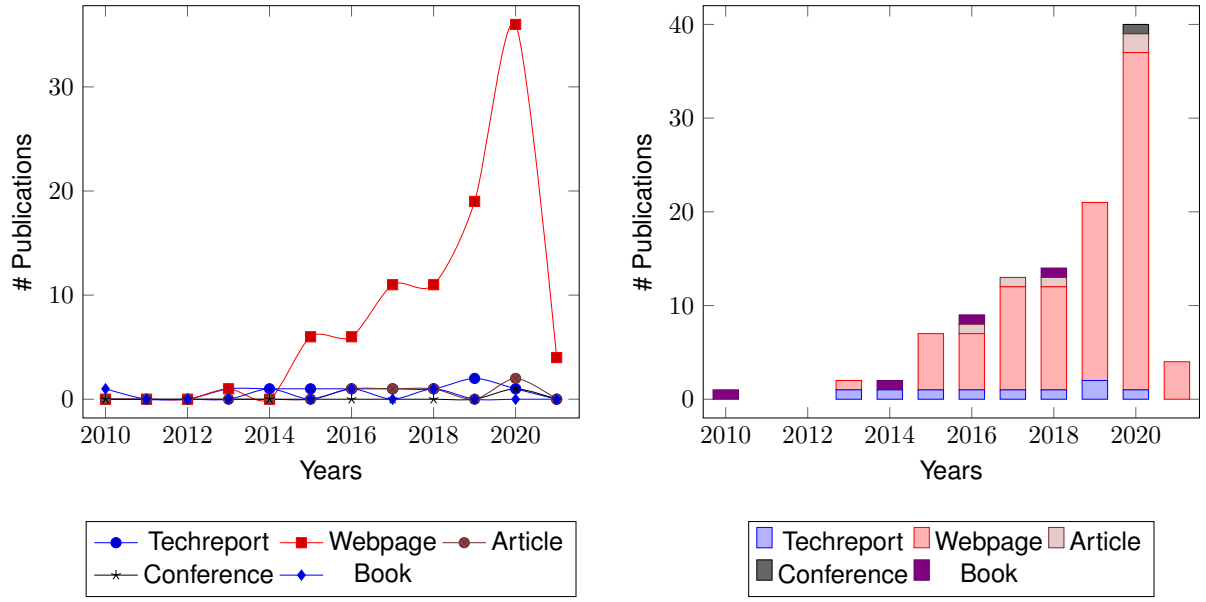

Figure 6.5: Distribution of publications per type over the years.

The Tech reports from 2013 and 2014 had a special importance in ramping up the interest on DevOps metrics topic and raising awareness for the fact that measurements are visible and actionable [72, 73]. While many companies were claiming to be data-driven, in the sense of gathering a lot of data, but relatively few really used that data to make educated choices. Thus posing the questions if DevOps analytics are being done on a regular basis and if action is being taken on them? It is proposed to improve organizational clarity by linking performance measurements to organizational goals rather than team or functional goals, and to turn data into meaningful, visible information that gives feedback to teams on important quality, performance, and productivity criteria.

Finally, the number of gray literature articles increased considerably in 2020, as demonstrated by the massive increase in web pages related content in that year, showing that practitioner writings grew far faster than scientific research.

In order to get an overview of how the various metrics have grown in the literature over the years, Figure 6.6, Figure 6.7 and Figure 6.8 can be observed.

In this study, 58 DevOps metrics were discovered for discussion in Section 6.3. These metrics are here listed from M01 to M58, within the following figures, in descending order, by the total amount of

mentions, from the publications accounted for in Section 6.2.1.

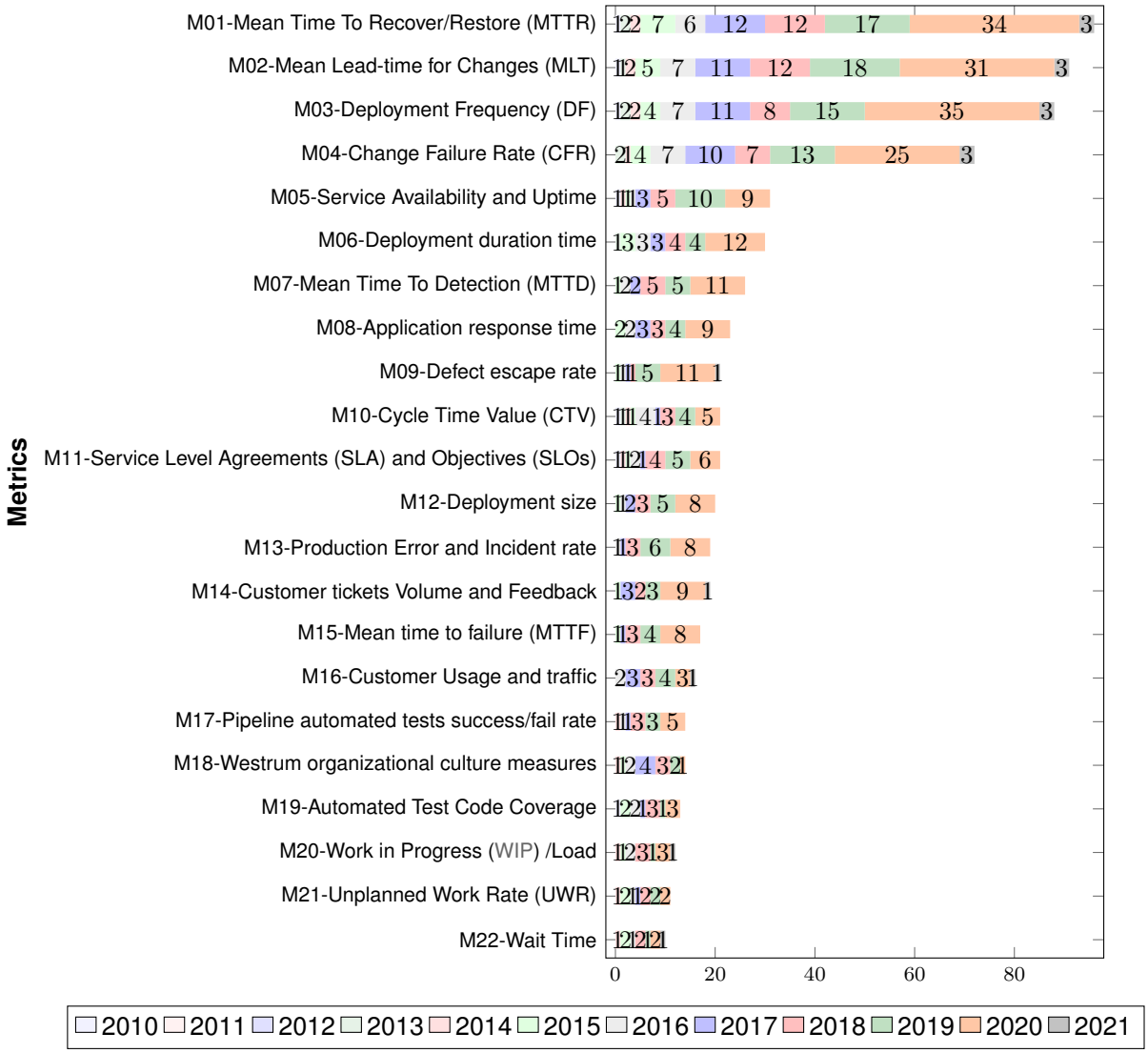

Figure 6.6: Top main metrics mentioned in publications over the years.

Because this MLR found 10 years with relevant publications out of the 12 years, it was accordingly chosen to use the metrics that are cited 10 or more times as the key DevOps measure for this research, present in Figure 6.6.

Therefore, in Figure 6.6 are shown the top 22 metrics, from M01 to M22. In this figure, a significant jump in 2020 can be perceived, namely, in *Mean Time to Recover/Restore (MTTR)*, *Mean Lead-time for Changes (MLT)*, *Deployment Frequency (DF)* and *Change Failure Rate (CFR)*. These four metrics were the most frequently stated, indicating that practitioners have agreed on their superior importance based on the articles reviewed.

If we compare MTTR (96 mentions) with CFR (72 mentions), there is only and a difference of 24

mentions, while if we compare MTTR with *Service Availability and Uptime* (30 mentions), there is a notable difference of 66 mentions. Therefore, while each of these 22 metrics have 10 or more mentions, there is a clear tendency of increased interest in M01,M02,M03 and M04.

Looking at the reasons pointed out during the 2020's jump in gray literature it is found that *Mean Time To Recovery MTTR (M01)* is an important metrics for DevOps teams [163, 199, 200]. The average cost of downtime for companies rises year after year [201]. MTTR emphasizes critical business outcomes that are directly related to customer experience, acquisition, and retention [202].

Mean Lead time (M02) is fundamental because it measures the time it takes for a code change to reach production. It gives insight into the DevOps process's efficiency, complexity, and the team's ability to meet customer needs [203, 204]. Short lead times suggest immediate feedback, while long lead times indicate inefficiency. The average time for a project to go from a concept to implementation is estimated at hours [205, 206].

Release Frequency (M03) approaches infinite in just-in-time manufacturing as batch size approaches zero [207], therefore release frequency of software is a Key Performance Indicator (KPI) for software delivery teams [41, 208–212]. A team that deploys more than once per week can fix outages in production faster and deliver value to customers more frequently [213–215].

Change Failure Rate (M04) is the percent of changes to production that require an immediate fix to resolve, thus a more complex metric [163]. If the issue generates an error, it's straightforward to track when compared to an issue affecting performance, making it important to differentiate feedback from a real issue.

An increasing failure rate reveals processes problems in the delivery pipeline [216–218]. In current industry benchmark all but low performers have a change fail rate between zero and 15 percent [140].

From the relation of these and the other metrics over the years, it is shown that practitioners are championing these metrics in their publications. Given that we are reporting the MLR in Section 6.3, the remaining main metrics are part of that reporting.

In Figure 6.7 the metrics are mentioned three to nine times in publications over the years. They are ordered from M39 to M58, have less relevance than Figure 6.6 but still are important to briefly describe here since they contain a fair number of references.

Out of the sixteen metrics exposed here, the most mentioned, with only nine mentions, is *M23-Number of Code commits* where too many commits may be indicative of low quality or lack of direction in development. If too low, it may be an indicator that the team is too taxed and non-productive [15, 76, 203, 205, 210, 219–222]. Next, *M24-Mean time between failures (MTBF)* is the average time between failures in production [5, 15, 139, 142, 162, 223–225]. These 2 metrics might be challenged by practitioners due to their questionable usefulness in DevOps adoption due to the misleading nature of dual interpretation of their values.

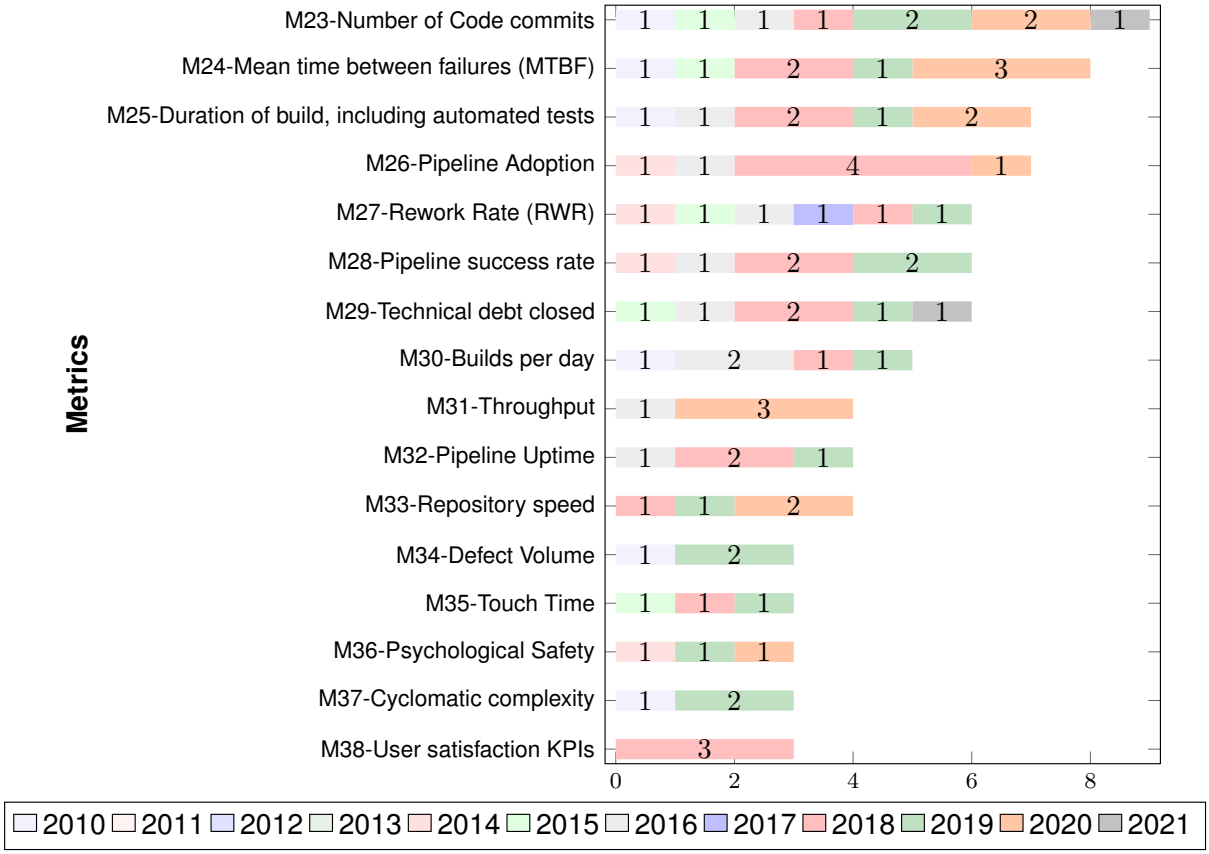

Figure 6.7: Metrics mentioned from three to nine times in publications over the years.

In the same line, with just seven mentions, it was found *M25-Duration of build, including automated tests*, the time it takes for quality verification [15, 148, 159, 220, 226–228] and *M26-Pipeline Adoption* measures the adoption of each definition done in using a DevOps pipeline [76, 143, 150, 224, 225, 228, 229], which are particularly interesting metrics, but still require possible further investigation.

Three other metrics are referenced six times, *M27-Rework Rate (RWR)* the proportion of items that must be reworked [16, 97, 150, 230–232], *M28-Pipeline success rate* the value of successful jobs in the CI/CD (Continuous Integration and Continuous Delivery) Pipeline [76, 150, 220, 227, 228, 233] and *M29-Technical debt closed* captures the resolution of a backlog of technology improvements that were first implemented less than optimally [5, 76, 108, 111, 162, 203].

From five publications, *M30-Builds per day* number of builds done per day [15, 76, 148, 220, 228]. Mentioned each with four references *M31-Throughput* is the amount of WIP that is put in production in a given period [15, 76, 148, 220, 228], *M32-Pipeline Uptime* measures the availability of the CI/CD Pipeline [76, 220, 227, 228], *M33-Repository speed* PR review duration and merge repository speed score is based on the time from Pull Request submission to merge (review duration) over the last 30 days [210, 213, 220, 228].

And finally with just three references *M34-Defect Volume* is the Number of defects for a particular application [15, 220, 230], *M35-Touch Time* is the amount of time spent adding value [229, 234, 235], *M36-Psychological Safety* means that team members feel secure to take risks and can be open with one another [111, 150, 236], *M37-Cyclomatic complexity* is a software metric used to determine the complexity of a program [15, 142, 234] and *M38-User satisfaction KPIs* can be affected by improving release frequency and quality [5, 108, 143].

The metrics in Figure 6.8 are ordered from M39 to M58 and have only been mentioned just one or two times in publications over the years, therefore they have the least relevance of the three figures in this section and in this MLR.

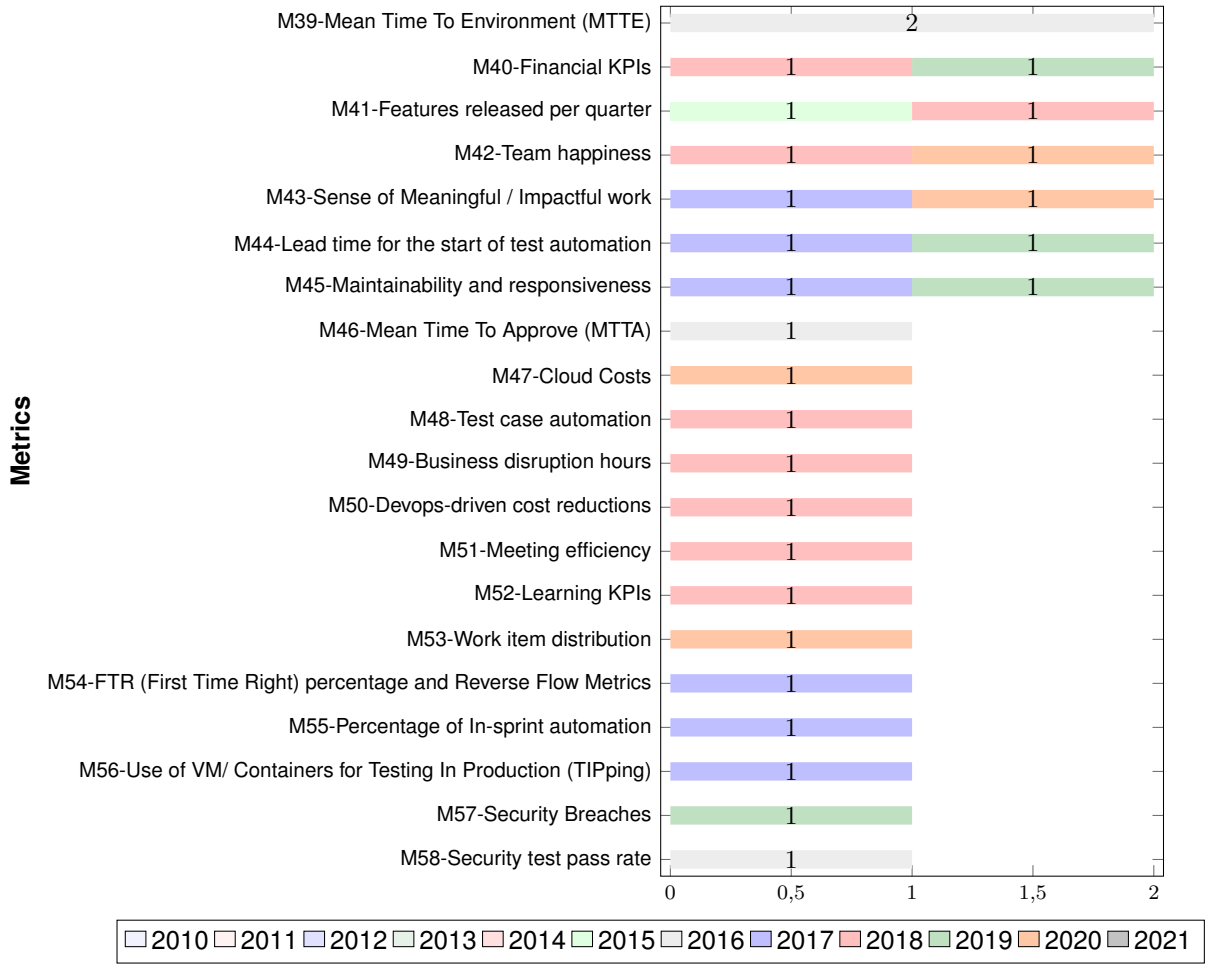

Figure 6.8: Metrics mentioned just one or two times in publications per year.

Aside from the fact that these have gained low relevance over time, it is important to notice that metrics M48 to M52 are all from the same publication [108] and the same happens in metrics M54 to M56 only mentioned in one article [237], thus confirming that the lack of variety on these metrics still

requires consensus within the practitioner's community.

Therefore, in line with what was stated regarding the lists present in Figure 6.8 and in Figure 6.7 these metrics are not going to be taken into account.

6.3 Reporting the MLR

The DevOps metrics are provided at this stage of the MLR, and all research questions are evaluated in light of the publications received through the study procedure.

6.3.1 RQ4 - What are the main DevOps metrics?

There are 22 main DevOps metrics found in this Multívocal Literature Review, gathered from all the publications, selected for review. The metrics are the following:

M01. Mean Time To Recover/Restore (MTTR)
M02. Mean Lead-time for Changes (MLT)
M03. Deployment Frequency (DF)
M04. Change Failure Rate (CFR)
M05. Service Availability and Uptime
M06. Deployment duration time
M07. Mean Time To Detection (MTTD)
M08. Application response time
M09. Defect escape rate
M10. Cycle Time Value (CTV)
M11. Service Level Agreements (SLAs) and Service level Objectives (SLOs)
M12. Deployment size
M13. Production Error and Incident rate
M14. Customer tickets Volume and Feedback
M15. Mean time to failure (MTTF)
M16. Customer Usage and traffic
M17. Pipeline automated tests success/fail rate
M18. Westrum organizational culture measures
M19. Automated Test Code Coverage
M20. Work in Progress (WIP) /Load
M21. Unplanned Work Rate (UWR)
M22. Wait Time

These are the main metrics, and in the following research question, a description of the purpose for each of these metrics is provided.

6.3.2 RQ5 - What is the purpose of each metric?

From the investigation done in this MLR it is shown in Table 6.3 the list on the main DevOps metrics including their ID, metric name, purpose description, the references that mention each specific metric and the total of references discovered.

Table 6.3: Purpose and references for each main DevOps metric.

ID	Metric	Purpose	References	Total
M01	Mean Time To Recover/Restore (MTTR)	Measures the mean of the time required to recover or restore service from a failure in production.	[5, 10, 15, 16, 19, 41, 44, 72, 73, 76, 97, 100, 108, 111, 113, 138–140, 142, 143, 146, 148, 150, 154, 155, 159, 162, 163, 199, 200, 202, 204–206, 208–211, 214, 216–227, 229–231, 233, 235, 236, 238–275]	96
M02	Mean Lead-time for Changes (MLT)	Indicates how long it takes for a change to go from code committed to code successfully running in production.	[5, 10, 15, 16, 41, 72, 73, 76, 97, 100, 108, 111, 113, 138–140, 142, 143, 146, 150, 153, 155, 159, 162, 163, 199, 200, 204–206, 209–212, 214–219, 221–223, 227–236, 238, 239, 241–243, 245–256, 258–261, 263–272, 276–281]	91
M03	Deployment Frequency (DF)	Checks how often changes are deployed to production.	[5, 10, 15, 16, 41, 44, 72, 73, 76, 97, 100, 108, 111, 113, 138–140, 142, 146, 148, 150, 153–155, 159, 162, 163, 199, 200, 204–206, 208–211, 213, 214, 216–220, 222–224, 229–231, 236, 238, 239, 241–256, 259–271, 273, 275–277, 279, 281]	88
M04	Change Failure Rate (CFR)	Informs how often a change in production fails and must be immediately remedied.	[5, 10, 16, 72, 73, 76, 97, 100, 111, 113, 138, 140, 142, 146, 148, 155, 159, 163, 199, 200, 204, 205, 208–212, 214, 216–218, 220, 222, 223, 229–232, 236, 238, 239, 242, 243, 245–254, 256, 259, 261–273, 275, 277, 279]	72
M05	Service Availability and Uptime	Shows the percentage a service is available during a period of time.	[5, 15, 44, 76, 84, 111, 139, 142, 143, 146, 150, 159, 206, 210, 213, 214, 220, 221, 230, 231, 233, 235, 237, 239, 244, 252, 254, 255, 260, 267, 277]	31

Continued on next page

Table 6.3 – *Continued from previous page*

ID	Metric	Purpose	References	Total
M06	Deployment duration time	Informs on how long it takes to deploy a set of changes.	[138, 139, 143, 148, 154, 159, 205, 206, 210, 214, 217, 220, 227, 229, 230, 235, 239, 246, 254, 255, 260, 262, 264, 270, 273, 275, 276, 279, 281, 282]	30
M07	Mean Time To Detection (MTTD)	Measures the mean of the time required to detect a failure in production.	[76, 108, 139, 142, 159, 202, 205, 206, 208, 210, 214, 219, 221, 223, 224, 229–232, 235, 239, 244, 252, 255, 260, 261]	26
M08	Application response time	How an application responds to increases or decreases in user traffic and activity.	[5, 76, 100, 139, 153, 154, 162, 205, 206, 210, 221, 229, 239, 244, 252, 255, 258, 260, 266, 267, 271, 274, 277]	23
M09	Defect escape rate	Indicates the number of defects discovered in production versus the number of defects found during development.	[108, 139, 148, 153, 154, 205, 206, 210, 211, 214, 217, 220, 223, 230, 239, 255, 257, 260, 273, 278]	21
M10	Cycle Time Value (CTV)	Provides information on the full Cycle Time Value, beginning with the idea and finishing with user feedback. .	[5, 15, 44, 72, 76, 148, 150, 159, 161, 162, 208, 215, 220, 225, 227, 230, 234, 240, 257, 265, 269]	21
M11	Service Level Agreements (SLAs) and Objectives (SLOs)	Sets customer expectations for service availability with SLA and internal teams with SLO.	[5, 15, 76, 84, 111, 139, 150, 153, 159, 161, 206, 214, 221, 230, 231, 235, 252, 255, 260, 276, 279]	21
M12	Deployment size	Shows the number of changes incorporated in each production release.	[139, 143, 159, 162, 205, 206, 210, 211, 217, 220, 228–230, 239, 255, 260, 267, 271, 277, 278]	20
M13	Production Error and Incident rate	Measures the frequency of faults and incidents in production following a deployment.	[139, 143, 154, 199, 206, 210, 217, 219–221, 224, 231–233, 255, 257, 258, 260, 276]	19
M14	Customer tickets Volume and Feedback	Indicates the level of satisfaction of customers using their feedback.	[139, 143, 159, 205, 206, 210, 211, 214, 222, 223, 230, 255, 260, 261, 267, 276, 278, 279]	19
M15	Mean time to failure (MTTF)	Exposes the average time a flawed deployment into a system will manage to run until it fails.	[139, 142, 159, 202, 205, 206, 213, 214, 219, 222, 224, 229–231, 241, 255, 260]	17
M16	Customer Usage and traffic	Measures usage and traffic of customer-facing applications when there are defined business goals to increase.	[76, 84, 108, 139, 148, 203, 206, 214, 221, 227, 255, 258, 260, 266, 267, 277]	16
M17	Pipeline automated tests success/fail rate	Shows the rate of success/failure of Pipeline automated test jobs.	[5, 76, 150, 205, 206, 210, 217, 220, 226–228, 255, 260, 277]	14
M18	Westrum organizational culture measures	Result of the Westrum cultural assessment [77]	[5, 19, 76, 84, 95, 111, 143, 146, 150, 155, 235, 262, 264, 265]	14
M19	Automated Test Code Coverage	Measures how many lines, statements, or blocks of code are tested using the suite of automated tests.	[15, 76, 139, 148, 213, 214, 220, 225, 227, 228, 235, 237, 273]	13

Continued on next page

Table 6.3 – *Continued from previous page*

ID	Metric	Purpose	References	Total
M20	Work in Progress (WIP) /Load	Presents the number of open issues of each type (story, defect, task).	[5, 76, 150, 203, 212, 213, 215, 221, 227, 234, 235, 281]	12
M21	Unplanned Work Rate (UWR)	Indicates the amount of time spent on tasks that weren't in the initial plan. Shouldn't be over 25%.	[5, 16, 84, 97, 150, 199, 205, 222, 230–232]	11
M22	Wait Time	Measures the amount of time spent waiting for the next step to add value.	[76,150,203,212,215,222,225,229,234,235]	10

Concluded

The 22 key metrics presented here are among the initial 58 identified in this MLR. It was taken into account not only the fact they were referenced ten times or more, as seen in Figure 6.6, but also their impact in profitability, productivity, product or service quality, operational efficiency, customer happiness, improvement of products or services supplied, and achievement of organizational and mission goals [243] through DevOps adoption, which are all traits and advantages leading to 22 main metrics, identified from the community.

6.3.3 RQ6 - Why is each metric important?

We now know what each metric is, but why is it necessary to measure each one? What is the significance or importance of each metric? Here the MLR goes over *why* these DevOps metrics are important.

M01. *Mean Time To Recover/Restore (MTTR)* handling unexpected outages should be as quick as possible and a focus for DevOps KPI monitoring, as it contributes to greater customer satisfaction, faster application delivery, and better cost control [41, 44, 204, 206].

M02. *Mean Lead-time for Changes (MLT)* is the hardest to measure of the top four key DevOps metrics. It is important to know how long it takes to deploy a change, and understand delaying problems like technical debt [72, 100, 243, 269].

M03. *Deployment Frequency (DF)* happens many times per day for elite industry performers [111, 140], where there is continuous development, testing, and integration of small changes continuously improving applications. Important to respond quickly to business requests for new features and to critical issues [260, 270].

M04. *Change Failure Rate (CFR)* should be as low as possible in DevOps. It is a critical metric that businesses must monitor, since unsuccessful deployments result in revenue losses and dissatisfied consumers [230, 252].

M05. *Service Availability and Uptime* should be more than 99.999% of the time for users of the

system, with a ratio based on availability, reliability and uptime. Achieving 100% availability is unrealistic, once planned downtime for maintenance is accounted for. Therefore, it is important to track and distinguish from unplanned downtime [206, 210].

M06. *Deployment duration time* allows to track the progress of the deployment. It can help identify potential problems and allow a dramatic increase in revenue by using that extra time to develop more value-added services [229, 260, 262] .

M07. *Mean Time To Detection (MTTD)* demonstrates the effectiveness of monitoring technologies and intelligent alerting techniques, assessing whether current response efforts are appropriate. High detection times may result in bottlenecks [139, 230].

M08. *Application response time* that are long may indicate bottlenecks that require attention, since it degrades user experience and satisfaction. The cause might be code, data access, protocol problems, or a variety of other factors [154, 206].

M09. *Defect escape rate* is an important DevOps metric to track how often defects make it to production. Abnormally high defect rates could be the first sign of problems in testing, qualification or in team performance. [206, 214, 255].

M10. *Cycle Time Value (CTV)* is the time it takes to go from idea to production, spanning all the steps of build, test, stage, and push to production. Slowing down the cycle with manual testing or assessments creates friction for developers and the business [230, 257].

M11. *Service Level Agreements (SLAs) and Objectives (SLOs)* serve to define external and internal availability goals as commitments between providers, clients and internal teams, defining how fast releasing is possible, while measuring that performance with Service level Indicators (SLIs) [111, 159, 161, 195, 230, 283].

M12. *Deployment size* is important to keep an eye on the deployment artifacts that are shipped to production with each release and track the amount of the bug fixes and feature requests delivered [139, 230, 255].

M13. *Production Error and Incident rate* tells the DevOps team how often new bugs appear in running applications. It is important to capture spikes in the error rate because these can indicate that something is not right. Not all errors are equally impactful for customer's trust [206, 224, 258].

M14. *Customer tickets Volume and Feedback* is a good assessment of a successful DevOps adoption. Reduce customer tickets by preventing bugs from reaching production, and repair them as fast as possible if they do, improving quality. Customer satisfaction leads to a competitive advantage [4, 139, 206, 223].

M15. *Mean time to failure (MTTF)* is an indication of how long on average the system or a component can run before failing after deployment. It can indicate problems with the deployment or quality of the software. For example, maybe there are not enough tests covering different scenarios that might contain

bugs [142, 219, 224].

M16. *Customer Usage and traffic* metrics allow tracking user engagement with application features. Increased engagement after an update may indicate users are pleased with the updates. If traffic reports indicate too much or no activity, there might be an issue, suggesting a malfunctioning component is generating the anomaly [139, 255, 260].

M17. *Pipeline automated tests success/fail rate* is another contributor to the speed of the DevOps process. Automated tests are faster than humans, but it should deliver the right results. To increase velocity, the team needs to make extensive usage of unit and functional testing. It is important to know how often changes are causing tests to break [206, 255, 260].

M18. *Westrum organizational culture measures* results are key to fostering a performance-oriented organizational environment stating Westrum et al. 2004 [77]. DevOps teams must be supported by transformational leadership [98, 284] to enable this culture, thus empowering strategic alignment and reducing conflict [146, 155, 262, 285].

M19. *Automated Test Code Coverage* is a DevOps best practice for measuring the percentage of unit or integrity tests coverage [139, 220].

M20. *Work in Progress (WIP) /Load* is a lean manufacturing principle shown in the Toyota Production System [189] that enhances teams overall throughput by limiting work in progress (partially completed work). This increases total velocity [203, 212, 213].

M21. *Unplanned Work Rate (UWR)* tracks the amount of time spent on unplanned work. A high UWR may indicate that efforts wasted on unexpected errors that were not identified early in the workflow. The difference between acting on warning signs or having an unexpected outage [286] is defined as unplanned work [16, 230].

M22. *Wait Time* (queuing or waste) is an estimate of the time that the work item spends idle in a non-productive state during its processing by the value stream. Wait time is in opposition to touch time when value is created [76, 212, 229].

6.3.4 DevOps Metrics Synthesis

This study was run on DevOps metrics. To find literature, Google search, Scopus, Web of Science, IEEE, ACM, and EBSCO were utilized, and 114 publications were recognized as relevant to this research area after applying the inclusion and exclusion criteria and snowballing.

- A definition of DevOps metrics was identified.
- In this literature review, 22 main DevOps metrics were identified out of a total of 58 metrics.
- From the 22 main metrics, there are top four metrics that the community agrees being the most important, and communities are focusing on them.

Table 6.4: Definition of DevOps metrics.

A **DevOps metric** is defined as a quantifiable, business-relevant, trustworthy, actionable and traceable [44, 139, 199, 225, 230] indicator that aids organizations in making data-driven decisions to continuously improve their software delivery process [19, 140, 205, 206, 223, 236, 237, 248, 251, 258, 261, 267, 268].

- The top four metrics are MTTR, MLT, DF and CFR.

It has been researched that these four key metrics have expected improvement outcomes from DevOps adoption. MTTR, determines the mean of the time required to recover or restore service from a failure in production. MLT, indicates how long it takes for a change to go from code committed to code successfully running in production. DF, ascertains how often changes are deployed to production. CFR, measures how often a change in production fails and must be immediately remedied.

From the extensive MLR done, it is understood that DevOps metrics should aim to quantify the right elements in order to understand if DevOps is working [162, 226, 251]. This is important to know, so that DevOps adoption can be measured and does not become more hype than value.

Like in the case of "value stream mapping" [124, 269], organizations have begun to adopt measurements techniques that will help identify areas that need improvement [15] and ensure produced software offers continuous improvements to the customer experience. To assess if DevOps efforts are successful, managers need consumable information based on a clear list of DevOps metrics comparing similar value streams across a common set of KPI [150, 212, 262].

A key performance indicator (KPI) is a metric that helps understand how an entity is doing against its own objectives [278, 287]. These metrics are fundamental to leverage the rigor of measurement, not only in DevOps, but also Software Engineering and Information Systems [11].

In Table 6.5 it is shown a few relevant properties that this MLR has identified from the publications. The most important factor is that almost all authors state metrics help improvements and efficiency (104) and a high number (86) associates the need of metrics with having pipeline automation in place.

This MLR benefited from the fact that more than half the publications (69) not only mention metrics, but also try to organize and explains each stated metric. There are 49 publications that also try to define what are DevOps metrics. Following those dispersed definitions this MLR is now able to propose a unified definition of DevOps metrics in Table 6.4.

The DevOps metrics typically measure either throughput, stability or quality [199], while quantifying a faster cadence (efficiency) and value addition (effectiveness) [215].

Metrics also enable DevOps teams to monitor and analyze collaborative workflows, as well as track progress toward high-level goals such as higher quality, quicker release cycles, and improved application performance [269].

It was found that DevOps metrics categorization is still dispersed and only a few authors try to cate-

gorize metrics (28), represented in Table 6.5.

Table 6.5: Six publication properties identified from the MLR.

Property	Publications	Total
Mentions that metrics help improvements and efficiency	[5,10,15,16,19,41,44,72,73,76,84,95,97,100,108,111,113,138–140,142,143,146,148,150,154,155,161–163,199,200,202–206,208–221,224–226,229–239,241–246,248–252,254,255,257–281]	104
Relates Pipeline Automation to Metrics	[5,10,15,16,19,41,44,72,73,76,84,97,100,108,111,113,138,139,142,146,148,150,153–155,159,161–163,199,200,202–204,210,212,213,216–220,223–230,233–235,237–239,241–248,251,253–257,259,263,265–270,273,274,276–280]	86
Organizes and Explains each Metric	[5,10,16,41,72,73,97,100,108,111,113,138–140,142,146,148,155,162,199,205,206,210–212,214–216,218–221,223,224,226,228–230,232,233,236,237,239,241,244,245,248,250–252,256,259–263,266,267,269–273,275–279]	69
Defines what are DevOps metrics	[5,10,16,19,44,73,76,97,100,111,113,139,140,146,148,150,155,163,199,200,202,205,206,208,215,218,223–225,229,230,233,235–237,245,248,251,253,255,257,258,261,264,267–269,272,276,278]	49
Tries to categorize metrics	[5,16,41,44,72,73,97,100,108,111,113,139,142,146,148,155,206,210,214,220–222,232,235,236,240,262,263]	28
Mentions associated Practice, Capability or Principles	[5,10,16,72,73,95,97,100,108,111,113,146,155,161,230,233,235,251,269]	19

The same is observed while trying to understand what metrics are associated with each practice, capability or principles of DevOps (19). These two missing pieces of structured knowledge are intriguing and a possible source of investigation in forthcoming studies.

In the various State of DevOps Reports [16, 72, 73, 97, 100, 111, 113, 155] a few important metrics have already been used over the years, namely a few IT Performance metrics.

Specifically, in the State of DevOps 2019 report [111], metrics are mentioned to mirror the effectiveness of the development and delivery process, and that they can be grouped in terms of *throughput* and *stability*. Throughput of the software delivery process is measured using *lead time for code changes* from check-in to release, along with *deployment frequency*. As for stability, it can be measured using *mean time to restore* a system and *change fail rate*, a measure of the quality of the release process. They provide a solid basis for an organization's metrics activities [200].

However, these high-level metrics can be drilled down into a more refined state or expanded in order to include others like *Service Availability and Uptime* — the time an application is available, *Defect escape Rate* — the number of defects that are found during a given unit of time, or *Mean Time To Detection* — the average time between when a problem arises in production and when it is detected. As part of this research, the main metrics are being expanded and defined is a list for Section 6.3.1 and Section 6.3.2, as mentioned in the objectives.

It was found that academic studies still demonstrate limited research in this area. However, the industry shows a rising interest in the usage of DevOps metrics. As a result, the employment of DevOps metrics should be thoroughly explored due to the possible influence on businesses.

For some of the most referenced metrics M02,M03,M04,M06 and M12 there seems to be a relation

to the continuous delivery DevOps capability in the references [15, 44, 111, 217, 251, 265] which could be of interest to conduct more investigation in future research towards exploring the relations to delivery practices.

7

Research proposal

Contents

This chapter focuses on the development of a proposed assessment model by eliciting the main DevOps metrics for each DevOps capability using the two MLR done and a set of semi-structured interviews as mentioned in Chapter 4.

Semi-structured interviews are a popular method in development research. They are very different from formal interviews, which follow a rigorous pattern of predetermined questions, focusing on certain themes but cover them in a conversational way [65].

7.1 Interviews with Practitioners

For the construction of the proposed artifact, this research used the results from the two MLR done in Chapter 5 and in Chapter 6 and a first set of 21 semi-structured interviews out from the total of the 31 Interviews done to DevOps practitioners seen in Table 7.1 and in Table 8.1. Semi-structured interviews were held online using Zoom[1] or Jitsi videoconferences[2] to broaden the possibility of reaching out to more practitioners worldwide. Slots of 45 minutes were made available via a webpage created for practitioners to signup for interviews at their preferred time and date, between May and July 2021.

The page informed participants of the research interview confidentiality, aimed to increase our understanding of how DevOps Capabilities can be further measured with their valuable opinion. The page also mentioned the research questions involved. A link to a spreadsheet was shared with participants before the interview showing the resulting lists of DevOps capabilities in Section 5.3.1 and metrics in Section 6.3.1 from the MLR. In this process, a form was used to collect the characterization presented in Table 7.1.

The first set with 21 interviews aimed to respond to the research questions present in this chapter, leading to a proposed artifact seen in Section 7.2. The second set aimed to evaluate the artifact in Chapter 8 by using 10 iterations of interviews.

7.1.1 Preparation

Preparation was done considering each practitioner's specific session to expedite and facilitate the interview process. After participants gave their consent to start the interview a spreadsheet with capabilities and metrics resulting from the MLR, that was empty at first, was screen-shared, and subsequently got populated and refined with answers, and the interviews followed a semi-structured framework show in Table A.1 aimed to focus on the research problem present in Chapter 1.

Before conducting each following semi-structured interview, a close observation was done on the previous interview results to have a thorough grasp of the issue and to reformulate appropriate semi-

[1] https://zoom.us
[2] https://jitsi.org

structured questions if needed [67]. As a result, better open-ended questions relating to the problem were included, to give the possibility for discovering new approaches to challenge and comprehend the artifact's evolution.

7.1.2 Practitioners Characterization

A total of 21 interviews were conducted in this section to build the research proposal. The practitioners interviewed and seen in Table 7.1 were: three individual contributors, six team leads and two principal engineers. The companies they work for are very diversified: Google, Noesis, Newdecision, Siemens, Azores Governement, Inuits.eu, IBM, amazee.io, Acquia, Drupal Association, Manifold, Bloomidea, Deeper Insights, Facebook, Dropsolid and some freelancers. Twelve practitioners have more than 3 years experience on DevOps, while eight practitioners have between 1 year and 3 years experience and only one have less than 1 year experience. From all the 21, eight are individual contributors with diverse roles, six are Team Leads, two Principals and other five are at management level or above.

Table 7.1: First batch of interviews with practitioners' details.

ID	Interview Date	Current position	Company	Years of DevOps practice	Age (years)	Country
I01	2021-05-07	Principal Architect	Google	More than 3 years	45-54	United States
I02	2021-05-14	Indvidual contributor	Noesis	Between 1 year and 3 years	25-34	Portugal
I03	2021-05-14	Team Lead	Freelance	Between 1 year and 3 years	35-44	Portugal
I04	2021-05-17	Indvidual contributor	Newdecision	More than 3 years	45-54	Portugal
I05	2021-05-18	Team Lead	Siemens	Between 1 year and 3 years	25-34	Portugal
I06	2021-05-18	Indvidual contributor	Azores Governement	Between 1 year and 3 years	45-54	Portugal
I07	2021-05-19	Principal	Inuits.eu	More than 3 years	45-54	Belgium
I08	2021-05-19	Team Lead	IBM	More than 3 years	45-54	United States
I09	2021-05-21	Team Lead	amazee.io	Between 1 year and 3 years	25-34	Switzerland
I10	2021-05-22	Indvidual contributor	Acquia	Less than 1 year	45-54	UK
I11	2021-05-24	Software Infrastructure Engineer	Freelance	Between 1 year and 3 years	45-54	Spain
I12	2021-05-26	Team Lead	Siemens	More than 3 years	35-44	Portugal
I13	2021-05-27	Indvidual contributor	Drupal Association	More than 3 years	35-44	United States
I14	2021-05-27	Manager	Manifold	More than 3 years	25-34	United States
I15	2021-05-31	Director	Bloomidea	More than 3 years	35-44	Portugal
I16	2021-06-01	Director	Deeper Insights	More than 3 years	35-44	Portugal
I17	2021-06-07	Team Lead	Facebook	More than 3 years	35-44	United States
I18	2021-06-09	VP	Acquia	More than 3 years	> 55	United States
I19	2021-06-10	Senior Security Engineer	Acquia	Between 1 year and 3 years	25-34	India
I20	2021-06-15	Director	Dropsolid	More than 3 years	35-44	Belgium
I21	2021-06-18	Indvidual contributor	Acquia	Between 1 year and 3 years	25-34	India

The majority of 7 practitioners is between 35-44 years of age, six are between 25-34 years, seven are between 45-54 years and one has more than 55 years. Their locations are Belgium, India, Portugal, Spain, Switzerland, USA, and United Kingdom.

7.1.3 Conducting

The same research questions, including "What are the main DevOps Capabilities or practices?" from Section 5.3.1 and "What are the main DevOps Metrics?" from Section 6.3.1, were asked to practitioners during the semi-structured interviews to confirm the findings in the two MLRs, with the addition of Section 7.1.5 "What are the main metrics for each DevOps capability?", present in this chapter.

Interviewees validated the suggested set of capabilities and metrics, with one deeper validation occurring through a follow-up interview from I12. As an outcome, for each participant, a digestible classification of capabilities and measurements emerged, as well as the basic relationships of the capabilities that have an influence on which metrics. Furthermore, in other cases, validation resulted in the capabilities and metrics being adjusted or increased to align with the interviewee's ideas. In this section, the highlights of those ideas in each interview are summarized.

The first interviewed practitioner (I01) gave substantial importance to the cultural aspects, starting with team collaboration capability (C01) first and then other capabilities, always focusing more on the culture measures, which are reflected in Table 7.5 and also reinforcing that it was his personal opinion that organizations need to come up with a process that makes sure that people are being included despite their diversity, culture, religion, or the fact of being remote. All that can be reflected in two new added cultural KPIs, suggested being **team happiness** and **talent retention**, that are a part from the Westrum organizational culture (C24). Therefore, and with the evolution of the research with other interviews, previously defined metric *team happiness* (M42) was pulled in from the MLR and a new outcome, *talent retention* (M59) was added as a valuable metric. When talent is leaving the organization in large numbers, it should raise a flag to management to review the related capabilities' status. This was also confirmed by all the practitioners in this first phase of interviews when asked. *Talent retention* can be simply expressed as seen in Equation (7.1).

$$Talent\ retention = \frac{Employees\ number - Employees\ departed}{Employees\ number} \tag{7.1}$$

Another important point brought up is that automation is not only in the code pipeline; it is end-to-end; automation will be a part of almost everything in DevOps. For example, if monitoring identifies a problem, it should alert the relevant persons so that people are aware of the situation before it becomes a big issue. The goal must be to decrease manual work and delivery time as much as possible while still explaining why we do what we do.

The second interview (I02) brought new insights into the definitions and categories of capabilities, giving a good base to the result worked thought out all interviews and seen in Table 7.3. In this interview, there was more focus in the top four metrics, with examples given that suggest the high relevance of these outcomes for organizations.

In interview three (I03) practitioner emphasized the importance of team style and the various approaches used in each organization, as well as a considerable number of metrics reflected in Table 7.4. Good automated security systems are rare, expensive, and complex, with security ending up being handled to the left of the process and usually in a manual form. It's also more common for the security team to examine a release rather than a code review, like in Shift left on security (C20), which could interfere if metrics gathering (C23) is automated. In turn, job satisfaction (C33) is essential for a well-functioning company to maintain a business. Finally, the organizations where this works best are those where the metrics to be assessed are well-defined because there aren't many discrepancies between the data producers and the data consumers. There is a good fit between the product and the type of data.

The fourth interview (I04) focused on reconciling the relationships between the capabilities that influence each metric and cementing the categorization of each one. It was mentioned that by experience, the transformation driven by DevOps capabilities does have impact on several of the metrics leading to a better work-life balance and positive business outcomes like in the case of MLT (M01), MTTR (M02), CFR (M04) and several others with a special focus on Emergency response (C17) in order automate response to failures. Cross team collaboration (C01), *Continuous Integration* (CI) (C02) and *Continuous Delivery or Deployment* (CD) (C03) were again mentioned to have sustained relevance in the practitioners company, giving several examples where the metrics impact was witnessed internally.

The concept of DevOps capability defined in Section 5.3.4 was confirmed in the fifth interview (I05) practitioner. Moreover, the capability Shift left on security (C20) has been discussed, given that this process normally occurs later in the release or even in production, and there is a lot of value in moving it to a more initial stage of the process to automate and optimize delivery and reduce the number of security issues in production that might have serious impact. The WIP/Load (M20) metric was mentioned as having significantly more weight in their daily lives due to their involvement in operational processes.

In the sixth interview (I06), the practitioner expressed an interest in implementing the study's findings in the current role, even though many of the capabilities had yet to be implemented in the organization. Many questions were placed relatively to the connection of the version control system (C07) to continuous integration (C02), the centralization of logs (C26), Trunk based development (C22) and user feedback (C30) because it passes through a formal internal hierarchical structure. Other valuable experiences were given, focusing on the visibility of the work in the value stream (C28), Code maintainability (C36) and the importance of feedback in determining whether the implementation is effective in production.

For interview seven (I07), the practitioner shown a high level of experience giving refreshed directions to the capabilities relating to metrics. Capabilities need to be grown from team level up and supported by Transformational leadership (C21). Team happiness (M42) and talent retention (M59) were pointed as critical, but it is really hard to measure the path to get there. Most failures in DevOps adoption reside

in the team happiness factor. Capability (C10) raised the fact that new tooling introduces changes to the process and eventually changes the people's culture. But if it is just tried to change the people, it will work for some, but others will remain indifferent.

Many businesses still have space for improvement in configuration management (C11), and many firms don't actually require all the cloud or cloud native benefits (C12) since their needs are simple and their load is static/predictable with a fixed number of users. Artifacts (C13) are a hard requirement, and (C18) Containerization done right is also wonderful, but most of the time fails. This interviewee's experience started with early DevOps days conferences advocating for open source mindset (C19), claiming the culture of ownership in house and avoiding the vendor lock in. More recently, in March 2021, when a large datacenter (OVH) burned down [288], his customers were able to withstand the outage due to their resilience gained with from DevOps culture, open source tooling, and automation Lastly it was pointed out a site with further valuable DevOps information [289] that was meanwhile analyzed.

In interview eight (I08), the practitioner stated that if a team collaborates well (C01), it indicates that they are providing a high-quality service and a team that knows their flow is probably high performant. In production, this would imply that it would have extremely low numbers for the time it takes to recover (M01) and very low numbers for lead time (M02) or wait time (M22). The interviewee mentioned that is favorite capability was (C31) Blameless Postmortems, a learning capability that should affect positively several metrics that were taken note. It was raised a concern that MTTF (M15) is a metric that is hard to track and should be more related to how long is the build broken for in the cases of deployment. From the other capabilities that were discussed in this interview, that one that also had more relevancy was (C09) Trust/empower teams to make decisions and changes correlated to having a Lightweight/streamlining change approval (C27).

It was noted in the ninth Interview (I09) that some organizations are presently focusing on embracing all the new technology, and that seems like a knowledge reset. Cross team collaboration (C01) and encouraging learning culture while experimenting (C08) are essential parts of DevOps adoption. Continuous improvement processes and workflows (C06) was also mentioned as essential to answer the need of looking at the work that interrupts from the outside, unplanned work rate (M21) also called toil. Lastly, open source mindset (C19) was also mentioned to be vitally connected to important metrics like team happiness and talent retention and that it is vital to communicate changes properly and that is where lightweight change approval (C31) should fit mostly.

Practician in the tenth videoconference (I10) is a seasoned Ops person and tends to give importance to learn more by experimenting (C08), knowing that experimenting is learning so when someone else wants to learn something a server is set up to do this, other Operational people will also know how to install this and that is a kind of training, which is also kind of experimenting. It is good if the organization

can say - “Okay, here is some time for learning, and part of the learning is you’re going to exercise and do something.” - for instance, on improving Proactive Monitoring, Observability and autoscaling (C04). Today, cultural attitude is crucial for team cooperation and communication (C01), and interviewee mentions achieving exception results in OPS, but that would be something to consider.

In interview, eleven (I11) is a Drupal[3] developer that started by mentioning that had looked at the shared spreadsheet and that he found this an interesting research, since in his experience there are organizations that still struggle with version control systems (C07) and that should evolve. Trusting the team to make decisions and adjustments (C09) is essential. There is a difference between when individuals try to avoid performing a task like Database change management (C15) by having someone else do it, or when they understand the danger and try to do it themselves, which makes a considerable impact. Continuous process and workflow improvement (C06) has led to, properly planned things, and we are now aiming for a release every three to four weeks. However, there was a time when we didn’t have a clear timetable and would just keep adding items until we thought they were finished, such as a security update on Thursday or something similar. This also had an impact on MLT.

The twelfth practitioner started by mentioning that regarding team happiness (M42) and talent retention (M59), his organization gives importance to the level of inclusion and diversity, so much so that the current job offers all mention this fact. On the operational side, it is essential that the team is focused on solving problems in production but also upstream, as the level of technical debt can exponentially increase the team’s work with unplanned work (M21). Incentives are given to contributors to improve the process to reduce production errors rate and incidents (M13). On the other hand, having Visual management capabilities (C37) helps to reduce Work in Progress (WIP)/Load. Finally, the practitioner also mentioned that the team performance is measured through a survey every 6 months, in which all team members and leadership are invited to participate. This leads to the fact that transformational leadership (C21) has a great impact on the outcome of the westrum model assessment (M18).

Practitioner (I13) mentioned that in the current organization, there was a focus on KPIs and measuring many things that were brainstormed during meetings, where the team came up with ways to measure it, but having the research artifact in Table 7.6 would make things easier. It was noted relating to Open source (C19) that it is sometimes used to things that are not just software adoption, such as open source data could be used in Test data management (C34), and so we are looking at how this open source adoption may be evaluated by these KPIs, as well as how and which ones it influences. Consider working in small batches (C25) so that engineers are in a position where there is time to contribute back to an open source project. Centralized Log Management (C26) was done on their systems recently, which had a very visible and positive impact.

Interviewee (I14) focused in grand part in Containerization (C18), stating it helps the testing cycle

[3]https://drupal.org

going to go faster and also impacting release deployment frequency. In his last organization, when they switched to containers, the deployment went five times more frequently. This was less not only related to switching to containers, but also because of switching to continuous deployments. It is unknown what capability had more influence there, but failure rate, which is the difficulty or possibility of a deployment succeeding.

The following participant (I14) made an important contribution to this research by focusing on capabilities such as Test Automation and environments (Continuous testing) (C05), Version Control System (C07), Customer/User Feedback (C30), and Data-driven Approach for Improvements (C32), highlighting their relationships to specific metrics and solidly improving the artifact. It was mentioned the time wasted in manual testing, leading to slow release frequency and decreasing team happiness. Therefore, a challenge faced in the practitioner's company is how to fully automate those tests and make that a part of releasing.

The sixteenth practitioner (I16) shown a lot of interest in this research diving first into the Focus on people, processes and technologies (C10) capability, mentioning evaluation in their company is based on the number of changes proposed by individuals and determining which ones were implemented. For these technologies, there is a three-phase onboarding process that includes assessment, trial, and adoption stages. In this well-known business strategy (C10), people employ process and technology to perform tasks for the companies they work for, but organizations might become immersed by new technology at times. As a result, they utilize them without weighing against current procedures. To solve this problem, it is important to drive the right balance between people, processes, and technologies. One example of this is pipeline adoption. Finally, Lightweight/streamlining change approval (C27) was mentioned to be associated to the peer review approval process to optimize and expedite the time for commits and releases to be accepted.

Practitioner in Interview seventeen (I17) was very proactive in going through the capabilities and metrics. It was mentioned that there is a perspective shift when it comes to practices versus capabilities. If talking about a capability, it is a third-party assessment of something you are looking at from the outside. For instance, the organization can release so many times a day and the lead time in releasing is this long and the defect rate is this low. Where a practice is more internally, saying we do practice these things. It was approached in detail the Artifacts versioning and registry (C13) mentioned on the high impact on several of the metrics pointed out in Table 7.5. On the other hand, for transformational leadership (C21) to happen there needs to be a coalition of leaders, and they are deliberately cross-functional (C01), to bridge the gap so that actions are aligned with the central goal or premise. Identifying your challenge, creating a strategy around that challenge, communicating that to people and support that culture (C08).

Cross team collaboration and communication (C01), is an historical challenge in engineering of many

organizations. It is siloed from the rest of the organization and even within the engineering organization. This becomes more apparent when attempting to organize releases with several teams engaged. Things don't line up because team A is reliant on team B, and team B is unaware that there is a deadline, so they're all out doing their thing, or they make a change without understanding it would affect another team. One of the ways we could help solved this is by monitoring systems to inform business decisions (C23), but that will still need to have the cross collaboration necessary for it to work.

Chaos engineering (C35), for the next practitioner (I19) began when designing a risk management framework for a DevOps pipeline, with the question: how do we reduce infrastructure and security incidents? It involves automation and machine learning, so that anomalies are automatically detected, try to correct them and bring the system back to normal, reducing MTTR to minimum if done right. It is an approach to keep the system, stable and consistently normal. Shift left on security (C20) entails including security checks and activities into the pipeline in development stage. The objective is to guarantee that the codebase is intended to be secure from the start using automation.

During the twentieth interview (I20), the practitioner mostly focused on Configuration Management (C11) and when comparing it to infrastructure as code (C16) the first is acceptable for the application on its own while infrastructure as code is valid for controlling a full platform. Configuration Management has a direct positive impact on MLT, CFR and several other metrics seen in Table 7.5.

The last interview done for the research proposal (I21) touched several capabilities as well but focused more on Loosely coupled architecture (C14), which means that microservices should know nothing about each other and that any modification to one service should have no impact on the others. For DevOps, it is important that even if one service fails, the others keep functioning. Furthermore, deploying an independent service that doesn't affect the others, but mostly test and verify the service without needing an integrated environment. On the other hand, Infrastructure as Code (C16) employs definition files, has self-documented systems and processes, is completely versioned in source control, allows for continuous testing of systems and processes, publishes minor modifications rather than batches and maintains services available at all times.

7.1.4 Data Saturation

In this study, data saturation is observed in order to determine the state of the artifact and prepare it for the evaluation phase. In qualitative research, saturation has gained wide support as a methodological concept, with the purpose of understanding when more data gathering and/or analysis is unnecessary based on the data that has already been gathered and analyzed [290].

This is observed at the point that new data tends to be residual when compared to data already collected. In interviews, for instance, when the researcher starts hearing the same remarks again and over, data saturation is being achieved [291, p. 372]. Then it's time to stop gathering data and start

evaluating what has been gathered [292, p. 26].

The reason for saturation in this case is due to a repeated observation of the same comments and a declining trend in the percentage of changes suggested by participants relative to the total possible relations, as seen in Table 7.2. Despite the fact that the percentage values are not zero in absolute terms, there is a clear trend in which the last five contributions are below one percent in relation to the total available relations in the matrix, and because there is a time restriction, that is defined as this phase ending criteria. As a result, we chose to stop after 21 interviews in order to move on to the next step of the DSR, which is the evaluation process in Chapter 8.

Table 7.2: Total contributions from participants during the build phase.

	Relations			Updated or added		
Interview	Contributed	Total	Percentage	Capabilities	Metrics	Categories
I01	18	18	2.03%	0	2	1
I02	14	32	1.58%	1	0	0
I03	26	58	2.93%	3	0	0
I04	22	80	2.48%	0	0	0
I05	17	97	1.91%	0	0	0
I06	23	120	2.59%	0	0	0
I07	35	155	3.94%	1	0	0
I08	28	183	3.15%	2	2	0
I09	24	207	2.70%	0	0	0
I10	20	227	2.25%	0	0	0
I11	15	242	1.69%	2	0	0
I12	18	260	2.03%	0	0	0
I13	13	273	1.46%	0	1	0
I14	9	282	1.01%	0	0	0
I15	12	294	1.35%	0	0	0
I16	11	305	1.24%	0	0	0
I17	8	313	0.90%	0	0	0
I18	6	319	0.68%	0	0	0
I19	7	326	0.79%	0	0	0
I20	4	330	0.45%	0	0	0
I21	2	332	0.23%	0	0	0

Data saturation, showing residual gains < 1%.

In Table 7.2 it is seen the number of contributed relations, the total growing since interview I01 and the percentage of relations added, relative to the total possible relations, where from 37 capabilities times 24 metrics it is possible to have 888, relations. For instance, in the case of I07, where $3.94\% = 100 \times 35/888$.

As described in the Section 7.1.3, there were cases where there were suggestions for improvements in Capabilities, Metrics, and categories and those numbers are also captured in Table 7.2. For instance, in I01 it was suggested and later confirmed by other practitioners to add a new metric called *talent retention* and to update the list with the existing metric **team happiness**.

7.1.5 RQ7 - How are DevOps capabilities categorized?

During the interviews the alignment of capabilities into categories was discussed, as well as their relevance and categorization leading to a more refined list as seen in Table 7.3.

Table 7.3: Categorization of DevOps capabilities.

Category	ID	DevOps Capability
Cultural	C01	Cross team collaboration and communication
Technical	C02	Continuous Integration
Technical	C03	Continuous Delivery and Deployment automation
Measurement	C04	Proactive Monitoring, Observability and autoscaling
Technical	C05	Test Automation and environments (Continuous testing)
Process	C06	Continuous Improvement of processes and workflows
Technical	C07	Version Control System
Cultural	C08	Support learning culture and experimentation
Technical	C09	Trust/empower teams to make decisions and changes
Process	C10	Focus on people, process and technology
Technical	C11	Configuration Management
Technical	C12	Cloud infrastructure and cloud native
Technical	C13	Artifacts versioning and registry
Technical	C14	Loosely coupled architecture/ microservices
Technical	C15	Database change management/ release alignment
Technical	C16	Infrastructure as Code
Measurement	C17	Emergency response/ proactive failure notification
Technical	C18	Containerization
Cultural	C19	Open source software adoption
Technical	C20	Shift left on security
Cultural	C21	Transformational leadership
Technical	C22	Trunk based development
Measurement	C23	Monitor systems to inform business decisions
Cultural	C24	Performance/Westrum organizational culture
Process	C25	Working in small batches
Technical	C26	Centralized log management
Process	C27	Lightweight/streamlining change approval
Process	C28	Visibility of work in the value stream
Measurement	C29	Working in progress limits
Process	C30	Customer/user feedback
Cultural	C31	Blameless Postmortems/reduced fear of failure
Process	C32	Data-driven approach for improvements
Cultural	C33	Job satisfaction
Technical	C34	Test data management
Technical	C35	Chaos Engineering
Technical	C36	Code maintainability
Measurement	C37	Visual management capabilities

With this resulting table, a matrix for DevOps capabilities and metrics was obtained that includes Cultural, Measurement, Process, and Technical categories to be used in Section 7.2.

7.1.6 RQ8 - How are the main metrics categorized?

With the objective of categorizing the main metrics Table 7.4 was elicited. There are shown Business, Change, Cultural and Operating types of Key Performance Indicators (KPIs).

Table 7.4: Categorization of main DevOps metrics.

Proposed Category	ID	Main Metric
Operating KPI	M01	Mean Time To Recover/Restore (MTTR)
Change KPI	M02	Mean Lead-time for Changes (MLT)
Change KPI	M03	Deployment Frequency (DF)
Change KPI	M04	Change Failure Rate (CFR)
Operating KPI	M05	Service Availability and Uptime
Change KPI	M06	Deployment duration time
Operating KPI	M07	Mean Time To Detection (MTTD)
Operating KPI	M08	Application response time
Change KPI	M09	Defect escape rate
Change KPI	M10	Cycle Time Value (CTV)
Operating KPI	M11	SLAs and SLOs
Change KPI	M12	Deployment size
Operating KPI	M13	Production Error and Incident rate
Business KPI	M14	Customer tickets Volume and Feedback
Change KPI	M15	Mean time to failure (MTTF)
Business KPI	M16	Customer Usage and traffic
Change KPI	M17	Pipeline automated tests success/fail rate
Cultural KPI	M18	Westrum organizational culture measures
Change KPI	M19	Automated Test Code Coverage
Operating KPI	M20	Work in Progress (WIP) /Load
Operating KPI	M21	Unplanned Work Rate (UWR)
Operating KPI	M22	Wait Time
Cultural KPI	M42	Team Happiness
Cultural KPI	M59	Talent retention

Interviewees also mentioned that it is difficult for organizations to track these metrics and capabilities, it requires a lot of effort and expertise to get this data (I03,I09). Thus, if a company switches from traditional systems with already figured out monitoring and metrics, they lose a lot of what was implemented, therefore if the migrations of these metrics is not timely considered, there is loss of what was experimented and learned over time, which could have a negative impact on the DevOps adoption.

7.1.7 RQ9 - What DevOps capabilities have a positive impact in which main metrics?

Given the objective of eliciting metrics that are impacted positively by capabilities that is represented in Table 7.5, having in mind the importance to focus on essential metrics rather than numerous metrics (I03,I08,I11,I16).

Table 7.5: DevOps capabilities influencing main metrics.

ID	M01	M02	M03	M04	M05	M06	M07	M08	M09	M10	M11	M12	M13	M14	M15	M16	M17	M18	M19	M20	M21	M22	M42	M59
C01	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓				✓					✓	✓
C02	✓	✓	✓	✓		✓			✓	✓		✓			✓		✓		✓					
C03	✓	✓	✓	✓		✓				✓					✓			✓				✓		
C04	✓				✓		✓	✓			✓		✓			✓					✓			
C05		✓	✓	✓						✓			✓	✓		✓	✓		✓				✓	
C06	✓	✓	✓							✓								✓		✓	✓	✓		
C07	✓	✓	✓										✓						✓			✓	✓	✓
C08			✓					✓		✓			✓				✓	✓					✓	✓
C09	✓	✓	✓	✓	✓	✓				✓			✓	✓	✓		✓	✓		✓		✓	✓	✓
C10		✓	✓											✓				✓					✓	✓
C11		✓		✓															✓				✓	✓
C12	✓	✓	✓		✓	✓		✓		✓	✓										✓	✓	✓	
C13	✓		✓				✓					✓									✓	✓		
C14	✓	✓	✓	✓				✓			✓	✓		✓	✓					✓		✓		
C15	✓		✓	✓	✓				✓				✓	✓	✓						✓		✓	
C16	✓	✓			✓	✓	✓			✓		✓								✓	✓	✓	✓	
C17	✓				✓		✓	✓			✓			✓						✓	✓	✓		
C18	✓		✓	✓	✓	✓		✓			✓	✓					✓		✓		✓		✓	
C19		✓	✓							✓									✓			✓	✓	✓
C20	✓	✓	✓	✓	✓		✓	✓	✓					✓						✓	✓		✓	✓
C21	✓	✓	✓	✓	✓					✓				✓				✓					✓	✓
C22	✓	✓	✓	✓					✓	✓		✓	✓					✓			✓		✓	
C23	✓			✓	✓			✓					✓				✓							
C24	✓	✓	✓	✓			✓								✓			✓				✓	✓	✓
C25		✓	✓			✓					✓	✓	✓	✓				✓		✓	✓	✓		
C26	✓			✓	✓		✓						✓	✓							✓	✓	✓	
C27	✓	✓	✓							✓								✓					✓	
C28	✓		✓	✓									✓	✓							✓		✓	
C29	✓	✓	✓															✓		✓			✓	✓
C30	✓			✓					✓				✓	✓		✓								
C31			✓	✓	✓	✓					✓		✓	✓		✓	✓			✓	✓	✓	✓	✓
C32	✓	✓	✓	✓							✓												✓	
C33	✓		✓	✓														✓					✓	✓
C34			✓	✓													✓					✓		
C35	✓			✓	✓		✓		✓		✓		✓	✓	✓		✓	✓			✓			✓
C36		✓																			✓	✓	✓	
C37	✓	✓	✓	✓		✓				✓	✓									✓	✓		✓	

Legend:

M01 - Mean Time To Recover; M02 - Mean Lead-time for Changes; M03 - Deployment Frequency; M04 - Change Failure Rate; M05 - Service Availability and Uptime; M06 - Deployment duration time; M07 - Mean Time To Detection; M08 - Application response time; M09 - Defect escape rate; M10 - Cycle Time Value; M11 - SLAs and SLOs; M12 - Deployment size; M13 - Production Error and Incident rate; M14 - Customer tickets Volume and Feedback; M15 - Mean time to failure; M16 - Customer Usage and traffic; M17 - Pipeline automated tests success/fail rate; M18 - Westrum organizational culture measures; M19 - Automated Test Code Coverage; M20 - Work in Progress/Load; M21 - Unplanned Work Rate; M22 - Wait Time; M42 - Team Happiness; M59 - Talent retention.

C01 - Cross team collaboration and communication; C02 - Continuous Integration; C03 - Continuous Delivery and Deployment automation; C04 - Proactive Monitoring; Observability and autoscaling; C05 - Test Automation and environments; C06 - Continuous Improvement of processes and workflows; C07 - Version Control System; C08 - Support learning culture and experimentation; C09 - Empower teams to make decisions and changes; C10 - Focus on people; process and technology; C11 - Configuration Management; C12 - Cloud infrastructure and cloud native; C13 - Artifacts versioning and registry; C14 - Loosely coupled architecture; C15 - Database change management; C16 - Infrastructure as Code; C17 - Emergency response; C18 - Containerization; C19 - Open source software adoption; C20 - Shift left on security; C21 - Transformational leadership; C22 - Trunk based development; C23 - Monitor systems to inform business decisions; C24 - Westrum organizational culture; C25 - Working in small batches; C26 - Centralized log management; C27 - Lightweight change approval; C28 - Visibility of work in the value stream; C29 - Working in progress limits; C30 - Customer feedback; C31 - Blameless Postmortems; C32 - Data-driven approach for improvements; C33 - Job satisfaction; C34 - Test data management; C35 - Chaos Engineering; C36 - Code maintainability; C37 - Visual management capabilities.

7.2 Proposed Capability Evaluation Matrix

The proposed DevOps capability evaluation matrix is an artifact that aims to be used for future strategic planning.

Table 7.6: Proposed artifact showing categorized DevOps capabilities influencing main metrics.

		Change KPI										Operating KPI									Cultural KPI			Business KPI	
		M02	M03	M04	M06	M09	M10	M12	M15	M17	M19	M01	M05	M07	M08	M11	M13	M20	M21	M22	M18	M42	M59	M14	M16
Cultural	C01	✓	✓	✓	✓	✓	✓					✓	✓	✓		✓					✓	✓	✓	✓	
	C08		✓				✓			✓					✓		✓				✓	✓	✓		
	C19	✓	✓				✓				✓									✓		✓	✓		
	C21	✓	✓	✓			✓					✓	✓								✓	✓	✓	✓	
	C24	✓	✓	✓					✓			✓		✓						✓	✓	✓	✓		
	C31		✓	✓	✓					✓			✓			✓	✓	✓	✓	✓		✓	✓	✓	✓
	C33		✓	✓								✓									✓	✓	✓		
Technical	C02	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓													
	C03	✓	✓	✓	✓		✓		✓			✓								✓	✓				
	C05	✓	✓	✓			✓			✓	✓						✓					✓		✓	✓
	C07	✓	✓								✓	✓					✓			✓		✓	✓		
	C09	✓	✓	✓	✓		✓		✓	✓		✓	✓				✓	✓		✓	✓	✓	✓	✓	
	C11	✓		✓							✓											✓	✓		
	C12	✓	✓		✓		✓					✓	✓		✓	✓			✓	✓		✓			
	C13		✓					✓				✓		✓					✓	✓					
	C14	✓	✓	✓				✓	✓			✓			✓	✓		✓		✓				✓	
	C15		✓	✓		✓			✓			✓	✓				✓		✓			✓		✓	
	C16	✓			✓		✓	✓				✓	✓	✓				✓	✓	✓		✓			
	C18		✓	✓	✓			✓		✓	✓	✓	✓		✓	✓			✓			✓			
	C20	✓	✓	✓		✓						✓	✓	✓	✓			✓	✓			✓	✓	✓	
	C22	✓	✓	✓		✓	✓	✓				✓					✓		✓		✓	✓			
	C26			✓								✓	✓	✓			✓		✓	✓		✓		✓	
	C34		✓	✓						✓										✓					
	C35			✓		✓			✓	✓		✓	✓	✓		✓	✓		✓		✓		✓	✓	
	C36	✓																	✓	✓		✓			
Measurement	C04											✓	✓	✓	✓	✓	✓		✓						✓
	C17											✓	✓	✓	✓	✓		✓	✓	✓				✓	
	C23			✓						✓		✓	✓		✓		✓								
	C29	✓	✓									✓						✓			✓	✓	✓		
	C37	✓	✓	✓	✓		✓					✓				✓		✓	✓			✓			
Process	C06	✓	✓				✓					✓						✓	✓	✓	✓				
	C10	✓	✓																		✓	✓	✓	✓	
	C25	✓	✓		✓			✓								✓	✓	✓	✓	✓	✓			✓	
	C27	✓	✓				✓					✓									✓	✓			
	C28		✓	✓								✓					✓		✓			✓		✓	
	C30			✓		✓						✓					✓							✓	✓
	C32	✓	✓	✓								✓				✓						✓			

Legend:

M01 - Mean Time To Recover; M02 - Mean Lead-time for Changes; M03 - Deployment Frequency; M04 - Change Failure Rate; M05 - Service Availability and Uptime; M06 - Deployment duration time; M07 - Mean Time To Detection; M08 - Application response time; M09 - Defect escape rate; M10 - Cycle Time Value; M11 - SLAs and SLOs; M12 - Deployment size; M13 - Production Error and Incident rate; M14 - Customer tickets Volume and Feedback; M15 - Mean time to failure; M16 - Customer Usage and traffic; M17 - Pipeline automated tests success/fail rate; M18 - Westrum organizational culture measures; M19 - Automated Test Code Coverage; M20 - Work in Progress/Load; M21 - Unplanned Work Rate; M22 - Wait Time; M42 - Team Happiness; M59 - Talent retention.

C01 - Cross team collaboration and communication; C02 - Continuous Integration; C03 - Continuous Delivery and Deployment automation; C04 - Proactive Monitoring; Observability and autoscaling; C05 - Test Automation and environments; C06 - Continuous Improvement of processes and workflows; C07 - Version Control System; C08 - Support learning culture and experimentation; C09 - Empower teams to make decisions and changes; C10 - Focus on people; process and technology; C11 - Configuration Management; C12 - Cloud infrastructure and cloud native; C13 - Artifacts versioning and registry; C14 - Loosely coupled architecture; C15 - Database change management; C16 - Infrastructure as Code; C17 - Emergency response; C18 - Containerization; C19 - Open source software adoption; C20 - Shift left on security; C21 - Transformational leadership; C22 - Trunk based development; C23 - Monitor systems to inform business decisions; C24 - Westrum organizational culture; C25 - Working in small batches; C26 - Centralized log management; C27 - Lightweight change approval; C28 - Visibility of work in the value stream; C29 - Working in progress limits; C30 - Customer feedback; C31 - Blameless Postmortems; C32 - Data-driven approach for improvements; C33 - Job satisfaction; C34 - Test data management; C35 - Chaos Engineering; C36 - Code maintainability; C37 - Visual management capabilities.

8

Evaluation

Contents

The evaluation of the artifact follows the evaluation process in the design cycle mentioned in Section 4.1, based on the framework for evaluation in design science (FEDS) [293] driven by the work of Pries-Heje, Baaskerville and Venable, who published several related articles [294–296]. Per the authors, the evaluation of an Information System's artifact can be conducted before the production of the proposal, known as "ex ante" viewpoint, or after the artifact's construction, known as "ex post" perspective. Semi-structured interviews iterations are conducted, results are summarized, and a validated artifact is then presented.

8.1 Semi-structured Interviews Iterations

This evaluation involved conducting semi-structured interviews, which is relatively cheaper to evaluate with real users in their real context and also maintains the goal of having a rigorous evaluation, establishing that utility/benefit will continue in real and long-term situations. For this phase, it was targeted to people with various skills to collect different visions and ensure a greater completeness of the validated artifact.

Ten semi-structured interviews were done with practitioners listed and characterized in Table 8.1, in order to perform several assessment iterations, following the outline in Table A.2, with the objective of refining and validating the artifact proposed in Section 7.2. The preparation is similar to what is described in Section 7.1.1

Table 8.1: Semi-structured interview iterations for evaluation with practitioners' details.

ID	Interview Date	Current position	Company	Years of DevOps practice	Age (years)	Country
IT01	2021-06-18	Indvidual contributor	Acquia	More than 3 years	35-44	Romania
IT02	2021-06-18	Associate DevOps Engineer	Acquia	Less than 1 year	24 =<	India
IT03	2021-06-21	Associate DevOps Engineer	Acquia	Less than 1 year	24 =<	India
IT04	2021-06-21	Associate Engineer	Acquia	Between 1 year and 3 years	24 =<	India
IT05	2021-06-21	Indvidual contributor	Boxboat	More than 3 years	25-34	United States
IT06	2021-06-22	Architect	Acquia	More than 3 years	25-34	United States
IT07	2021-06-24	Indvidual contributor	Acquia India Pvt. Ltd.	More than 3 years	25-34	India
IT08	2021-06-24	Indvidual contributor	Acquia	Between 1 year and 3 years	35-44	United States
IT09	2021-06-29	VP	Acquia	Between 1 year and 3 years	35-44	United States
IT10	2021-07-02	Product Owner, Drupalista	Liip	More than 3 years	35-44	Switzerland

A total of 10 interviews were conducted in this section to evaluate the research proposal. The practitioners interviewed and seen in Table 8.1 were: seven individual contributors, one architect, one VP and one product owner. The companies they work for are: Acquia, Boxboat and Liip. Five practitioners have more than 3 years experience on DevOps, while three practitioners have between 1 year and 3 years experience, and two have less than one year experience. From all the 21, nineteen are individual contributors with diverse roles, six are Team Leads and other six are at management level or above. The

majority of 4 practitioners is between 25-34 years of age, three are between 35-44 and three have less than 24 years. Their locations are India, Romania, Switzerland and USA.

8.1.1 Evaluation Iterations

In this section, every practitioner was requested to go through the suggested research artifact while the interview followed the outline in Table A.2. The interviewees were incentivized to suggest recommendations for changing or adding the relationships that they would see fit, as well as double-check the metrics and capabilities categorization.
In interview IT01, it was asked to analyze the artifact and make suggestion for improvements of any missing relations. The practitioner mentioned and justified some extra metrics that are impacted positively by Trunk based development (C22) like MTTF, automated tests pass, Automation Code Coverage, and wait time. All seen in Table 8.2 and Table 8.3. It was also suggested that Continuous Improvement of processes and workflows (C06) improves customer feedback, keeping SLAs and SLOs, reliability and deployment speed. Furthermore, in terms of expenses and environments for CI, an organization needs the infrastructure to accomplish test and build, but the team can probably start by doing it on the desktop or in a virtual environment to save costs. For that, it was stated, that an open source operating system like GNU/Linux[1] is much better suited. Regarding releases, if things don't improve, by the end of the process an engineer is always caught up into something else and doesn't even have time to enjoy the fact that his work is in production because deployment takes too long. The practitioner concluded that the proposed categorization was sound.

For interview IT02 the same was proposed to the practitioner and Performance Oriented Culture (C24) was identified as highly important, since there will be a lot of confidence between the team and the leadership having a positive impact on deployment speed, automated tests pass, pipeline success rate, reliability, SLOs and customer feedback. Also mentioned as related to Continuous Improvement (C06), influencing cultural KPIs like team Happiness and talent retention. Businesses with a high-performance culture may consistently achieve high levels of performance and outcomes. Therefore, many companies place an effort on cultivating a high-performance culture, since it may be the difference between mediocrity and growth, efficiency and falling behind.

During interview IT03 the DevOps engineer focused in examining Performance organizational culture (C24) capability, impacting change volume, application performance, production errors rate and UWR and at the same time mentioning that the most creative companies and high-performing organizations are continuously seeking to improve and never consider themselves done with their development or transformational journey. Several other interesting aspects were mentioned like when there is technical depth closed that means, whatever the technical issues there are closed, organizational culture is the

[1] https://www.gnu.org/gnu/linux-and-gnu.en.html

enabler for that change. An important lesson learned is that growth requires a careful mix of challenge and nurture. Too much challenge, repeated too frequently, it's overwhelming and can lead to burnout. Too little challenge prevents us from growing and does not improve the needed outcomes.

In interview IT04 Centralized Dashboards for Monitoring systems to inform business decisions (C23) associated with mostly operating KPIs and customer feedback, which are influencing Visual management capabilities (C37) were the main focus of discussion settled on how this affects operational mindset, specially on reliability. It was also discussed what are the characteristics that we consider a metric to be an Operational KPI and what are the factors that we consider a metric to be a Change KPI. Dashboards improve decision-making and may streamline the entire decision-making process, saving time and decreasing paperwork. Improves participation and cooperation by giving everyone first-hand access to data and allowing them to witness the real-time outcomes of their efforts.

The practitioner in interview IT05 examined how empowering the team to make decisions and changes (C09) is affecting a few more change and operating KPIs and definitely improves all the cultural KPIs and customer feedback. Employees will only strengthen their decision-making skills via experience and learning, especially if the firm is experiencing rapid development. Things may not go as planned the first time, or even the second, which is where leading and mentoring is fundamental. Regardless of industry, business size, or location, high-performance cultures share several common traits. Transformational leadership and empowered teams are common examples. Continuous learning and experimentation, as well as an openness to change, are essential. Another approached was Visibility of work in the value stream (C28), a process capability that also improves CTV and wait time metrics. Finally, customer feedback has positive impact from C02 CI and C03 CD.

The engineering architect in interview IT06 is an expert on microservices and distributed systems and so contributed more to examining Loosely Coupled Architecture and Microservices (C14), where he argued the importance of adding deployment duration, reliability and a few other operating and cultural KPIs like team happiness and talent retention as well as customer traffic. Loose coupling is a design pattern that hides implementation details inside each microservice in this capacity. When components are loosely connected, systems have several advantages for the organization. Containerization (C18) was also briefly discussed, with the result of adding deployment speed to Table 8.3 and confirming others in the list of metrics impacted by this capability. Containerization is also a big benefit in continuous integration CI and continuous delivery CD: containers greatly simplify integration testing and standardize CI/CD through Docker images.

Examination in interview IT07 focused on Cloud Infrastructure and Cloud Native (C12) validating already existing or new change KPIs relations like MTTF and also operating KPIs production errors rate and incidents, WIP and customer usage. Moreover, traditional business apps, which are generally run in an on-premises data center, require an entirely different design than cloud-native applications. Security

should be shifted to the left and reduce risks to a minimum. Taking this strategy lowers the chances of suffering a data breach or security issue. Shift Left on security (C20) impacts all cultural KPIs and some of the change KPIs like defect escape rate, CTV, deployment size, also impacting operating KPIs wait time and organizational culture metrics. Finally, team happiness cultural KPI was also confirmed to be impacted by the existing relations in the matrix. Some indications of employee dissatisfaction are easy to identify, like low talent retention.

In interview IT08, three capabilities were assessed in depth. The first was Support learning culture and experimentation (C08) where change KPIs MLT, deployment speed, and work in progress were explained to have a strong impact while practicing this capability. The second, Transformational leadership (C21) was also confirmed to have and influence on customer feedback due to attention that these leaders are putting in all the life cycle process in order to improve it. Lastly, Lightweight change approval (C27) was mentioned to be missing an obvious impact relationship in wait time operating KPI. It was also stated the importance of visionary inspirational communication, personal recognition and intellectual stimulation by supportive leadership while adopting DevOps capabilities.

In interview IT09 the most evaluated capability was Database Change management (C15). For operating KPIs there were missing relations on SLAs / SLOs, wait time and cultural KPI talent retention. Database change management is one of a set of capabilities that helps organizations produce better software and perform better. It should be possible to track database changes and those should be implemented and reconciled as quickly as feasible, while still protecting data privacy inside the database context. In the end, talent retention cultural KPI was also stated that by personal experience when C15 improves there is not only team happiness rising but also fewer people leaving.

In the last evaluation interview IT10 the previous Database Change management (C15) relationships were confirmed also adding MTTD and in Shift Left on security (C20) adding related KPIs pipeline success rate and production error rate. Regarding managing database changes, it was stated to be a difficult process. It is much more challenging when just getting started on a project and the data model is continuously changing. There was also a valuable discussion about capabilities being multifaceted and dynamic, allowing different areas of the company to tailor their approach to improvement and focus on the skills that would bring them the most value.

8.1.2 Evaluation Results

In this section, the key results of the evaluation done with ten interviews iterations are summarized. Primarily it is seen in Table 8.2 that the additions or updates to the matrix have become residual, which confirms the data saturation already observed in Section 7.1.4.

A summary of all updated relations in each iteration can be found in Table 8.2, together with the total relations and the relative percentage of each change. There were residual relations suggested between

capabilities and metrics, but there were no suggestions for changing capabilities, metrics or categories, therefore not expressed in the table.

Table 8.2: Relations updated in the artifact during each evaluation iteration.

Evaluation	Change KPI										Operating KPI									Cultural KPI			Business KPI		Updated		
Iterations	M02	M03	M04	M06	M09	M10	M12	M15	M17	M19	M01	M05	M07	M08	M11	M13	M20	M21	M22	M18	M42	M59	M14	M16	Relations	Total	Percentage
IT01				C06				C22	C22	C22		C06			C06				C22				C06		8	340	0.90%
IT02				C24					C24			C24			C24						C06	C06	C24		7	347	0.79%
IT03							C24							C24		C24		C24							4	351	0.45%
IT04								C23				C37	C23		C23				C23				C23		6	357	0.68%
IT05					C09	C28	C09								C09			C09	C28				C02,C03		8	365	0.90%
IT06				C14				C18				C14	C14								C14	C14		C14	7	372	0.79%
IT07						C20	C20	C12								C12	C12		C20	C20				C12	8	380	0.90%
IT08	C08			C08													C08		C27					C21	5	385	0.56%
IT09															C15				C15			C15			3	388	0.34%
IT10									C20				C15			C20									3	391	0.34%

Legend:

M01 - Mean Time To Recover; M02 - Mean Lead-time for Changes; M03 - Deployment Frequency; M04 - Change Failure Rate; M05 - Service Availability and Uptime; M06 - Deployment duration time; M07 - Mean Time To Detection; M08 - Application response time; M09 - Defect escape rate; M10 - Cycle Time Value; M11 - SLAs and SLOs; M12 - Deployment size; M13 - Production Error and Incident rate; M14 - Customer tickets Volume and Feedback; M15 - Mean time to failure; M16 - Customer Usage and traffic; M17 - Pipeline automated tests success/fail rate; M18 - Westrum organizational culture measures; M19 - Automated Test Code Coverage; M20 - Work in Progress/Load; M21 - Unplanned Work Rate; M22 - Wait Time; M42 - Team Happiness; M59 - Talent retention.

C02 - Continuous Integration; C03 - Continuous Delivery and Deployment automation; C06 - Continuous Improvement of processes and workflows; C08 - Support learning culture and experimentation; C09 - Empower teams to make decisions and changes; C12 - Cloud infrastructure and cloud native; C14 - Loosely coupled architecture; C15 - Database change management; C18 - Containerization; C20 - Shift left on security; C21 - Transformational leadership; C22 - Trunk based development; C23 - Monitor systems to inform business decisions; C24 - Westrum organizational culture; C27 - Lightweight change approval; C28 - Visibility of work in the value stream; C37 - Visual management capabilities.

As can be seen, the changes to the artifact are now modest in contrast to the previous phase, indicating that the evaluation was successful and supporting the proposal's design [290].

8.2 Validated Artifact

Using the validated artifact here presented, business executives and organization leaders will be able to evaluate the various DevOps capabilities in relation to the outcomes, identifying the capabilities that require the most significant long-term enhancements and highlighting the capabilities that should become the primary focus of future IT investments as seen in Table 8.3.

As a few participants mentioned, this will also allow the organization to stimulate discussions about how each capability impacts DevOps adoption, allowing them to figure out where they can get the most value from. Like what capabilities, if included from the start, would have the most influence on a variety of KPIs.

This capability matrix is outcome-based, focusing on important outcomes (KPIs) and how capabilities promote change in those outcomes. This gives clear guidance and strategy on high-level goals (with an emphasis on capabilities to enhance key outcomes) to technical leadership. It also allows team leaders and individual contributors to define progress targets for the current time period based on the capabilities their team is working on.

Table 8.3: Validated artifact with categorized DevOps capabilities influencing main metrics.

		Change KPI										Operating KPI									Cultural KPI			Business KPI	
		M02	M03	M04	M06	M09	M10	M12	M15	M17	M19	M01	M05	M07	M08	M11	M13	M20	M21	M22	M18	M42	M59	M14	M16
Cultural	C01	✓	✓	✓	✓	✓	✓					✓	✓	✓		✓					✓	✓	✓	✓	
	C08	✓	✓		✓		✓			✓					✓		✓	✓			✓	✓	✓		
	C19	✓	✓				✓				✓									✓		✓	✓		
	C21	✓	✓	✓			✓					✓	✓								✓	✓	✓	✓	✓
	C24	✓	✓	✓	✓			✓	✓	✓		✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	
	C31		✓	✓	✓					✓			✓			✓	✓	✓	✓	✓		✓	✓	✓	✓
	C33		✓	✓								✓									✓	✓	✓		
Technical	C02	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓												✓	
	C03	✓	✓	✓	✓		✓		✓			✓								✓	✓			✓	
	C07	✓	✓								✓	✓					✓			✓		✓	✓		
	C09	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	
	C11	✓		✓							✓											✓	✓		
	C12	✓	✓		✓		✓		✓			✓	✓		✓	✓	✓	✓	✓	✓		✓			✓
	C13		✓					✓				✓		✓					✓	✓					
	C14	✓	✓	✓	✓			✓	✓			✓	✓	✓	✓	✓		✓		✓		✓	✓	✓	✓
	C15		✓	✓		✓			✓			✓	✓	✓		✓	✓		✓	✓		✓	✓	✓	
	C16	✓			✓		✓	✓				✓	✓	✓				✓	✓	✓		✓			
	C18		✓	✓	✓			✓	✓	✓	✓	✓	✓		✓	✓			✓			✓			
	C20	✓	✓	✓		✓	✓	✓		✓		✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	
	C22	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓					✓		✓	✓	✓	✓			
	C26			✓								✓	✓	✓			✓		✓	✓		✓		✓	
	C34		✓	✓						✓										✓					
	C35			✓		✓			✓	✓		✓	✓	✓		✓	✓		✓		✓		✓	✓	
	C36	✓																	✓	✓		✓			
Measurement	C04											✓	✓	✓	✓	✓	✓		✓						✓
	C17											✓	✓	✓	✓	✓		✓	✓	✓				✓	
	C23			✓					✓	✓		✓	✓	✓	✓	✓	✓			✓				✓	
Process	C29	✓	✓									✓						✓			✓	✓	✓		
	C37	✓	✓	✓	✓		✓					✓	✓			✓		✓	✓			✓			
	C06	✓	✓		✓		✓					✓	✓			✓		✓	✓	✓	✓	✓	✓	✓	
	C10	✓	✓																		✓	✓	✓	✓	
	C25	✓	✓		✓			✓								✓	✓	✓	✓	✓	✓			✓	
	C27	✓	✓				✓					✓								✓	✓	✓			
	C28		✓	✓			✓					✓					✓		✓	✓		✓		✓	
	C30			✓		✓						✓					✓							✓	✓
	C32	✓	✓	✓								✓				✓						✓			

Legend:

✓ : Updated relation.

M01 - Mean Time To Recover; M02 - Mean Lead-time for Changes; M03 - Deployment Frequency; M04 - Change Failure Rate; M05 - Service Availability and Uptime; M06 - Deployment duration time; M07 - Mean Time To Detection; M08 - Application response time; M09 - Defect escape rate; M10 - Cycle Time Value; M11 - SLAs and SLOs; M12 - Deployment size; M13 - Production Error and Incident rate; M14 - Customer tickets Volume and Feedback; M15 - Mean time to failure; M16 - Customer Usage and traffic; M17 - Pipeline automated tests success/fail rate; M18 - Westrum organizational culture measures; M19 - Automated Test Code Coverage; M20 - Work in Progress/Load; M21 - Unplanned Work Rate; M22 - Wait Time; M42 - Team Happiness; M59 - Talent retention.

C01 - Cross team collaboration and communication; C02 - Continuous Integration; C03 - Continuous Delivery and Deployment automation; C04 - Proactive Monitoring; Observability and autoscaling; C05 - Test Automation and environments; C06 - Continuous Improvement of processes and workflows; C07 - Version Control System; C08 - Support learning culture and experimentation; C09 - Empower teams to make decisions and changes; C10 - Focus on people; process and technology; C11 - Configuration Management; C12 - Cloud infrastructure and cloud native; C13 - Artifacts versioning and registry; C14 - Loosely coupled architecture; C15 - Database change management; C16 - Infrastructure as Code; C17 - Emergency response; C18 - Containerization; C19 - Open source software adoption; C20 - Shift left on security; C21 - Transformational leadership; C22 - Trunk based development; C23 - Monitor systems to inform business decisions; C24 - Westrum organizational culture; C25 - Working in small batches; C26 - Centralized log management; C27 - Lightweight change approval; C28 - Visibility of work in the value stream; C29 - Working in progress limits; C30 - Customer feedback; C31 - Blameless Postmortems; C32 - Data-driven approach for improvements; C33 - Job satisfaction; C34 - Test data management; C35 - Chaos Engineering; C36 - Code maintainability; C37 - Visual management capabilities.

9

Conclusion

Contents

In this chapter, we conclude the research done with communication, general conclusions, limitations and future work.

9.1 Communication

Out of the research already done, the two MLRs have been submitted for approval and publication in two different journals with Q1 rank. The MLR on DevOps capabilities has been submitted for publication in *Transactions on Software Engineering* (IEEE, h-index:169) and was recommended by the editors to undergo a Major Revision. The paper is being revised, and a new revision will be re-submitted soon. The MLR on DevOps metrics was submitted to *Information Processing and Management* (Elsevier, h-index:101) on 26th September 2021.

9.2 Research Conclusions

This study has brought important contributions to both academia and industry on the DevOps topic. In summary, a Design Science research was done that includes two MLRs on DevOps capabilities, in DevOps metrics and 21 semi-structured interviews in the build phase. To find literature, Google search, Scopus, Web of Science, IEEE, ACM, and EBSCO were utilized, and after applying the inclusion and exclusion criteria and snowballing, 207 papers were identified as relevant to these study topics. In order to evaluate the research proposal, 10 semi-structured interviews were done, resulting in a validated Capability Evaluation Matrix.

- This study proposes a consensus definition distinguishing capabilities from practices, based on academic and industry literature review.
- A thorough investigation was conducted in order to find a consensus definition of DevOps metrics across academics and practitioners.
- It was investigated and exposed where the capabilities and practices are mentioned in literature (RQ2) and what are their differences seen (RQ3).
- The purpose of each metric is identified in detail (RQ5) and the reason why each metric is important is analyzed (RQ6).
- From all the literature review done in this research, 37 DevOps capabilities (RQ1) and 24 validated DevOps metrics (RQ4) were identified.
- DevOps capabilities have been researched, explained and after a careful evaluation categorized (RQ7). The same rigorous work was conducted for DevOps metrics (RQ8).

- In order to develop a core study proposal, this investigation drafted the DevOps capabilities that have a beneficial influence in each of the key metrics (RQ9).

- The major outcome of this research is a Capability Evaluation Matrix presented in Section 8.2, which has been proposed, debated, and validated by DevOps practitioners.

A set of interesting conclusions arise when identifying the analysis vector that connects the capability categories to the KPI categories based on the validated artifact in Table 8.3. In Table 9.1 the most evident conclusion is that cultural capabilities have a strong impact for improving the cultural KPIs and the highest overall impact.

Table 9.1: Capability categories weight on KPI categories.

	Change KPIs	Operating KPIs	Cultural KPIs	Business KPIs	Total AVG
Cultural capabilities	45.71%	39.68%	90.48%	42.86%	54.68%
Technical capabilities	49.44%	61.90%	46.30%	33.33%	47.74%
Measurement capabilities	20.00%	64.44%	26.67%	30.00%	35.28%
Process capabilities	30.00%	33.33%	52.38%	42.86%	39.64%

Nevertheless, operating KPIs are mostly impacted by technical and measurement capabilities. However, process capabilities are the second most impactful on cultural KPIs and when looking at what are the main drivers of change those are technical, cultural and process capabilities. As a result, organizations should prioritize not only technology, but also cultural and process improvements. A DevOps capability is defined as the ability to do perform a DevOps practice or by the quality, or state of being capable. These capabilities are, dynamic and have been growing and changing over the years, being defined by the ability of an organization to perform DevOps practices, or by the quality, or state of being capable. The two capabilities with most relations to metrics are, in order, C09-Empower teams to make decisions and changes and C24-Performance organizational culture.

A DevOps metric is defined as a quantifiable, business-relevant, trustworthy, actionable and traceable indicator that aids organizations in making data-driven decisions to continuously improve their software delivery process. The five top metrics with most of the relations in the model, that an organization should start by measuring are, in order, M03-DF, M01-MTTR, M42-Team happiness, M02-MLT and M04-CFR. Metrics have been extended to 24, divided into four KPI categories: change, operating, cultural and business.

Finally, it was also perceived that releasing software with both speed and stability is achievable if the company is continuously monitoring the appropriate metrics and improving the right capabilities by focusing on the outcomes, rather than just following a prescribed path for each team.

9.3 Limitations

Identified limitations of this study include the fact that it is based on MLR, therefore, a part of the material has not gone through the critical peer-review process that academic research is typically exposed to. To mitigate the impact of this danger, it was chosen to design the review procedure using the recommendations given by Garousi et al. [3] and to conduct each step using this method.

Semi-structured interviews used in this research usually require a large enough variety sample to yield precision and variety of opinions. To address this problem 31 interviews were conducted [297]. The use of search terms and search engines may result in an inadequate selection of primary materials. Formal searches were conducted using particular keywords, and specific source code was used to decrease the chance of missing all relevant studies and increase the dependability of repeating this study.

For the year 2021, this research was limited to only three months. Although the year 2021 is present, it is irrelevant for the extraction of the data because the three-month analysis is insignificant. Lastly, the inclusion of English-only publications, which may exclude relevant research in other languages, was a limitation.

9.4 Future Work

What was revealed in this study will assist to feed new research so that future studies can evaluate if certain metrics are still common and should be explored further researching the possible ways to put these capabilities, metrics and relations into practice.

The relationship matrix used in the artifact may also be expanded to indicate whether the influence or impact on the metric can be verified as positive or negative. Based on this, it should be possible to produce a heat map for a capability model that would be easier to use in management decisions.

At the organizational level, we still don't know which measures are already being utilized by which industries. What other organizational factors have the most influence on each of the main, important metrics and if we can influence these factors, or how? Is it possible to expose the metrics, capabilities and their influencing factors in information systems in order to support management decisions [11]?

In contrast, there is still debate going on [100,203,204,238] regarding: Should all of these indicators be tracked regularly? Which metrics are capable of being monitored automatically? Which metrics can only be monitored using surveys? All are interesting questions to investigate.

Lastly, as demonstrated in Section 6.3, better monitoring the software delivery process is extremely significant and sought. DevOps metrics should try to measure efficiently the right aspects in order to determine whether DevOps is effective.

Bibliography

[1] Gene Kim, “The Three Ways: The Principles Underpinning DevOps,” 2012. [Online]. Available: https://itrevolution.com/the-three-ways-principles-underpinning-devops/

[2] A. R. Hevner, S. T. March, J. Park, and S. Ram, “Design science in information systems research,” *MIS Quarterly: Management Information Systems*, vol. 28, no. 1, pp. 75–105, 2004.

[3] V. Garousi, M. Felderer, and M. V. Mäntylä, “Guidelines for including grey literature and conducting multivocal literature reviews in software engineering,” *Information and Software Technology*, vol. 106, no. September 2018, pp. 101–121, feb 2019.

[4] M. Senapathi, J. Buchan, and H. Osman, “DevOps Capabilities, Practices, and Challenges,” in *Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018 - EASE’18*, ser. EASE’18, no. June. New York, USA: ACM Press, jun 2018, pp. 57–67.

[5] N. Forsgren, J. Humble, and G. Kim, *Accelerate: The science of lean software and devops: Building and scaling high performing technology organizations*. IT Revolution, 2018.

[6] I. T. Revolution, “24 Key Capabilities to Drive Improvement in Software Delivery - IT Revolution,” 2020. [Online]. Available: https://itrevolution.com/24-key-capabilities-to-drive-improvement-in-software-delivery/

[7] Google, “DevOps capabilities | DORA - Google Cloud,” 2020. [Online]. Available: https://cloud.google.com/solutions/devops/capabilities

[8] DORA, “DORA research program,” 2020. [Online]. Available: https://www.devops-research.com/research.html

[9] S. Mäkinen, M. Leppänen, T. Kilamo, A.-L. Mattila, E. Laukkanen, M. Pagels, and T. Männistö, “Improving the delivery cycle: A multiple-case study of the toolchains in Finnish software intensive enterprises,” *Information and Software Technology*, vol. 80, pp. 175–194, dec 2016.

[10] N. Forsgren, M. C. Tremblay, D. VanderMeer, and J. Humble, "DORA Platform: DevOps Assessment and Benchmarking," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Springer, 2017, pp. 436–440.

[11] J. P. L. . K. C. Laudon, *Management Information Systems: Managing the Digital Firm, Global Edition*, 15th ed. Pearson Education, 2017.

[12] A. Dyck, R. Penners, and H. Lichter, "Towards definitions for release engineering and DevOps," in *Proceedings - 3rd International Workshop on Release Engineering, RELENG 2015*. IEEE Press, may 2015, p. 3.

[13] J. Wettinger, U. Breitenbücher, and F. Leymann, "DevOpSlang – Bridging the Gap between Development and Operations," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. IFIP, 2014, vol. 8745 LNCS, pp. 108–122. [Online]. Available: http://link.springer.com/10.1007/978-3-662-44879-3_8

[14] N. Forsgren and J. Humble, "The Role of Continuous Delivery in it and Organizational Performance," *SSRN Electronic Journal*, pp. 1–15, 2015. [Online]. Available: http://www.ssrn.com/abstract=2681909

[15] J. Humble and D. Farley, *Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation*. Addison-Wesley Professional, dec 2010.

[16] Puppet Labs, "2015 State of DevOps Report," Puppet Labs, Tech. Rep. 877, 2015. [Online]. Available: http://puppetlabs.com/2015-devops-report

[17] G. Kim, K. Behr, K. Spafford, and G. Spafford, *The phoenix project: A novel about IT, DevOps, and helping your business win*. IT Revolution, 2014. [Online]. Available: https://books.google.pt/books?id=H6x-DwAAQBA

[18] U. S. Bititci, P. Garengo, A. Ates, and S. S. Nudurupati, "Value of maturity models in performance measurement," *International Journal of Production Research*, vol. 53, no. 10, pp. 3062–3085, may 2015.

[19] N. Forsgren and M. Kersten, "DevOps Metrics: Your biggest mistake might be collecting the wrong data." *Queue*, vol. 15, no. 6, pp. 19–34, dec 2017.

[20] J. Smeds, K. Nybom, and I. Porres, "DevOps: A Definition and Perceived Adoption Impediments," in *Lecture Notes in Business Information Processing*. Springer, 2015, vol. 212, pp. 166–177.

[21] R. Pereira and J. Serrano, “A review of methods used on IT maturity models development: A systematic literature review and a critical analysis,” *Journal of Information Technology*, vol. 35, no. 2, pp. 161–178, jun 2020.

[22] R. Jabbari, N. bin Ali, K. Petersen, and B. Tanveer, “What is DevOps?” in *Proceedings of the Scientific Workshop Proceedings of XP2016*, ser. XP ’16 Workshops. New York, NY, USA: ACM, may 2016, pp. 1–11.

[23] J. Humble and J. Molesky, “Why enterprises must adopt devops to enable continuous delivery,” *Cutter IT Journal*, vol. 24, no. 8, pp. 6–12, 2011.

[24] J. Willis, “DevOps Culture (Part 1) - IT Revolution,” 2012. [Online]. Available: https://itrevolution.com/devops-culture-part-1/

[25] L. Sousa, A. Trigo, and J. Varajão, “DevOps – foundations and perspectives,” 2019. [Online]. Available: https://www.researchgate.net/publication/339311150_DevOps_-_fundamentos_e_perspetivas

[26] B. Tessem and J. Iden, “Cooperation between developers and operations in software engineering projects,” *Proceedings - International Conference on Software Engineering*, pp. 105–108, 2008.

[27] L. E. Lwakatare, T. Kilamo, T. Karvonen, T. Sauvola, V. Heikkilä, J. Itkonen, P. Kuvaja, T. Mikkonen, M. Oivo, C. Lassenius, V. Heikkila, J. Itkonen, P. Kuvaja, T. Mikkonen, M. Oivo, and C. Lassenius, “DevOps in practice: A multiple case study of five companies,” *Information and Software Technology*, vol. 114, no. March 2017, pp. 217–230, oct 2019.

[28] L. Riungu-Kalliosaari, S. Mäkinen, L. E. Lwakatare, J. Tiihonen, and T. Männistö, “DevOps adoption benefits and challenges in practice: A case study,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10027 LNCS, pp. 590–597, 2016.

[29] S. J. Wu, S. A. Melnyk, and B. B. Flynn, “Operational Capabilities: The Secret Ingredient,” *Decision Sciences*, vol. 41, no. 4, pp. 721–754, 2010.

[30] K. M. Eisenhardt and J. A. Martin, “Dynamic capabilities: What are they?” *Strategic Management Journal*, vol. 21, no. 10-11, pp. 1105–1121, 2000.

[31] G. Cepeda and D. Vera, “Dynamic capabilities and operational capabilities: A knowledge management perspective,” *Journal of Business Research*, vol. 60, no. 5, pp. 426–437, 2007.

[32] J. F. Henri, “Management control systems and strategy: A resource-based perspective,” *Accounting, Organizations and Society*, vol. 31, no. 6, pp. 529–558, 2006.

[33] J. Karimi and Z. Walter, “The role of dynamic capabilities in responding to digital disruption: A factor-based study of the newspaper industry,” *Journal of Management Information Systems*, vol. 32, no. 1, pp. 39–81, 2015.

[34] F. Erich, C. Amrit, M. Daneva, A. Jedlitschka, P. Kuvaja, M. Kuhrmann, T. Männistö, J. Münch, and M. Raatikainen, “Report: DevOps Literature Review,” University of Twente, Cham, Tech. Rep. October, 2014.

[35] F. Erich, C. Amrit, and M. Daneva, “A mapping study on cooperation between information system development and operations,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8892, no. 1, pp. 277–280, 2014.

[36] F. M. A. Erich, C. Amrit, and M. Daneva, “A qualitative study of DevOps usage in practice,” *Journal of Software: Evolution and Process*, vol. 29, no. 6, p. e1885, 2017.

[37] L. Prates, J. Faustino, M. Silva, and R. Pereira, “DevSecOps Metrics,” in *Lecture Notes in Business Information Processing*, M. J. Wrycza S., Ed. Instituto Universitário de Lisboa (ISCTE-IUL), Lisbon, Portugal: Springer, 2019, vol. 359, pp. 77–90.

[38] D. Budgen and P. Brereton, “Performing systematic literature reviews in software engineering,” in *Proceedings of the 28th international conference on Software engineering*, Keele University and Durham University Joint Report. New York, NY, USA: ACM, may 2006, pp. 1051–1052.

[39] J. F. Pérez, W. Wang, and G. Casale, “Towards a DevOps approach for software quality engineering,” in *WOSP-C 2015 - Proceedings of the 2015 ACM/SPEC Workshop on Challenges in Performance Methods for Software Development, in Conjunction with ICPE 2015*, ser. WOSP ’15. New York, New York, USA: ACM Press, 2015, pp. 5–10. [Online]. Available: https://doi.org/10.1145/2693561.2693564

[40] A. D. Nagarajan and S. J. Overbeek, “A DevOps Implementation Framework for Large Agile-Based Financial Organizations,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, P. H.A., M. R., A. C.A., P. H., D. C., and R. D., Eds. Department of Information and Computing Sciences, Utrecht University, Utrecht, Netherlands: Springer Verlag, 2018, vol. 11229 LNCS, pp. 172–188. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85055814441&doi=10.1007%2F978-3-030-02610-3_10&partnerID=40&md5=0773fd0a92908313a45a0c93448915e8

[41] N. Forsgren, M. A. Rothenberger, J. Humble, J. B. Thatcher, and D. Smith, “A taxonomy of software delivery performance profiles: Investigating the effects of devops practices,”

26th Americas Conference on Information Systems, AMCIS 2020, pp. 0–5, 2020. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85097711091&partnerID=40&md5=7c2dbc3859bd41e964feac16170bd302

[42] S. Abdelkebir, Y. Maleh, and M. Belaissaoui, "An Agile Framework for ITS Management In Organizations," in *Proceedings of the 2nd International Conference on Computing and Wireless Communication Systems - ICCWCS'17*, ser. ICCWCS'17. New York, New York, USA: ACM Press, 2017, pp. 1–8.

[43] A. Bertolino, G. D. Angelis, A. Guerriero, B. Miranda, R. Pietrantuono, and S. Russo, "DevOpRET: Continuous reliability testing in DevOps," *Journal of Software: Evolution and Process*, jul 2020. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85088248050&doi=10.1002%2Fsmr.2298&partnerID=40&md5=7e6aa3c834bb767e1422efdaf28f9791

[44] A. Mishra and Z. Otaiwi, "DevOps and software quality: A systematic mapping," *Computer Science Review*, vol. 38, no. 1, p. 100308, nov 2020.

[45] D. Marijan and S. Sen, "Devops enhancement with continuous test optimization," in *Proceedings of the International Conference on Software Engineering and Knowledge Engineering, SEKE*, vol. 2018-July. Simula, Norway: Knowledge Systems Institute Graduate School, 2018, pp. 536–541. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85056906246&doi=10.18293%2FSEKE2018-168&partnerID=40&md5=b53bb414499c946d9f6f8cc1f52df98d

[46] J. Angara, S. Gutta, and S. Prasad, "DevOps with continuous testing architecture and its metrics model," in *Advances in Intelligent Systems and Computing*. K.L. University, Vijayawada, AP, India: Springer Verlag, 2018, vol. 709, pp. 271–281. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85056272895&doi=10.1007%2F978-981-10-8633-5_28&partnerID=40&md5=4b68a8768d2bed014baabb78fdff8e52

[47] W. Pourmajidi, A. Miranskyy, J. Steinbacher, T. Erwin, and D. Godwin, "Dogfooding: Using IBM cloud services to monitor IBM cloud infrastructure," *CASCON 2019 Proceedings - Conference of the Centre for Advanced Studies on Collaborative Research - Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering*, pp. 344–353, jul 2020. [Online]. Available: http://arxiv.org/abs/1907.06094

[48] M. M. A. Ibrahim, S. M. Syed-Mohamad, and M. H. Husin, "Managing Quality Assurance Challenges of DevOps through Analytics," in *Proceedings of the 2019 8th International Conference on Software and Computer Applications*, ser. ICSCA '19, vol. Part F1479. New York, NY, USA: ACM, feb 2019, pp. 194–198.

[49] D. Sun, M. Fu, L. Zhu, G. Li, and Q. Lu, "Non-Intrusive Anomaly Detection with Streaming Performance Metrics and Logs for DevOps in Public Clouds: A Case Study in AWS," *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 2, pp. 278–289, apr 2016. [Online]. Available: http://search.ebscohost.com/login.aspx?direct=true&db=eoah&AN=39385533&lang=pt-pt&site=ehost-live&scope=site

[50] D. A. Tamburri, D. Di Nucci, L. Di Giacomo, and F. Palomba, "Omniscient DevOps Analytics," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, M. M. B. J.-M. Meyer B. Meyer B., Ed. Springer International Publishing, 2019, vol. 11350 LNCS, pp. 48–59.

[51] J. Díaz, J. E. Pérez, M. A. Lopez-Peña, G. A. Mena, and A. Yagüe, "Self-service cybersecurity monitoring as enabler for DevSecops," *IEEE Access*, vol. 7, no. 1, pp. 100 283–100 295, 2019.

[52] Y. Wang, M. V. Mäntylä, S. Demeyer, K. Wiklund, S. Eldh, and T. Kairi, "Software test automation maturity: A survey of the state of the practice," *ICSOFT 2020 - Proceedings of the 15th International Conference on Software Technologies*, pp. 27–38, 2020. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85091748062&partnerID=40&md5=671c7ffe725f4a1b7229a9c500188c51https://doi.org/10.5220/0009766800270038

[53] Z. Ding, J. Chen, and W. Shang, "Towards the use of the readily available tests from the release pipeline as performance tests." in *Proceedings - International Conference on Software Engineering*, ser. ICSE '20. New York, NY, USA: ACM, jun 2020, pp. 1435–1446.

[54] B. Snyder and B. Curtis, "Using Analytics to Guide Improvement During an Agile/DevOps Transformation," *IEEE Software*, 2018.

[55] S. T. March and G. F. Smith, "Design and natural science research on information technology," *Decision Support Systems*, vol. 15, no. 4, pp. 251–266, dec 1995.

[56] C. Sonnenberg and J. vom Brocke, "Evaluation Patterns for Design Science Research Artefacts," in *Practical Aspects of Design Science*, 2012, vol. 286, no. Chapter 7, pp. 71–83.

[57] A. R. Hevner, "A Three Cycle View of Design Science Research," *Scandinavian Journal of Information Systems*, vol. 19, no. 2, pp. 87–92, 2007.

[58] B. J. Oates, *Researching Information Systems and Computing*. Sage Publications Ltd., 2006. [Online]. Available: https://dl.acm.org/doi/book/10.5555/1202299

[59] R. L. Baskerville, M. Kaul, and V. C. Storey, "Genres of Inquiry in Design-Science Research: Justification and Evaluation of Knowledge Production," *MIS Quarterly*, vol. 39, no. 3, pp. 541–564, mar 2015.

[60] K. Peffers, T. Tuunanen, Charles E Gengler, Matti Rossi, Wendy Hui, Ville Virtanen, and Johanna Bragge, "The Design Science Research Process: A Model For Producing and Presenting information Systems Research," *First International Conference on Design Science Research in Information Systems and Technology*, pp. 83–16, 2006.

[61] M. Sánchez-Gordón and R. Colomo-Palacios, "A Multivocal Literature Review on the use of DevOps for e-learning systems," *ACM International Conference Proceeding Series*, pp. 883–888, 2018.

[62] B. B. Nicolau de França, H. Jeronimo, and G. H. Travassos, "Characterizing DevOps by hearing multiple voices," in *ACM International Conference Proceeding Series*, ser. SBES '16, E. DeAlmeida, Ed., Unicesumar; Colivre; Espweb; Tasa Eventos. New York, USA: Assoc Computing Machinery, 2016, Proceedings Paper, pp. 53–62.

[63] R. T. Ogawa and B. Malen, "Towards Rigor in Reviews of Multivocal Literatures: Applying the Exploratory Case Study Method," *Review of Educational Research*, vol. 61, no. 3, pp. 265–286, sep 1991.

[64] V. Garousi, M. Felderer, and M. V. Mäntylä, "The need for multivocal literature reviews in software engineering," in *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering - EASE '16*. New York, New York, USA: ACM Press, 2016, pp. 1–6.

[65] K. Raworth, C. Sweetman, S. Narayan, J. Rowlands, and A. Hopkins, *Conducting semi-structured Interviews*. Oxfam, 2012.

[66] W. C. Adams, "Conducting Semi-Structured Interviews," in *Handbook of Practical Program Evaluation: Fourth Edition*. Hoboken, NJ, USA: John Wiley and Sons, Inc., oct 2015, pp. 492–505.

[67] D. Cohen and B. Crabtree, "Semi-structured Interviews," 2006. [Online]. Available: http://www.qualres.org/HomeSemi-3629.html

[68] C. M. Conway and K. Roulston, "Conducting and Analyzing Individual Interviews," *The Oxford Handbook of Qualitative Research in American Music Education*, no. June, pp. 1–24, 2014.

[69] BMC, "State of DevOps 2020: A Report Roundup," *BMC*, 2020. [Online]. Available: https://www.bmc.com/blogs/state-of-devops/

[70] J. Mitlöhner, S. Neumaier, J. Umbrich, and A. Polleres, "Characteristics of open data CSV files," in *2016 2nd International Conference on Open and Big Data (OBD)*. IEEE, 2016, pp. 72–79.

[71] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," *ACM International Conference Proceeding Series*, 2014.

[72] Puppet Labs, “2013 State of DevOps Report,” Puppet Labs, Tech. Rep., 2013.

[73] ——, “2014 State of DevOps Report,” Puppet Labs, Tech. Rep., 2014. [Online]. Available: http://puppetlabs.com/2014-devops-report

[74] A. Novak, “Six Core Capabilities of a DevOps Practice – The New Stack,” 2014. [Online]. Available: https://thenewstack.io/six-core-capabilities-of-a-devops-practice/

[75] D. Russo, P. P. H. Hanel, S. Altnickel, and N. van Berkel, “The Daily Life of Software Engineers during the COVID-19 Pandemic,” *Proceedings of the 43th International Conference on Software Engineering*, jan 2021.

[76] G. Kim, J. Humble, P. Debois, and J. Willis, *The DevOps Handbook : How to Create World-Class Agility, Reliability, and Security in Technology Organizations*. IT Revolution Press, 2016.

[77] R. Westrum, “A typology of organisational cultures,” *Quality and Safety in Health Care*, vol. 13, no. SUPPL. 2, pp. 22–27, 2004. [Online]. Available: www.qshc.com

[78] T. F. Düllmann, C. Paule, and A. Van Hoorn, “Exploiting devops practices for dependable and secure continuous delivery pipelines,” in *Proceedings - International Conference on Software Engineering*, IEEE Comp Soc; Assoc Comp Machinery; SIGSOFT; IEEE Tech Council Software Engn. New York, NY, USA: Association for Computing Machinery, 2018, Proceedings Paper, pp. 27–30.

[79] C. S. Tov, “DevOps Best Practices: Take Your Pipeline to the Next Level - Codemotion,” 2020. [Online]. Available: https://www.codemotion.com/magazine/dev-hub/devops-engineer/devops-best-practices/

[80] M. Zulfahmi Toh, S. Sahibuddin, and M. N. Mahrin, “Adoption issues in DevOps from the perspective of continuous delivery pipeline,” in *ACM International Conference Proceeding Series*, ser. ICSCA ’19, vol. Part F1479. New York, NY, USA: Association for Computing Machinery, 2019, pp. 173–177.

[81] L. Yin and V. Filkov, “Team Discussions and Dynamics during DevOps Tool Adoptions in OSS Projects,” in *Proceedings - 2020 35th IEEE/ACM International Conference on Automated Software Engineering, ASE 2020*, ser. ASE ’20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 697–708.

[82] R. Mao, H. Zhang, Q. Dai, H. Huang, G. Rong, H. Shen, L. Chen, and K. Lu, “Preliminary Findings about DevSecOps from Grey Literature,” in *2020 IEEE 20th International Conference on Software Quality, Reliability and Security (QRS)*. IEEE, dec 2020, pp. 450–457.

[83] I. Karamitsos, S. Albarhami, and C. Apostolopoulos, “Applying devops practices of continuous automation for machine learning,” *Information (Switzerland)*, vol. 11, no. 7, pp. 1–15, jul 2020.

[84] Puppet Labs, “2019 State of DevOps Report,” Puppet Labs, Tech. Rep., 2019.

[85] M. A. Silva, J. P. Faustino, R. Pereira, and M. Mira da Silva, “Productivity gains of DevOps adoption in an IT team: A case study,” in *Proceedings of the 27th International Conference on Information Systems Development: Designing Digitalization, ISD 2018*, B. C. L. M. L. H. S. C. Andersson B. Johansson B., Ed. Association for Information Systems, jul 2018, p. 12. [Online]. Available: https://repositorio.iscte-iul.pt/handle/10071/16388

[86] I. Pavlenko, “DevOps: Principles, Practices, and DevOps Engineer Role,” 2021. [Online]. Available: https://www.altexsoft.com/blog/engineering/devops-principles-practices-and-devops-engineer-role/

[87] L. Bass, Len; Weber, Ingo; Zhu, *DevOps: A Software Architect’s Perspective*, 1st ed. Addison-Wesley Professional, 2015.

[88] J. Wade, “Devops Best Practices Checklist · GitHub,” pp. 1–8, 2017. [Online]. Available: https://gist.github.com/jpswade/4135841363e72ece8086146bd7bb5d91

[89] C. Us and StarAgile, “Top 7 DevOps practices for Successful Implementation of DevOps,” pp. 1–8, 2020. [Online]. Available: https://staragile.com/blog/devops-best-practices

[90] P. Rodríguez, M. Mäntylä, M. Oivo, L. E. Lwakatare, P. Seppänen, and P. Kuvaja, “Advances in Using Agile and Lean Processes for Software Development,” in *Advances in Computers*, ser. Advances in Computers, A. Memon, Ed. Faculty of Information Technology and Electrical Engineering, University of Oulu, Finland: Academic Press Inc., 2019, vol. 113, pp. 135–224.

[91] H. Ketterer, “Leaner, Faster, and Better with DevOps,” 2017. [Online]. Available: https://www.bcg.com/publications/2017/technology-digital-leaner-faster-better-devops

[92] PuppetGuideCIOs, “The 5 Stages of DevOps Evolution: A Guide for CIOs,” 2018. [Online]. Available: https://www.thinkahead.com/wp-content/uploads/2018/12/puppet-5-stages-devops-evolution-cio-guide.pdf

[93] O. Mikhalchuk, “5 core DevOps principles for a painless culture shift | Forte Group,” 2020. [Online]. Available: https://fortegrp.com/what-are-the-core-devops-principles/

[94] Infopulse, “6 DevOps Best Practices to Launch Enterprise-Wide Transformations | Continuous Delivery, Microservices Architecture, Collaboration | Infopulse,” 2018. [Online]. Available: https://www.infopulse.com/blog/6-devops-best-practices-to-launch-enterprise-wide-transformations/

[95] S. Badshah, A. A. Khan, and B. Khan, “Towards Process Improvement in DevOps: A Systematic Literature Review,” in *ACM International Conference Proceeding Series*, ser. EASE ’20. New York, NY, USA: Association for Computing Machinery, apr 2020, pp. 427–433.

[96] M. D. Z. Imam, “What are the Best Practices For Successful Implementation Of DevOps?” 2018. [Online]. Available: https://www.knowledgehut.com/blog/devops/best-practices-for-successful-implementation-of-devops

[97] Puppet Labs, “2017 State of DevOps Report,” Puppet Labs, Tech. Rep., 2017. [Online]. Available: https://puppetlabs.com/solutions/devops/

[98] E. DeBoer, “Accelerate: A Principle-based DevOps Framework,” 2019. [Online]. Available: https://blog.sonatype.com/principle-based-devops-frameworks-accelerate

[99] S. Team, “Understanding 6 Essential DevOps Principles - SKILLOGIC Official Blog,” 2018. [Online]. Available: https://skillogic.com/blog/understanding-6-essential-devops-principles/

[100] Puppet Labs, “2020 State of DevOps Report,” Puppet Labs, Tech. Rep., 2020. [Online]. Available: https://puppet.com/resources/report/2020-state-of-devops-report/

[101] Netapp, “What Is DevOps? - Practices and Benefits Explained | NetApp,” pp. 1–12, 2019. [Online]. Available: https://www.netapp.com/devops-solutions/what-is-devops/

[102] R. de Feijter, S. Overbeek, R. van Vliet, E. Jagroep, and S. Brinkkemper, “DevOps competences and maturity for software producing organizations,” *Lecture Notes in Business Information Processing*, vol. 318, pp. 244–259, 2018.

[103] Vilmate, “7 DevOps Best Practices for a Project Success | Vilmate,” 2020. [Online]. Available: https://vilmate.com/blog/devops-best-practices/

[104] RedHat, “What is DevSecOps?” 2018. [Online]. Available: https://www.redhat.com/en/topics/devops/what-is-devsecops

[105] D. Teixeira, R. Pereira, T. A. Henriques, M. Silva, and J. Faustino, “A Systematic Literature Review on DevOps Capabilities and Areas,” *International Journal of Human Capital and Information Technology Professionals*, vol. 11, no. 2, pp. 1–22, apr 2020.

[106] J. Faustino, “DevOps Practices in Incident Management Process,” Thesis, ISCTE-IUL, 2018. [Online]. Available: https://repositorio.iscte-iul.pt/bitstream/10071/18294/1/Master_Joao_Carvalho_Faustino.pdf

[107] N. Ceresani, “The 4 Core Capabilities of DevOps,” 2017. [Online]. Available: https://dzone.com/articles/the-4-core-capabilities-of-devops

[108] I. Sacolick, “15 KPIs to track devops transformation,” 2018. [Online]. Available: https://www.infoworld.com/article/3297041/15-kpis-to-track-devops-transformation.html

[109] H. T. C. G. Services, “24 DevOps Capabilities - DevOps Efficiency Matrix,” 2020. [Online]. Available: https://devopsefficiency.com/list.html

[110] C. A. Technologies, “How Can DevOps Practices Help You Increase - Broadcomdocs.broadcom.com,” 2017. [Online]. Available: https://docs.broadcom.com/doc/how-can-devops-practices-help-you-increase-innovation-velocity-and-business-agility-on-the-mainframe

[111] DevOps Research and Assessment (DORA), D. Research, and A. (DORA), “State of DevOps 2019 - DORA,” DORA, Tech. Rep., 2019. [Online]. Available: https://services.google.com/fh/files/misc/state-of-devops-2019.pdf

[112] K. Mikko, “DevOps Capability Assessment in a Software Development Team,” *Science*, vol. 135, pp. 408–415, 2018. [Online]. Available: https://www.theseus.fi/bitstream/handle/10024/334710/MikkoKurkela_DevOpsCapabilityAssessmentInASoftwareDevelopmentTeam.pdf

[113] Puppet Labs, “2018 State of DevOps Report,” Puppet Labs, Tech. Rep., 2018. [Online]. Available: https://media.webteam.puppet.com/uploads/2019/11/Puppet-State-of-DevOps-Report-2018_update.pdf

[114] B. I. Staff, “10 DevOps Best Practices To Know | Built In,” pp. 1–7, 2020. [Online]. Available: https://builtin.com/software-engineering-perspectives/devops-best-practices

[115] G. B. Ghantous and A. Q. Gill, “DevOps: Concepts, practices, tools, benefits and challenges,” in *Proceedings ot the 21st Pacific Asia Conference on Information Systems: ”Societal Transformation Through IS/IT”, PACIS 2017*. Association for Information Systems, 2017, p. 12.

[116] K. Eby, “The Tools and Technology of DevOps | Smartsheet,” pp. 1–26, 2017. [Online]. Available: https://www.smartsheet.com/devops-tools

[117] C. Crowley, L. McQuillan, and C. O’Brien, “Understanding DevOps: Exploring the origins, composition, merits, and perils of a DevOps Capability,” in *Proceedings of the 4th International Conference on Production Economics and Project Evaluation, ICOPEV 2018, Guimarães, Portugal*, ser. ICOPEV International Conference on Project Economic Evaluation, M. Araujo, Ed., Univ Minho, Sch Engn, Ind Engn and Management, Algoritmi Res Ctr. Guimaraes: Universade do Minho, 2018, Proceedings Paper, pp. 29–37.

[118] Scaledagile, “DevOps - Scaled Agile Framework,” 2021. [Online]. Available: https://www.scaledagileframework.com/devops/

[119] Wikipedia, “DevOps - Wikipedia,” 2021. [Online]. Available: https://en.wikipedia.org/wiki/DevOps

[120] E. Mueller, “What Is DevOps? | the agile admin,” 2019. [Online]. Available: https://theagileadmin.com/what-is-devops/

[121] T. Hall, “DevOps Best Practices | Atlassian,” 2020. [Online]. Available: https://www.atlassian.com/devops/what-is-devops/devops-best-practices

[122] Veritis, “DevOps Capabilities: 6-point Principles for Business Success,” 2018. [Online]. Available: https://www.veritis.com/blog/devops-capabilities-a-6-point-principle-that-drives-business-success/

[123] D. Teixeira, R. Pereira, T. Henriques, M. M. D. Silva, J. Faustino, O. Faustino, and M. Silva, “A maturity model for DevOps,” *International Journal of Agile Systems and Management*, vol. 13, no. 4, p. 464, 2020.

[124] A. Lichtenberger, “Blog: Agile: Dead End? Taking the next step by applying DevOps Practices effectively - impact matters Blog,” 2019. [Online]. Available: https://www.impactmatters.ch/blog/agiledevops-deadend/

[125] Department of Energy Quality Managers: Software Quality Assurance Subcommittee, “Software Configuration Management (SCM) A Practical Guide,” United States Department of Energy, Tech. Rep., 2000. [Online]. Available: http://energy.gov/sites/prod/files/cioprod/documents/scmguide.pdf

[126] ANSI/IEEE, “IEEE Std 1042-1987 Guide to Software Configuration Management,” 1987. [Online]. Available: http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=89631

[127] P. M. Mell and T. Grance, “The NIST definition of cloud computing,” National Institute of Standards and Technology, Gaithersburg, MD, Tech. Rep., 2011. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

[128] D. Stahl, T. Martensson, and J. Bosch, “Continuous practices and devops: beyond the buzz, what does it all mean?” in *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, aug 2017, pp. 440–448.

[129] A. Balalaie, A. Heydarnoori, and P. Jamshidi, “Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture,” *IEEE Software*, vol. 33, no. 3, pp. 42–52, may 2016.

[130] ——, “Migrating to Cloud-Native Architectures Using Microservices: An Experience Report,” *Communications in Computer and Information Science*, vol. 567, pp. 201–215, jul 2015.

[131] M. Stillwell and J. G. F. Coutinho, “A DevOps approach to integration of software components in an EU research project,” in *Proceedings of the 1st International Workshop on Quality-Aware DevOps*, Imperial College London United Kingdom. New York, NY, USA: ACM, sep 2015, pp. 1–6.

[132] R. Souza, F. Silva, L. Rocha, and I. Machado, “Investigating agile practices in software startups,” *ACM International Conference Proceeding Series*, pp. 317–321, 2019. [Online]. Available: http://doi.org/10.1145/3350768.3350786https://doi.org/10.1145/3350768.3350786

[133] A. Steffens, H. Lichter, and J. S. Döring, “Designing a next-generation continuous software delivery system: Concepts and architecture,” in *Proceedings - International Conference on Software Engineering*, ser. RCoSE ’18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 1–7.

[134] BoxBoat, “What is DevOps? Exploring DevOps Principles & Benefits | BoxBoat,” 2018. [Online]. Available: https://boxboat.com/2018/12/26/what-is-devops/

[135] M. E. Pasten, M. Siyam, and D. Academy, “Role-based DevOps Capability Model,” pp. 1–12, 2020. [Online]. Available: https://devonacademy.com/enterprise-devops-capability-model/

[136] C. Marnewick and J. Langerman, “DevOps and Organisational Performance: The Fallacy of Chasing Maturity,” *IEEE Software*, pp. 0–0, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9190017/

[137] R. d. F. R. v. V. E. J. S. O. S. Brinkkemper, “Towards the adoption of DevOps in software product organizations: A maturity model approach Rico,” *Journal of Chemical Information and Modeling*, vol. 53, 2019.

[138] A. Valdes, “5 DevOps Metrics and KPIs that CTOs Must Monitor,” pp. 1–9, 2020. [Online]. Available: https://www.clickittech.com/devops/devops-metrics-and-kpis/

[139] V. Fedak, “DevOps metrics: what to track, how and why do it.” pp. 1–11, 2020. [Online]. Available: https://medium.com/@FedakV/devops-metrics-what-to-track-how-and-why-do-it-e08dc6864eab

[140] E. Lock, “Measure DevOps Metrics That Matter,” 2020. [Online]. Available: https://www.devopsdigest.com/measure-devops-metrics-that-matter

[141] D. Sato, “Practices for DevOps and Continuous Delivery,” 2015. [Online]. Available: https://www.infoq.com/articles/book-DevOps-continuous-delivery/

[142] Devopedia, “DevOps Metrics,” 2019. [Online]. Available: https://devopedia.org/devops-metrics

[143] G. Motroc, “Key DevOps metrics that matter: How well does your team sleep?” pp. 1–13, 2018. [Online]. Available: https://jaxenter.com/devops-influencers-interview-series-4-142312.html

[144] AWS, “What is DevOps? - Amazon Web Services (AWS),” 2021. [Online]. Available: https://aws.amazon.com/devops/what-is-devops/

[145] CMMI Product Team, “CMMI for Development, Version 1.3,” *Software Engineering Process Management Program*, 2010.

[146] N. Forsgren, J. Humble, G. Kim, A. Brown, N. Kersten, Dr. Nicole Forsgren, Jez Humble, and Gene Kim, “Accelerate: State of DevOps 2018 Strategies for a New Economy,” *Report. DevOps Research & Assessment (DORA)*, p. 78, 2018. [Online]. Available: https://cloudplatformonline.com/rs/248-TPC-286/images/DORA-StateofDevOps.pdf

[147] Intellipaat, “What is DevOps - Introduction to DevOps Architecture & Benefits,” 2016. [Online]. Available: https://intellipaat.com/blog/what-is-devops/

[148] H. Packard, “Measuring DevOps success,” 2016. [Online]. Available: http://www.baldrover.com/wp-content/uploads/Measuring-DevOps-Success.pdf

[149] H. Myrbakken and R. Colomo-Palacios, “DevSecOps: A multivocal literature review,” *Communications in Computer and Information Science*, vol. 770, no. 1, pp. 17–29, 2017.

[150] J. Humble and B. O’Reilly, *Lean enterprise : adopting continouos delivery, devops, and lean startup at scale*. O’Reilly Media, Inc., 2014.

[151] M. Shahin, M. Ali Babar, and L. Zhu, “Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices,” *IEEE Access*, vol. 5, pp. 3909–3943, 2017. [Online]. Available: http://ieeexplore.ieee.org/document/7884954/

[152] A. R. Chowdhury, “DevOps Best Practices: A Complete Guide,” 2019. [Online]. Available: https://stackify.com/devops-best-practices-a-complete-guide/

[153] K. Wakayama, “How to Ensure the Success of DevOps in Your Organization,” pp. 1–10, 2020. [Online]. Available: https://codersociety.com/blog/articles/devops-success-in-organization

[154] Ubiq, “Top DevOps Metrics and KPIs To Monitor Regularly - Ubiq BI Blog,” pp. 1–12, 2020. [Online]. Available: http://ubiq.co/analytics-blog/top-devops-metrics-kpis-to-monitor-regularly/

[155] Puppet Labs, “2016 State of DevOps Report,” Puppet Labs, Tech. Rep. 7, 2016. [Online]. Available: https://puppetlabs.com/solutions/devops/

[156] A. Wiedemann, N. Forsgren, M. Wiesche, H. Gewald, and H. Krcmar, “Research for practice: The Devops phenomenon,” *Communications of the ACM*, vol. 62, no. 8, pp. 44–49, 2019.

[157] M. Outlaw, “The DevOps Handbook – The Technical Practices of Flow,” pp. 1–7, 2020. [Online]. Available: https://www.codingblocks.net/podcast/the-devops-handbook-the-technical-practices-of-feedback/

[158] D. Cukier, “DevOps patterns to scale web applications using cloud services,” in *SPLASH 2013 - Proceedings of the 2013 Companion Publication for Conference on Systems, Programming, and Applications: Software for Humanity*. New York, New York, USA: ACM Press, 2013, pp. 143–152. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2508075.2508432

[159] Veritis, C. Services, D. D. T. Services, D. D. T. Services, A. Management, M. Services, T. Advisory, F. Services, R. Estate, C. Studies, W. Papers, and Veritis, “Measuring DevOps: Key ‘Metrics’ and ‘KPIs’ That Drive Success!” pp. 1–10, 2020. [Online]. Available: https://www.veritis.com/blog/measuring-devops-key-metrics-and-kpis-that-drive-success/

[160] N. Tomas, J. Li, and H. Huang, “An empirical study on culture, automation, measurement, and sharing of DevSecOps,” in *2019 International Conference on Cyber Security and Protection of Digital Services, Cyber Security 2019*, jun 2019, pp. 1–8.

[161] S. I. Mohamed, “DevOps Maturity Calculator DOMC -Value oriented approach,” *International Journal of Engineering Research and Science*, vol. 2, no. 2, pp. 2395–6992, 2016.

[162] P. Waterhouse, “DevOps Practitioner Series - Metrics That Matter,” 2015. [Online]. Available: https://docs.broadcom.com/doc/devops-practitioner-series-metrics-that-matter-developing-and-tracking-key-indicators

[163] S. Smith, “High performing DevOps metrics,” pp. 1–11, 2020. [Online]. Available: https://samlearnsazure.blog/2020/04/30/high-performing-devops-metrics/

[164] V. Gupta, P. K. Kapur, and D. Kumar, “Modeling and measuring attributes influencing DevOps implementation in an enterprise using structural equation modeling,” *Information and Software Technology*, vol. 92, no. 1, pp. 75–91, 2017. [Online]. Available: http://search.ebscohost.com/login.aspx?direct=true&db=eoah&AN=42845256&lang=pt-pt&site=ehost-live&scope=site

[165] RDC Partner, “6 core capabilities of DevOps practice,” pp. 2–6, 2018. [Online]. Available: https://rdcpartner.nl/6-core-capabilities-of-devops-practice/

[166] C. Rudder, “10 ways DevOps helps digital transformation | The Enterprisers Project,” 2019. [Online]. Available: https://enterprisersproject.com/article/2019/8/devops-role-digital-transformation

[167] S. Yusuf, “Ebbs and Flows Of DevOps Debugging PART 1 - Security Boulevard,” pp. 1–32, 2021. [Online]. Available: https://securityboulevard.com/2021/02/ebbs-and-flows-of-devops-debugging-part-1/

[168] Puppet, “5 Foundational DevOps Practices: How to Establish & Build on Them | Puppet,” 2018. [Online]. Available: https://puppet.com/resources/whitepaper/5-foundational-devops-practices-how-establish-build-them/

[169] T. Jachja, “CI/CD Patterns and Practices — DevOps Institute,” 2020. [Online]. Available: https://devopsinstitute.com/ci-cd-patterns-and-practices/

[170] C. Patel, “DevOps Best Practices - DZone DevOpsdzone.com,” 2020. [Online]. Available: https://dzone.com/articles/devops-best-practices

[171] R. Powell, “Essential DevOps Principles for 2021 | CircleCIcircleci.com,” 2020. [Online]. Available: https://circleci.com/blog/essential-devops-principles/

[172] M. Patil, “Maximizing DevOps ROI with Modern Application Practices | HCL Blogs,” 2020. [Online]. Available: https://www.hcltech.com/blogs/maximizing-devops-roi-modern-application-practices

[173] D. Online, “The Practice of DevOps,” 2020. [Online]. Available: https://www.dqindia.com/the-practice-of-devops/

[174] C. Solutions, “8 Best Practices for Successful Implementation of DevOps,” 2019. [Online]. Available: https://dev.to/credencys/8-best-practices-for-successful-implementation-of-devops-in-your-enterprise-2k93

[175] G. Pousseo, “DevOps Practices: Agility without DevOps is pointless,” 2019. [Online]. Available: https://www.pentalog.com/blog/agility-without-devops-is-pointless

[176] A. D. Rayome, “5 foundational DevOps practices your enterprise needs to succeed - TechRepublic,” 2018. [Online]. Available: https://www.techrepublic.com/article/5-foundational-devops-practices-your-enterprise-needs-to-succeed/

[177] K. Kuusinen, V. Balakumar, S. C. Jepsen, S. H. Larsen, T. A. Lemqvist, A. Muric, A. Ø. Nielsen, and O. Vestergaard, “A large agile organization on its journey towards DevOps,” in *Proceedings - 44th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2018*. Institute of Electrical and Electronics Engineers Inc., aug 2018, pp. 60–63.

[178] P. CIO, “Building a Strong DevOps Foundation | CIO,” 2018. [Online]. Available: https://www.cio.com/article/3319077/building-a-strong-devops-foundation.html

[179] N. Stamenkovic, “DevOps - principles, practices and why should you care? - Ingsoftware Blog,” pp. 1–11, 2018. [Online]. Available: https://www.ingsoftware.com/devops-principles-and-practices

[180] Pinkelephant, “DevOps: Why The 15 Essential Practices Are Key To Your Organization’s Survival | Pink Elephant Blog,” 2018. [Online]. Available: https://blog.pinkelephant.com/blog/devops-why-the-15-essential-practices-are-key-to-your-organizations-surviva

[181] PuppetSplunk, “The 5 foundational devops practices,” 2018. [Online]. Available: https://www.1sttechguide.com/wp-content/uploads/2019/09/The-5-Foundational-DevOps-Practices.pdf

[182] A. Crouch, A. Gaspari, and A. Crouch, “6 Steps to a Successful DevOps Adoption | AgileConnection,” 2017. [Online]. Available: https://www.agileconnection.com/article/6-steps-successful-devops-adoption

[183] C. Pang and A. Hindle, “Continuous maintenance,” in *Proceedings - 2016 IEEE International Conference on Software Maintenance and Evolution, ICSME 2016*. Institute of Electrical and Electronics Engineers Inc., oct 2017, pp. 458–462.

[184] A. Wadhera, “10 Key DevOps Practices to Improve IT Efficiency,” 2016. [Online]. Available: https://www.tothenew.com/blog/10-key-devops-practices-to-improve-it-efficiency/

[185] L. Zhu, L. Bass, and G. Champlin-Scharff, “DevOps and Its Practices,” *IEEE Software*, vol. 33, no. 3, pp. 32–34, may 2016.

[186] D. Linthicum, “DevOps tools best practices: A 7-step guide,” 2016. [Online]. Available: https://techbeacon.com/devops/7-steps-choosing-right-devops-tools

[187] K. Horvath, “DevOps Part 2: Methods, Practices and Tools,” 2015. [Online]. Available: https://content.intland.com/blog/agile/devops/devops-part-2-methods-practices-and-tools

[188] M. Croker, “DevOps: innovative engineering practices for continuous software delivery,” 2015. [Online]. Available: https://www.accenture.com/_acnmedia/PDF-18/Accenture-DevOps-brochure-new.pdf

[189] M. Rother, *Toyota Kata: Managing people for improvement, adaptiveness and superior results*. MGH, New York, 2019.

[190] D. N, “Quick List of DevOps Best Practices,” pp. 1–16, 2019. [Online]. Available: https://www.whizlabs.com/blog/devops-best-practices/

[191] J. Groll, “What is a DevOps ’Best Practice’? - DevOps.com,” 2016. [Online]. Available: https://devops.com/devops-best-practice/

[192] Cambridge, “CAPABILITY | meaning in the Cambridge English Dictionary,” 2021. [Online]. Available: https://dictionary.cambridge.org/dictionary/english/capability

[193] Merriam-Webster, “Capability | Definition of Capability by Merriam-Webster,” 2021. [Online]. Available: https://www.merriam-webster.com/dictionary/capability

[194] T. Biggs, M. Shah, and P. Srivastava, *Technological capabilities and learning in African enterprises*. The World Bank, 1995.

[195] D. N. Blank-edelman, *Seeking SRE: Conversations About Running Production Systems at Scale*. O’Reilly Media, Inc., 2018.

[196] S. Rafi, W. Yu, and M. A. Akbar, “RMDevOps,” in *Proceedings of the Evaluation and Assessment in Software Engineering*. New York, NY, USA: ACM, apr 2020, pp. 413–418.

[197] P. Perera, R. Silva, and I. Perera, “Improve software quality through practicing DevOps,” in *2017 Seventeenth International Conference on Advances in ICT for Emerging Regions (ICTer)*, ser. 2017, no. September 2017. IEEE, sep 2017, pp. 1–6.

[198] W. P. Luz, G. Pinto, and R. Bonifácio, “Adopting DevOps in the real world: A theory, a model, and a case study,” *Journal of Systems and Software*, vol. 157, no. July, p. 110384, nov 2019. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0164121219301517

[199] G. Logic, “DevOps Success: What to Measure and Why - Gorilla Logic,” pp. 1–6, 2020. [Online]. Available: https://gorillalogic.com/blog/devops-success-what-to-measure-and-why/

[200] Gitlab, “Getting Started with Agile/DevOps Metrics | GitLab,” 2020. [Online]. Available: https://about.gitlab.com/handbook/marketing/strategic-marketing/devops-metrics/

[201] I. Technology and I. Consulting, “2019 Global Server Hardware, Server OS Reliability Report,” Information Technology Intelligence Consulting (ITIC) Corp., Tech. Rep. March, 2019.

[202] AlertOps, “MTTD vs MTTF vs MTBF vs MTTR - Resolve Major IT Incidents Quickly,” 2018. [Online]. Available: https://alertops.com/mttd-vs-mttf-vs-mtbf-vs-mttr/

[203] Cprime, “DevOps Metrics to Monitor Software Development - Cprime,” pp. 1–15, 2021. [Online]. Available: https://www.cprime.com/resources/blog/devops-metrics-to-monitor-software-development/

[204] Flosum, “Keys to Improve Salesforce DevOps Efficiency - Flosum - Continuous Integration, release management,” 2021. [Online]. Available: https://flosum.com/keys-to-improve-salesforce-devops-efficiency/

[205] S. Miteva, “13 DevOps Metrics for Increased Productivity,” pp. 1–9, 2020. [Online]. Available: https://dev.to/microtica/13-devops-metrics-for-increased-productivity-5084

[206] T. Development, “A Guide To Measuring DevOps Success an Proving ROI,” 2020. [Online]. Available: https://www.tiempodev.com/blog/measuring-devops/

[207] R. A. Sarker, L. R. Khan, and R. A. Sarker, “An optimal batch size for a JIT manufacturing system,” *Computers & Industrial Engineering*, vol. 42, no. 2-4, pp. 127–136, 2002.

[208] R. Pruess, “DevOps Best Practices: 5 Key Performance Indicators,” pp. 1–7, 2020. [Online]. Available: https://flexagon.com/devops-best-practices-5-key-performance-indicators/

[209] Opsgenie and O. Support, “DevOps Metrics,” pp. 1–15, 2020. [Online]. Available: https://docs.opsgenie.com/docs/devops-metrics-global

[210] T. Nero, “DevOps Metrics : 15 KPIs that Boost Results and RoI - Cuelogic Technologies Pvt. Ltd.” 2020. [Online]. Available: https://www.cuelogic.com/blog/devops-metrics

[211] Appdynamics, “DevOps Metrics and KPIs: How To Measure DevOps?” pp. 1–14, 2020. [Online]. Available: https://www.appdynamics.com/topics/devops-metrics-and-kpis

[212] J. Gelo, “DevOps Metrics Matter: Why, Which Ones, and How - HCL SW Blogs,” 2020. [Online]. Available: https://blog.hcltechsw.com/accelerate/devops-metrics-matter-why-which-ones-and-how-2/

[213] C. C. Ann Marie Fred, “6 proven metrics for DevOps success | TechBeacon,” 2020. [Online]. Available: https://techbeacon.com/devops/6-proven-metrics-devops-success

[214] ReleaseTEAM, “DevOps Metrics: Measure Your DevOps Results,” 2020. [Online]. Available: https://www.releaseteam.com/measure-your-devops-results/

[215] A. Khanduri, “DevOps Metrics: Measuring What Matters,” 2020. [Online]. Available: https://blog.sonatype.com/devops-metrics-measuring-what-matters

[216] G. Guimarães, “On the Four Key DevOps Metrics, and why I measure them differently – SourceLevel,” 2020. [Online]. Available: https://sourcelevel.io/blog/on-the-four-key-devops-metrics-and-why-i-measure-them-differently

[217] A. Ramesh, “Ten Key DevOps Metrics to Accelerate Your Continuous Delivery Pipeline,” pp. 1–6, 2020. [Online]. Available: https://www.go2group.com/resources/blog/devops-metrics-to-accelerate-ci-cd/

[218] C. Lawrence, “The Four Key Metrics of DevOps,” pp. 1–16, 2020. [Online]. Available: https://humanitec.com/blog/devops-key-metrics

[219] V. Kamani, “7 crucial DevOps metrics that you need to track,” pp. 1–8, 2019. [Online]. Available: https://hub.packtpub.com/7-crucial-devops-metrics-that-you-need-to-track/

[220] R. Shinde, “Is your DevOps successful?” 2019. [Online]. Available: https://www.accenture.com/us-en/blogs/software-engineering-blog/reshma-shinde-devops-success-metrics

[221] N. Relic, “Measuring DevOps,” 2018. [Online]. Available: https://newrelic.com/devops/measuring-devops

[222] M. Michaelis, “DevOps Metrics - IntelliTect,” 2015. [Online]. Available: https://intellitect.com/devops-metrics/

[223] R. Honig, P. Page, K. Base, P. Page, K. Base, O. P. Debugger, O. Visual, S. Extension, P. Page, K. Base, P. Page, K. Base, O. P. Debugger, O. Visual, S. Extension, O. P. Debugger, O. Visual, S. Extension, O. P. Debugger, O. Visual, S. Extension, and R. Honig, “Bridge the DevOps Development Chasm to Boost DevOps KPIs - Ozcode,” pp. 1–14, 2020. [Online]. Available: https://oz-code.com/blog/devops/bridging-the-devops-development-observability-chasm-to-boost-devops-kpis

[224] J. Kernel, “DevOps Metrics: 7 KPIs to Evaluate Your Team’s Maturity,” 2020. [Online]. Available: https://www.xplg.com/devops-metrics-7-kpis/

[225] H. Vora, “Software Quality Metrics for Agile and DevOps success - QMetry,” pp. 1–14, 2018. [Online]. Available: https://www.qmetry.com/blog/software-quality-metrics-for-agile-and-devops-success/

[226] R. Powell, “How to measure DevOps success: 4 key metrics,” 2020. [Online]. Available: https://circleci.com/blog/how-to-measure-devops-success-4-key-metrics/

[227] T. Gilmore, D. Toolchain, and D. Metrics, “DevOps Metrics - ADAPT Model Community,” 2018. [Online]. Available: http://www.adapttransformation.com/devops-toolchain/monitor/devops-metrics/

[228] R. Osherove, “Ten Devops an Agility Metrics to Check at the Team Level - Pipeline Driven,” pp. 1–3, 2018. [Online]. Available: https://pipelinedriven.org/article/ten-ideas-for-things-you-can-measure-as-a-team-on-your-devops-journey

[229] F. J. Moraes, “DevOps KPI in Practice — Chapter 1 — Deployment Speed, Frequency and Failure,” pp. 1–18, 2018. [Online]. Available: https://medium.com/@fabiojose/devops-kpi-in-practice-chapter-1-deployment-speed-frequency-and-failure-2fd0a9303249

[230] B. Dobran, “15 DevOps Metrics and Key Performance Indicators (KPIs) To Track,” 2019. [Online]. Available: https://phoenixnap.com/blog/devops-metrics-kpis

[231] R. Bobbett, “DevOps Value: How to Measure the Success of DevOps,” pp. 1–6, 2018. [Online]. Available: https://www.fpcomplete.com/blog/devops-value-how-to-measure-the-success-of-devops/

[232] D. Better, S. Faster, and A. Weiss, “Measuring DevOps Flow by otomato,” 2016. [Online]. Available: https://devopsflowmetrics.org/

[233] D. Holloran, “Top Metrics for Measuring DevOps Delivery Value,” pp. 1–12, 2019. [Online]. Available: https://victorops.com/blog/top-metrics-for-measuring-devops-delivery-value

[234] Digital.ai, “4 DevOps Metrics to Improve Delivery Performance on Vimeo,” pp. 1–5, 2019. [Online]. Available: https://vimeo.com/331032185

[235] N. Forsgren, “Metrics for DevOps Initiatives - nicole forsgren,” 2015. [Online]. Available: https://nicolefv.com/s/DOES_forum_metrics_102015.pdf

[236] D. Kodjamanova, “4 DevOps Metrics To Maximize Success - MentorMate,” pp. 1–20, 2020. [Online]. Available: https://mentormate.com/blog/how-devops-metrics-pave-the-way-to-better-performance/

[237] I. Software, “6 Metrics to Measure DevOps Test Automation,” 2017. [Online]. Available: https://www.indiumsoftware.com/blog/devops-test-automation-metrics/

[238] R. Edwards, “Research highlights challenges of Salesforce DevOps in 2020 -,” 2021. [Online]. Available: https://www.enterprisetimes.co.uk/2021/02/16/research-highlights-challenges-of-salesforce-devops-in-2020/

[239] Digital.ai, “10 DevOps Metrics You Should Know,” 2020. [Online]. Available: https://digital.ai/resources/infographic/10-devops-metrics-you-should-know

[240] Stackexchange, D. S. Exchange, I. Management, C. Management, R. M. To, R. C. Analysis, Stackexchange, and M. Hossain, “What key performance indicators (KPIs) are used to measure DevOps?” pp. 30–32, 2020. [Online]. Available: https://devops.stackexchange.com/questions/738/what-key-performance-indicators-kpis-are-used-to-measure-devops

[241] K. Ahmad, “DevOps KPIs and "Design for Failure",” pp. 1–4, 2020. [Online]. Available: https://www.linkedin.com/pulse/devops-kpis-design-failure-khalil-ahmad

[242] Plutora, “DORA DevOps Metrics - Accelerate your Value Stream - Plutora.com,” 2020. [Online]. Available: https://www.plutora.com/resources/videos/devops-dora-metrics

[243] J. Riggins, “Google’s Formula for Elite DevOps Performance – The New Stack,” 2020. [Online]. Available: https://thenewstack.io/googles-formula-for-elite-devops-performance/

[244] P. Gallagher, “Tracking Success in DevOps Pipelines,” pp. 1–14, 2020. [Online]. Available: https://blog.goodelearning.com/subject-areas/devops/how-to-measure-success-in-devops/

[245] D. G. Portman, “Using the Four Keys to measure your DevOps performance,” 2020. [Online]. Available: https://cloud.google.com/blog/products/devops-sre/using-the-four-keys-to-measure-your-devops-performance

[246] M. Hossain, “What key performance indicators (KPIs) are used to measure DevOps?” 2020. [Online]. Available: https://www.quora.com/What-key-performance-indicators-KPIs-are-used-to-measure-DevOps

[247] O. Nasser, “What Metrics Should DevOps Teams Be Tracking?” pp. 1–5, 2020. [Online]. Available: https://cto.ai/blog/what-metrics-should-devops-teams-be-tracking/

[248] M. Bizzantino, “4 fundamental metrics to measure DevOps performances,” 2019. [Online]. Available: https://www.kiratech.it/en/blog/4-fundamental-metrics-to-measure-devops-performances

[249] D. Dingley, V. S. As, A. Coach, V. Solutions, D. Dingley, I. Team, and H.-c. Interaction, “4 Key Metrics for DevOps Success Video,” 2019. [Online]. Available: https://www.veracitysolutions.com/4-key-metrics-for-devops-success

[250] L. Cīrule, “Analyze DevOps Metrics With eazyBI,” pp. 1–8, 2019. [Online]. Available: https://eazybi.com/blog/analyze-devops-metrics-with-eazybi

[251] O. Habib, B. Home, C. Delivery, C. Integration, C. C. Management, S. Page, G. Started, and O. Habib, “DevOps Accelerate Metrics | Harness Platform-as-a-Service,” pp. 1–13, 2019. [Online]. Available: https://harness.io/blog/featured/continuous-insights-accelerate-devops-metrics/

[252] R. Cyber, “DevOps KPIs to Measure Success,” 2019. [Online]. Available: https://www.royalcyber.com/blog/devops/devops-kpis-to-measure-success/

[253] M. Baldani, “DORA Metrics - Getting on the Bandwagon,” pp. 1–5, 2019. [Online]. Available: https://www.cloudbees.com/blog/dora-metrics-getting-bandwagon

[254] M. Samuel, “How to Successfully Scale Agile and DevOps – Part 3: Driving Success with Technology,” 2019. [Online]. Available: https://www.hcltech.com/blogs/how-successfully-scale-agile-and-devops-part-3-driving-success-technology

[255] G. Singh, D. Transformation, S. Providers, H. Multi-cloud, E. H. M.-c. Environment, D. Intelligence, E. D. Strategy, E. D. Strategy, D. I. Management, P. Strategy, E. Design, E. Devops, M. Cyber, and S. Solutions, “Measuring DevOps Success with DevOps Metrics,” pp. 1–14, 2019. [Online]. Available: https://www.xenonstack.com/blog/devops-metrics/

[256] A. Hawkins, “Measuring DevOps Success: What, Where, and How - Cloud Academy,” pp. 1–16, 2019. [Online]. Available: https://cloudacademy.com/blog/measuring-devops-success-what-where-and-how/

[257] A. Schurr, “Mobile App DevOps Metrics that Matter - NowSecure,” 2019. [Online]. Available: https://www.nowsecure.com/blog/2019/02/27/mobile-app-devops-metrics-that-matter/

[258] TestEnvMgt, “Top 5 DevOps Metrics – Test Environment Management,” pp. 1–8, 2019. [Online]. Available: https://www.testenvironmentmanagement.com/top-5-devops-metrics/

[259] P. Duvall, “Measuring DevOps Success with Four Key Metrics | Stelligent,” pp. 1–5, 2018. [Online]. Available: https://stelligent.com/2018/12/21/measuring-devops-success-with-four-key-metrics/

[260] M. Watson, “15 Metrics for DevOps Success,” pp. 1–19, 2017. [Online]. Available: https://stackify.com/15-metrics-for-devops-success/

[261] L. Karam, “DevOps metrics you must take into account,” 2017. [Online]. Available: https://apiumhub.com/tech-blog-barcelona/devops-metrics/

[262] S. Watts, “DevOps: Metrics and Key Performance Indicators (KPIs),” 2017. [Online]. Available: https://itchronicles.com/devops/devops-metrics-kpis/

[263] DevOpsEnterpriseSummit, “Featured Resource: Metrics for DevOps Initiatives - IT Revolution,” 2017. [Online]. Available: https://itrevolution.com/devops-resource-metrics/

[264] N. Forsgren, “How to use metrics, measurement to drive DevOps,” 2017. [Online]. Available: https://techbeacon.com/devops/how-use-metrics-measurement-drive-devops

[265] J. A. X. London, “Measuring DevOps: The Key Metrics That Matter - JAX London,” 2017. [Online]. Available: https://jaxlondon.com/blog/devops-continuous-delivery/measuring-devops-key-metrics-matter/

[266] R. Technology, “Seven Metrics That Matter When Measuring DevOps Success,” 2017. [Online]. Available: https://www.aternity.com/blogs/seven-metrics-matter-measuring-devops-success/

[267] C. Shabe, “Understanding DevOps metrics,” pp. 1–5, 2017. [Online]. Available: https://betanews.com/2017/08/24/devops-metrics/

[268] KnowledgeHut, “What are the metrics and why do they matter for DevOps Success?” 2017. [Online]. Available: https://www.knowledgehut.com/blog/agile/metrics-matters-devops-success

[269] T. Hall, “DevOps metrics | Atlassian,” 2016. [Online]. Available: https://www.atlassian.com/devops/frameworks/devops-metrics

[270] C. Staff, A. Development, A. Testing, D. Engineering, A. D. Training, D. Training, S. S. T. Training, S. S. T. Training, E. U. Classes, Q. Starts, A. Coveros, O. Team, and C. Staff, "Essential Quantitative DevOps Metrics - Coveros," pp. 1–5, 2016. [Online]. Available: https://www.coveros.com/essential-quantitative-devops-metrics/

[271] Cigniti, "Top 6 DevOps Metrics that Enterprise Dashboards Should Capture," 2016. [Online]. Available: https://www.cigniti.com/blog/6-devops-metrics-for-enterprise-dashboards/

[272] Valtech, "4 metrics for measuring DevOps success," pp. 1–12, 2015. [Online]. Available: https://www.valtech.com/insights/4-metrics-for-measuring-devops-success/

[273] P. Arora, B. Branches, and C. Coverage, "Measuring the Success of DevOps – Prashant Arora's Blog," 2015. [Online]. Available: https://aroraprashant.wordpress.com/2015/04/14/measuring-the-success-of-devops/

[274] Pagerduty, "The Best Metrics for Driving Cultural Change in DevOps Teams," 2015. [Online]. Available: https://www.pagerduty.com/blog/best-metrics-devops-culture/

[275] A. Earnshaw, "5 KPIs that Make the Case for DevOps," 2013. [Online]. Available: https://puppet.com/blog/5-kpis-make-case-for-devops/

[276] P. Lab, "Metrics for Successful DevOps ?" 2020. [Online]. Available: https://performancelabus.com/successful-devops-metrics/

[277] S. Watts, "DevOps Metrics and KPIs – BMC Blogs," pp. 1–8, 2019. [Online]. Available: https://www.bmc.com/blogs/devops-kpi-metrics/

[278] M. Edwards, "Measuring for Success - Change or Hold - DevOps," 2019. [Online]. Available: https://www.tesm.com/resources-blog-measuring-for-success-should-you-change-or-should-you-hold-devops-pt-5/

[279] H. Idan, "The Must Have Metrics Any DevOps and SRE Manager Should Measure," pp. 1–8, 2018. [Online]. Available: https://www.overops.com/blog/the-must-have-metrics-any-devops-and-sre-manager-should-measure/

[280] B. Gracely, "The Most Important DevOps Metric to Measure," 2017. [Online]. Available: https://www.openshift.com/blog/important-devops-metric-measure

[281] K. de Boer, "2 Most Important DevOps Metrics Tools," 2016. [Online]. Available: https://labs.sogeti.com/the-two-most-important-metrics-for-devops/

[282] T. Palko, "The Missing Metrics of DevOps," 2015. [Online]. Available: https://insights.sei.cmu.edu/devops/2015/05/the-missing-metrics-of-devops.html

[283] B. Beyer, C. Jones, J. Petoff, and N. R. Murphy, *Site Reliability Engineering: How Google Runs Production Systems*. O'Reilly Media, Inc., 2016. [Online]. Available: https://landing.google.com/sre/sre-book/toc/

[284] C. J. Kuiper, "Relationship of Transformational Leadership and Organizational Change During Enterprise Agile and DevOps Initiatives In Financial Service Firms," Ph.D. dissertation, Liberty University, School of Business, 2019.

[285] A. Kankanhalli, B. C. Tan, and K. K. Wei, "Contributing knowledge to electronic knowledge repositories: An empirical investigation," *MIS Quarterly: Management Information Systems*, vol. 29, no. 1, pp. 113–143, 2005.

[286] K. Behr, G. Kim, and G. Spafford, *The Visible Ops Handbook: Implementing ITIL in Four Practical and Auditable Steps*. IT Process Institute (ITPI), 2005.

[287] A. Kaushik, *Web analytics 2.0: The art of online accountability and science of customer centricity*. John Wiley and Sons, 2009.

[288] Kris Buytaert, "Help, My Datacenter is on fire," 2021. [Online]. Available: https://www.slideshare.net/KrisBuytaert/help-my-datacenter-is-on-fire

[289] ——, "Homepage of Kris Buytaert," 2021. [Online]. Available: https://krisbuytaert.be/index.shtml

[290] B. Saunders, J. Sim, T. Kingstone, S. Baker, J. Waterfield, B. Bartlam, H. Burroughs, and C. Jinks, "Saturation in qualitative research: exploring its conceptualization and operationalization," *Quality and Quantity*, vol. 52, no. 4, pp. 1893–1907, jul 2018.

[291] T. Power, D. Jackson, B. Carter, and R. Weaver, "Misunderstood as mothers: women's stories of being hospitalized for illness in the postpartum period," *Journal of Advanced Nursing*, vol. 71, no. 2, pp. 370–380, feb 2015.

[292] M. P. Grady, *Qualitative and action research: A practitioner handbook*. Phi Delta Kappa International, 1998.

[293] J. Venable, J. Pries-Heje, and R. Baskerville, "FEDS: a Framework for Evaluation in Design Science Research," *European Journal of Information Systems*, vol. 25, no. 1, pp. 77–89, jan 2016.

[294] ——, "A Comprehensive Framework for Evaluation in Design Science Research," in *Proceedings of the 7th International Conference on Design Science Research in Information Systems*, 2012, pp. 423–438.

[295] J. Pries-Heje, R. Baskerville, and J. Venable, "Strategies for design science research evaluation," *16th European Conference on Information Systems, ECIS 2008*, 2008.

[296] J. Pries-Heje and R. Baskerville, “The design theory nexus,” *MIS Quarterly: Management Information Systems*, vol. 32, no. 4, pp. 731–755, 2008.

[297] M. J. McIntosh and J. M. Morse, “Situating and constructing diversity in semi-structured interviews,” *Global Qualitative Nursing Research*, vol. 2, 2015.

Appendix A. Interview Outline

Table A.1: Questions used in the research proposal, adapted from [9].

ID	Related RQs	Type	Question
1			**Background Information**
1.1	-	Closed-ended	How large is your organization?
1.2	-	Open-ended	How do you see your current DevOps adoption?
1.3	-	Closed-ended	What is your team size currently?
2			**Methods**
2.1	-	Open-ended	What software engineering methods or capabilities are you using?
2.2	RQ1	Open-ended	What capabilities of DevOps are you using and why?
2.3	RQ4	Open-ended	What metrics should be tracked in the DevOps process and why?
3			**Categorization**
3.1	RQ7	Open-ended	How are DevOps capabilities categorized?
3.2	RQ8	Open-ended	How are the main metrics categorized?
4			**Software Life Cycle Impact**
4.1	-	Open-ended	What aspects most impact your day to day work?
4.2	RQ9	Open-ended	What DevOps capabilities have a positive impact in which main metrics?
4.3	RQ9	Closed-ended	Do you see any relations that are missing or incorrect in the shown table?
5			**DevOps Capabilities Challenges and Benefits**
5.1	-	Open-ended	Any other DevOps capabilities challenges or benefits you would like to mention?
6			**DevOps Metrics Challenges and Benefits**
6.1	-	Open-ended	Any other DevOps metrics challenges or benefits you would like to mention?

Table A.2: Interview iterations topics used in the evaluating the proposed artifact, adapted from [9].

ID	Related RQs	Type	Question
1			**Background Information**
1.1	-	Closed-ended	How large is your organization?
1.2	-	Open-ended	How do you see your current DevOps adoption?
1.3	-	Closed-ended	What is your team size currently?
2			**Categorization evaluation**
2.1	RQ7	Closed-ended	Do you agree with the DevOps capabilities categorization shown?
2.2	RQ7	Open-ended	If not how would you categorize capabilities?
2.3	RQ8	Closed-ended	Do you agree with the DevOps metrics categorization shown?
2.4	RQ8	Open-ended	If not how would you categorize capabilities?
3			**Capabilities and Metrics evaluation**
3.1	-	Open-ended	Do you see value in using this evaluation matrix and why?
3.2	RQ1	Open-ended	Do you agree or disagree with any of these DevOps capabilities and why?
3.3	RQ4	Open-ended	Do you agree or disagree with any of these DevOps metrics and why?
4			**Impact based evaluation**
4.1	RQ9	Closed-ended	Do you disagree with any of the relations shown in the table and why?
4.2	RQ9	Open-ended	What more DevOps capabilities have a positive impact in which main metrics?
4.3	-	Open-ended	What would be your expected results from applying these capabilities and metrics in your organization?

B

Appendix B. Source Code

Listing B.1: Python code for consistent fetching of large number of Google search results.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# This code fetches google search results in a systematic form for MLR
# Author: Ricardo Amaro
# Date: February 2021
# Requirements: `pip install requests bs4 pandas`

from requests import get
from bs4 import BeautifulSoup
import pandas
import time

def csv_dump(results, name):
    print(results, name)
    df = None
    df = pandas.DataFrame(results)
    df.index += 1
    df.to_csv(name + '.csv')

def parse_results(raw_html):
    soup = BeautifulSoup(raw_html, 'html.parser')
    result_block = soup.find_all('div', attrs={'class': 'g'})
    for result in result_block:
        link = result.find('a', href=True)
        title = result.find('h3')
        if link and title:
            yield { 'URL': link['href'], 'Title': link.text.strip() }

def google_search(query,max_results=500,num_results=100,lang="en"):
    usr_agent ={'User-Agent': 'Mozilla/5.0 (X11; Linux x86_64; rv:10.0) \
    AppleWebKit/537.36 (KHTML, like Gecko) Chrome/61.0.3163.100 Safari/537.36'}
    escaped_query = query.replace(' ', '+')
    results = []
    for start in range(0,max_results,num_results):
        google_url = 'https://www.google.com/search?q={}&num={}&start={}&hl={}'.format
            (
            escaped_query, num_results+1, start, lang)
        print(google_url)
        time.sleep(3)
        response = get(google_url, headers=usr_agent)
        response.raise_for_status()
        results += parse_results(response.text)
    return results

def get_results(query, name):
    print(query, name)
    results = google_search(query)
    print(results)
    csv_dump(results, name)

if __name__ == '__main__':
    get_results(
        'devops AND (practices OR capabilities)',
        'capabilities')
    get_results(
        'devops AND (metrics OR measures OR kpi OR indicator)',
        'metrics')
```