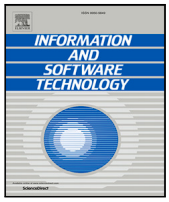
Information and Software Technology 157 (2023) 107150

Contents lists available at ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof



DevOps critical success factors — A systematic literature review

Nasreen Azad *, Sami Hyrynsalmi

Department of Software Engineering, LUT University, PL 20, Lappeenranta, FI-53851, Finland

ARTICLE INFO

Keywords:
DevOps
Development and operations
Success factors
Barriers
Literature review

ABSTRACT

Context: DevOps is a set of software development and operation practices and a recent addition to a large family of different kinds of software process models. The model emerged out of the observation that information systems operations and developments should be closely integrated activities to ensure the success of any organization. Thus, DevOps methods are an additive tool for companies to improve overall performance in their software development processes and operations.

Objective: This paper aims to identify the various critical success factors (CSFs) of DevOps projects that have been discussed in prior research. In addition, this study proposes a comprehensive framework for depicting how these CSFs impact or drive DevOps success.

Method: This study consists of a systematic literature review to collect the primary articles for the analysis.

Results: After searches in four major publication databases and snowballing, we selected 38 primary studies for the analysis. Nearly 100 different CSFs were identified, which were then categorized into *Technical*, *Organizational*, and *Social & Cultural* dimensions. Based on the results of the literature analysis, a comprehensive framework is proposed that depicts how the CSFs impact or drive DevOps success.

Conclusion: This paper presents a DevOps framework with various CSFs based on prior literature. The proposed framework will provide collective knowledge of DevOps success factors, which will allow researchers and practitioners to enhance their understanding of CSFs and learn how to handle DevOps issues in organizations. In particular, the paper highlights a number of future research directions related to CSFs.

1. Introduction

To compete in a highly-volatile market and maintain a competitive advantage, software organizations must release products that are both effective and sustainable against competition [1,2]. Software companies need to deliver product features in short cycles [1], since for customers, it is important to have new features within an efficient time frame [3]. For better software quality and faster delivery of new software features, software-intensive companies continue to work on improving their software development processes [4]. An empirical study conducted on 14 agile software companies found that time was one of the most important value aspects among the participants in terms of the delivery process [5]. Therefore, continuous practices have been implemented by the software development industry to enable organizations to release new features and products in faster ways [6–8]. However, as the software business has matured over the years, the requirements for speed and efficiency have also changed [9]. As a response to this changing business environment, software development processes and life cycles have also been evolving.

The need for an agile method was raised due to the incapability of the waterfall method as it was not capable enough to adjust to the changing environment of the system. The delivery process was time-consuming for conventional waterfall methods and the system was unable to ensure high business values with quality [10]. DevOps entered to remove the barriers between teams so that the development teams can deliver to operations teams more quickly with higher frequency. Agile software development methods are a set of practices that helps to improve the effectiveness of the software development processes of organizations, teams, and professionals [11]. Agile practices aim to discover user requirements and develop solutions through collaboration with cross-functional teams and end users [12].

However, agile practices have some limitations and generate complexity when development is scaled into larger settings [13].

DevOps is the combined process of *"development"* and *"operations"* practices that is used for software development to speed up the delivery process with efficiency [14]. The main promise that DevOps brings is the continuous delivery of software for fast and frequent releases [15]. Among the many continuous practices in the integration of software delivery, continuous deployment and continuous delivery are generally well accepted [16]. Due to the continuous nature of these practices, software deployment can be processed to production

* Corresponding author.
E-mail address: nasren.azad@lut.fi (N. Azad).

https://doi.org/10.1016/j.infsof.2023.107150
Received 22 April 2022; Received in revised form 11 January 2023; Accepted 18 January 2023
Available online 21 January 2023

whenever needed [17]. According to Fitzgerald et al. [16], continuous integration consists of code compilation and software build packages. The software-intensive industry is transforming towards a value-driven business paradigm with real-time adaptation [18], into which DevOps introduces the agile perspective for delivering software products in a short cycle time to reduce the delay of software processes [19].

According to the study by Shahin et al. [7] there are seven critical success factors (CSFs) that represents continuous practices. These practices include (1) testing, (2) transparency and team awareness, (3) design principles, (4) customer environment, (5) a motivated and highly skilled team, (6) accurate infrastructure, and (7) an application domain. Although their study identified seven CSFs, there was no validation for these factors [7]. Dikert et al. [6] suggested that more research should be conducted to identify the challenges and success factors in agile transformations and to figure out which factors are most important to consider.

Ram et al. [20] investigated whether CSFs can improve the performance of an organization in practice. They also suggested that successful implementation could impact the relationship between CSFs and organizational improvement. Sometimes, identifying important CSFs might not benefit organizations, since it may be more crucial to identify whether those identified factors are really critical for the organization. Moreover, the successful performance of the organizations should be linked to the success factors so that the process of success can be measured [20]. Some examples of measuring benefits for the organization include satisfaction of the customer and the successful ending of a project, both of which are intangible benefits. There may also some tangible benefits, including cost reduction, cash flow, and cash income.

Since the emergence and increased interest in DevOps, various issues have been researched academically [21]. However, there seems to be a lack of a comprehensive understanding of the critical factors for successfully using DevOps. Thus, the potential research gap that will be addressed in this paper is this lack of knowledge regarding the factors that are required to achieve successful results. To answer this call, this paper synthesizes the findings of a literature review into a framework that provides a clear, comprehensive idea about the *CSFs* of DevOps projects.

This article extends the work previously presented in [22] by incorporating three new indexing databases into the search protocol. Thus, the primary study selection and analysis have been completely redone. The results now provide a more accurate representation of the extant academic literature. As a consequence, the existing model has been improved upon, and the individual factors are discussed more thoroughly.

The remainder of this paper is structured as follows. Section 2 will briefly present the theoretical background, as well as the key concepts of DevOps and CSFs, and Section 3 will outline the objectives of the study. The results are presented in Section 4, and their implications are discussed in Section 5. Finally, the study is concluded in Section 6.

2. Theoretical background

2.1. DevOps

The concept of DevOps was first pioneered by the practitioner Patrick Debois [23]. The main trigger for this concept was the need to tackle inefficiencies in the established software development phases (i.e., the process split between development and operation) [24]. Due to these inefficiencies, both the development and operations units faced various challenges that included a poor flow of information and an unsatisfactory testing environment. The teams also faced conflicting goals in terms of agility and stability [25].

DevOps has since become a widely used development strategy that helps to minimize software development costs through implementation and adoption. The aim of DevOps is to provide continuous development and delivery for the software development process [26]. DevOps allows for collaboration between development and operations teams within the organization and provides an effective delivery process for software development [26].

DevOps represents a change in IT culture, focusing on rapid IT service delivery through the adoption of agile, lean practices in the context of a system-oriented approach. DevOps emphasizes people (and culture) and seeks to improve collaboration between operations and development teams. The implementations applied in this approach also utilize technology, especially automation tools that can leverage an increasingly programmable and dynamic infrastructure from a life cycle perspective [27].

However, it is worth noting that the term *"DevOps"* has been considered ambiguous, which has led to different interpretations and descriptions of the meaning [28–30]. For instance, according to a professional developer, as quoted in [31], *"Some people define DevOps as a culture – it's a way of doing things"*. Accordingly, there has been a series of works trying to clarify the concept. For instance, in Callanan's view [32], DevOps is an ongoing requirement for software companies. Other works have attempted to clarify the DevOps process, methodology, and characteristics [29,33]. Riungu-Kalliosaari et al. [34] and Shahin et al. [7] have suggested that acknowledging the advantages and disadvantages of DevOps implementation strategies is necessary for the organization. Erich et al. [2] argued that there are some significant characteristics of DevOps, among which collaboration, automation, and microservices are prominent.

DevOps is a development and operations approach used by professionals for various stages of creation to ensure product quality through team collaboration, allowing for the delivery of software applications based on agile and lean principles. The continuous delivery process thus enables businesses to encounter new opportunities by reducing the time for feedback from customers [35]. As a result, DevOps is best understood as a paradigm, which includes a method, a set of principles, and practices that bring a team together through communication and collaboration and facilitate effective teamwork between developers and operators [36–38]. Dyck, Penners, and Lichter [38] proposed that DevOps is an organizational approach that accelerates good communications and empathy between teams, and the team's observation on release engineering practices for continuous delivery.

The following section will focus on the DevOps life cycle and its closely related practices.

2.2. DevOps infinite loop and life-cycle

DevOps can be considered as an infinite loop of operations, whereby actions follow each other in sequence. In this DevOps infinite loop, there are seven phases, which include planning, developing code, verifying/building, testing, deploying, operating, and monitoring [39]. In the planning stage, the objective and the requirements of the software product are defined. This part is followed by the development phase, in which the software developers focus on development and code reviewing for the software. Based on the build automation and integration, the unit test takes place, and the code goes through frequent commits in code repositories. In the verification or build stage, the artifacts are checked and evaluated based on the established requirements.

For the testing phase, automation testing is performed to ensure compliance with software artifact standards. In this test phase, there could also be trial versions released that are connected to end users. For example, canary testing could be performed to reduce risk and validate the new software for a small group of people. In the deploy phase, the code goes into production. The codes need to be deployed in an automated process, and if there are any major changes, then the codes should first be deployed in production for monitoring. After the deployment stage, the operation stage begins, in which the configuration and management of software applications are handled. In the monitoring stage, the performance of the deployed applications is assessed. For this purpose, data are collected and analyzed, which helps

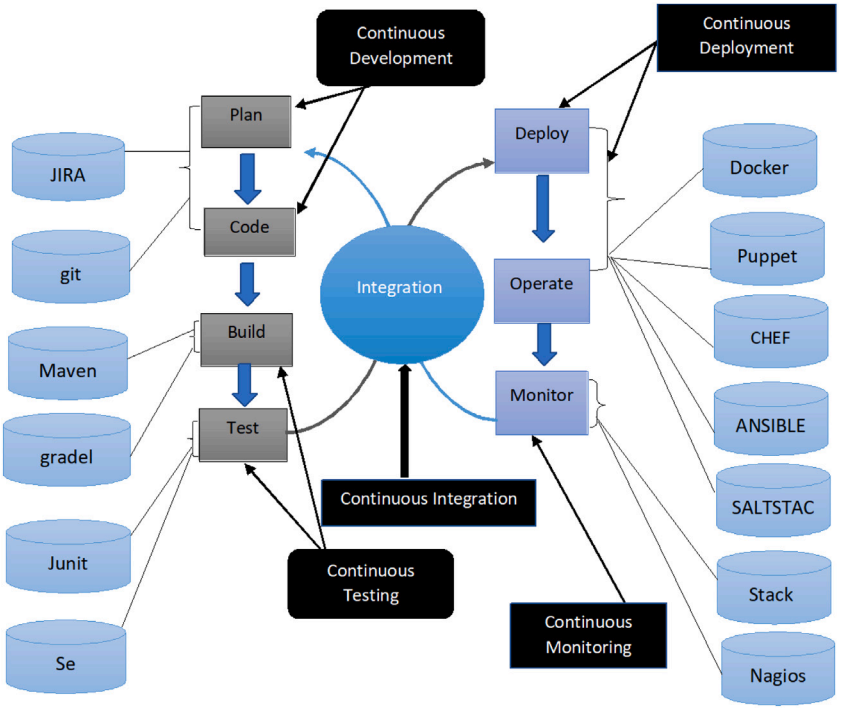


Fig. 1. DevOps life-cycle, including various phases (gray and blue boxes), tools (illustrated as cylinders), and practices.
Source: Adapted from [41].

to identify problems and elicit feedback for iterative improvement of the software [39,40].

There are various practices used in the DevOps life-cycle, which include continuous development, continuous delivery, continuous deployment, continuous testing, and continuous monitoring. Below, we discuss all the practices that are essential for DevOps process execution. Fig. 1 shows an adapted DevOps life-cycle diagram.

2.2.1. Continuous delivery

For software engineering, continuous delivery is an essential process [1]. According to Humble and Farley [18], continuous delivery is a set of principles and design practices that increases deployment frequency. As a result, this practice is welcomed by most companies, as it increases the quality, efficiency, and reliability of the product [1]. Companies are willing to introduce continuous delivery because that decreases the cycle time for software delivery [8]. According to Laukkanen et al. [8], when adopting continuous delivery in companies, several problems and issues arise. Specifically, these problems were found in design, testing, release, and integration activities, and others were related to the organizational, human, and resource aspects of continuous delivery.

2.2.2. Continuous deployment

Continuous deployment is a process that helps to deploy software codes quickly and automatically by maintaining a standard of quality [17,18,42–44]. During the process of continuous deployment, there is no need for any manual phases or developer decision making for any issues related to production deployment [8]. Skelton and O'Dell [42] have also suggested that continuous deployment is an approach in which automated tests are conducted while developers commit new features. Continuous deployment is a good way to gather feedback from users, which also reduces production costs.

2.2.3. Continuous development

Continuous development consists of the combination of two phases, which include planning and coding. In this phase, the aim of the project is determined, and the developers start generating codes for the specific application. Some of the DevOps tools that are used in this phase include JIRA and Git. This is a very essential phase in the DevOps life cycle. [45]

2.2.4. Continuous testing

Continuous testing is the stage where the developed application goes through continuous tests to detect bugs. There are various automation tools to support the testing process. According to Zimmerer [46] continuous testing has two characteristics. There could be testing for the whole life-cycle, meaning testing continuously from the beginning to the end, or strategic test automation, meaning tests are continuously adapted and executed based on the requirements [46].

2.2.5. Continuous integration

Continuous integration is a process that combines various steps, such as code compilation, code validation coverage, testing acceptance, compliance of coding standards, and deployment package building [43, 47,48]. Continuous integration initiates faster feedback for the developers by detecting integration errors [47]. To maintain a good practice, developers need to integrate their work onto a common repository on a daily basis [8,49]. According to Laukkanen et al. [8], each integration should be followed by building and testing, so that the system can remain functional after new changes are introduced by developers. This iteration of integration, source code testing, and fixing problems will efficiently increase system progress [50]

2.2.6. Continuous monitoring

Monitoring is another important stage in the DevOps lifecycle. According to Schlossnagle [51], monitoring is a system for observing and determining the correctness of the system's behavior. This is a phase in which the developers continuously monitor the performance of the software system. In the monitoring phase, all information related to the software application is given earlier, so that vital information can be processed quickly to recognize the application [51].

2.3. Critical success factors

The notion of a CSF is a management literature concept, which dates back to the beginning of the 1960s [52]. Rockart [53] explained CSFs as referring to the limited number of areas in which the successful competitive performance of the organization is ensured by satisfactory results. If the results in these areas are not satisfactory, it is likely that the overall results of the project at hand are not what was hoped for. While several researchers have defined CSFs in various dimensions [54], they can be defined here as *"the few key areas of activity in which favorable results are absolutely necessary for a particular manager to reach his goals"* [55, p. 4].

Leidecker and Bruno [56] have explained how characteristics, variables, or conditions impact the success of a firm that is competing in a specific industry, under the condition that the property is well managed, maintained, and sustained. According to Ram et al. [20], a factor can be named as a CSF when it impacts performance improvements with satisfactory results. CSFs have also been applied in the various contexts of the software engineering industry. For example, Chow and Cao [57] surveyed the CSFs of agile software projects.

In addition, there is a number of CSFs specifically pertaining to DevOps projects, which have been discussed in prior empirical studies. In recent years, literature reviews have described the DevOps concept and its adoption, implementation (with agile), and specific use, as well as its challenges, benefits, and success factors [33,58–62]. For instance, Shahin et al. [7] suggested seven CSFs for continuous practices based on testing, awareness of teams, better design principles, team motivation, skilled team building, customer environment, and better infrastructure. However, these factors have not been validated and require more clarification. Dikert et al. [6] suggested that there should be more research on the challenges and success factors in agile transformation, specifically by determining which factors should be given more attention based on transformations.

3. Research objective and process

3.1. Research question

This study aims to provide a better understanding of the CSFs of DevOps projects and the relationships between them. We will use the extant academic literature as a starting point for this inquiry. Specifically, we will focus on identifying the various success factors of DevOps and will categorize them in this paper.

This study aims to answer the following research question:

RQ What are the CSFs of DevOps projects?

For this study, we limited the scope to only the research literature (i.e., we excluded grey literature) and placed emphasis on studies reporting empirically acquired results. The rationale behind this was to focus on verified and peer-reviewed information to create a credible CSF model of DevOps projects.

3.2. Research process

We conducted a systematic literature review (SLR) to gather the primary research articles for this study. This is an efficient method to distinguish, evaluate, and analyze papers related to specific research questions. Thus, an SLR helps to collect the primary studies done on a specific topic. An SLR has predefined steps that differentiate it from unstructured literature reviews [63].

We have followed the methodology outlined by Kitchenham and Charters [63] for conducting the article search. To perform the search properly, we identified suitable keywords. After the search, we created an initial list of articles. Various criteria and parameters were then applied to filter out irrelevant articles. After that, we developed the final list of articles that would allow us to adequately answer the research question. The overall SLR process is illustrated in Fig. 2.

3.3. Search terms and strategy

The most important step in an SLR is to define the search protocol. After establishing the research question of this study, it was decided to utilize search terms and queries in publication databases to ensure that the largest number of relevant studies published on the theme could be identified. Utilizing a manual search strategy for selected publication venues could have biased the study heavily toward issues discussed in those venues; therefore, using an electronic literature search strategy was expected to provide a more comprehensive picture of the research done in the field.

After this decision, a set of relevant keywords to be used in the searches was identified. After considering abbreviations and synonyms, we used the keywords of *DevOps*, *development and operation*, *success factors*, *failures*, *drivers*, and *barriers*. We used the logical connections "AND" and "OR" to create a usable search string.

Finally, we retrieved the primary studies from the databases to identify the articles using the search strings. We used the following search string:

```
(devops OR "development and operation") AND ("success factors" OR failure OR barriers OR drivers)
```

For each publication database selected for the review, the search term was adapted to fit the database's systems and its properties.

3.4. Search resources

The use of *Scopus* and *Web of Science*, along with the *IEEE Xplore* and *ACM Digital Library* indexing databases, was deemed to be sufficient for conducting an SLR in the domain of software engineering [64]. Therefore, these publication databases were selected based on the relevance of the study scope. *Scopus* was included because it claims to have the largest database for computer science studies, and it comprises the most extensive database of citations and abstracts for peer-reviewed literature. *Web of Science* is an excellent source in the field of technology, and it covers publication forums that have a high impact in this area. *IEEE Xplore* and *ACM Digital Library* were included because they have a large number of scientific publications. Finally, *Google Scholar* was used in the snowballing phase for acquiring additional information, as it is a broad database that has an interdisciplinary academic index across the internet.

3.5. Exclusion and inclusion criteria

The following exclusion and inclusion criteria were applied to extract the final list of articles.

The inclusion criteria were as follows:

1. Study is discussing DevOps success or failure factors.
2. Study is reporting any kind of empirical evidence.
3. Study is written in English.

The exclusion criteria were as follows

1. Theoretical or literature review-based study or non-peer reviewed article, such as a column, book review, etc.
2. Study not written in English
3. Full text not possible to acquire through any media.

3.6. Selection of the final list of articles and analysis

The searches were performed at the beginning of 2022. Initially, we have found 1,909 articles from the *Scopus*, *IEEE Xplore*, *ACM Digital Library*, and *Web of Science* databases. At first, we looked at the title of each article and excluded those that were clearly not relevant for answering our research question. After excluding the articles based on their titles, we were left with 1,404 articles (including duplicates).

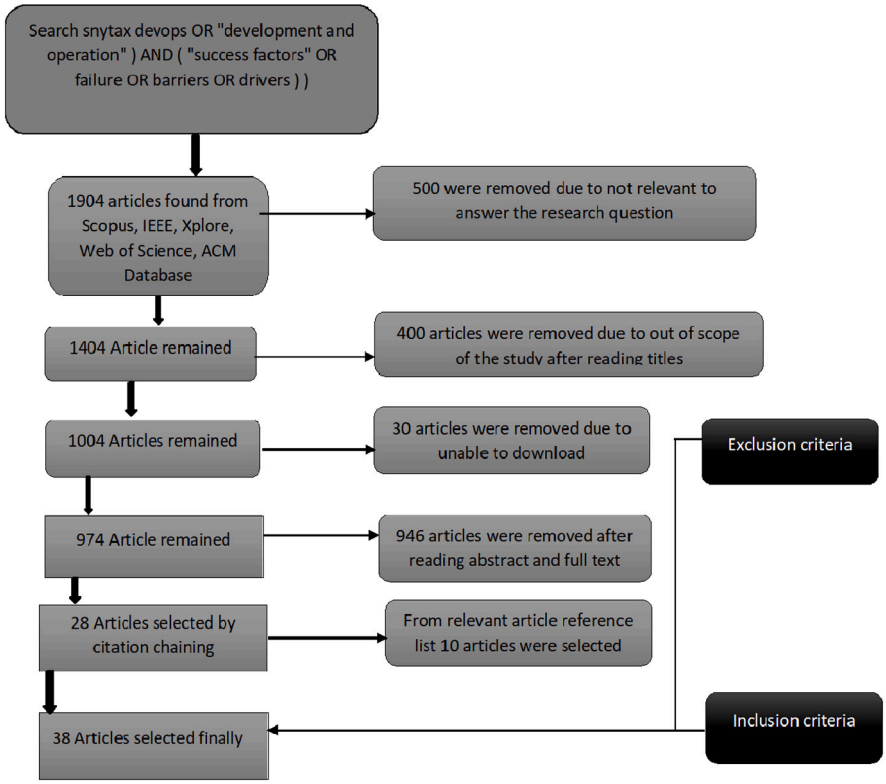


Fig. 2. SLR process used to collect the primary studies for analysis.

We then reviewed the abstracts of these articles and found that 1,004 articles were relevant to our study. Among these 1,004 articles, 30 were not possible to acquire when this study was carried out. Therefore, we downloaded 974 articles, among which 946 were excluded after screening the full texts due to an evident lack of empirical research or focus on DevOps. For the remaining papers (28), we used citation chaining (also known as snowballing) and found 10 new relevant articles. Finally, we were able to consider 38 relevant papers for this study.

After the final list of relevant primary studies was generated, all articles were read through by two researchers. The researchers used an unstructured model for extracting the data from the primary studies; more specifically, they listed all success factors found in the articles without pre-existing codes or categories. In addition, general details regarding the studies were coded. Both researchers agreed upon and verified the developed coding. In the case of disagreements, consensus was established through discussion.

After all the studies were reviewed, the researchers began categorizing the first-level codes into larger hierarchies. Finally, a three-tiered model was created, wherein the first level included the factors as they were given in the primary studies. The second level combined the factors into commonly understood logical groups. The third level then categorized the second level codes into large, remarkably differentiated perspectives. The researchers agreed upon the end result.

4. Results

4.1. Descriptive statistics

After the selection process, a total of 38 articles remained for the analysis. The year-wise distribution of the selected papers, as shown in Fig. 3, demonstrates the growing interests in the phenomenon at hand. The selected articles are shown in Table 1. The articles used a number of different approaches. In Fig. 4, a word cloud created from the terms appearing in the abstracts of the selected papers illustrates the diversity of the research themes.

From the 38 relevant primary studies, nearly 100 different factors were identified. The individual factors were categorized into larger CSFs, and then divided into the logical groups as demonstrated in Table 2. The complete list of identified factors, as well as the relevant primary studies, are given Table A.3 in Appendix. In the next step, DevOps CSFs were classified into ten groups that formed three major categories. The major categories were *Technical*, *Organizational*, and *Social and Cultural*. All categories, as well as their respective CSFs, are discussed in the following subsections.

4.2. Organizational success factors

Organizational success factors comprise an essential category, as they represent the characteristics and resources of an organization with successful DevOps practices. After reviewing the studies, we found three subcategories for organizational success factors. These included *intra-organizational collaboration and communication*, *organizational hierarchy*, and *strategic planning*.

Intra-organizational collaboration and communication. According to Diaz et al. [72,73], the collaboration between Dev and Ops teams, misalignment among departments, and the inflexibility of communication processes are the factors that might impact the success of DevOps practices. Riungu et al. [34] claim that collaboration between departments boosts communication and employee welfare, and that the production environment has a role in intra-organizational communication. According to Akbar et al. [60], real-time feedback is crucial for the successful implementation of DevOps practices, and in Chen's view [69], mechanisms for quality feedback are essential for DevOps use and can impact collaboration.

According to prior studies, when there is a lack of collaboration in teams that results in failure. It has been evident in many automotive companies where, collaboration represents the cultural intuition of an

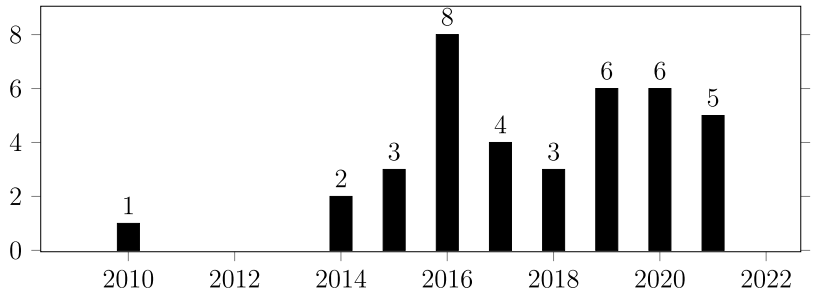


Fig. 3. Yearly distribution of selected publications.

Table 1
The publications included in this systematic review.

#	Authors	Year	*Title*
1	Abdelkebir et al. [65]	2017	An agile framework for ITS management In organizations: A case study based on DevOps
2	Akbar et al. [60]	2020	Identification and prioritization of DevOps success factors using fuzzy-AHP approach
3	Alnamlah et al. [66]	2021	The necessity of a lead person to monitor development stages of the DevOps pipeline
4	Ben Mesmia et al. [67]	2021	DevOps workflow verification and duration prediction using non-Markovian stochastic Petri nets
5	Bezemer et al. [68]	2019	How is performance addressed in DevOps?
6	Callanan and Spillane [32]	2016	DevOps: Making it easy to do the right thing
7	Chen [69]	2019	Improving the software logging practices in DevOps
8	Chen [70]	2020	Performance regression detection in DevOps
9	Claps et al. [44]	2015	On the journey to continuous deployment: Technical and social challenges along the way
10	Díaz et al. [71]	2018	DevOps in practice: An exploratory case study
11	Díaz et al. [72]	2019	Devops in practice — A preliminary analysis of two multinational companies
12	Díaz et al. [73]	2021	Why are many businesses instilling a DevOps culture into their organization?
13	Azad [74]	2022	Understanding DevOps critical success factors and organizational practices
14	Erich et al. [2]	2017	A qualitative study of DevOps usage in practice
15	Rafi et al. [75]	2022	Decision-making taxonomy of DevOps success factors using preference ranking organization method of enrichment evaluation
16	Karamitsos et al. [76]	2020	Applying DevOps practices of continuous automation for machine learning
17	Hüttermann and Rosenkranz [77]	2019	DevOps: Walking the shadowy bridge from development success to information systems success
18	Luz et al. [78]	2018	Building a collaborative culture: A grounded theory of well succeeded DevOps adoption in practice
19	Lwakatare et al. [4]	2016	An exploratory study of DevOps extending the dimensions of DevOps with practices
20	Marnewick and Langerman [79]	2020	DevOps and organizational performance: The fallacy of chasing maturity
21	Maroukian and Gulliver [80]	2020	Leading DevOps practice and principle adoption
22	Mohan and Othmane [81]	2016	SecDevOps: Is it a marketing buzzword? — Mapping research on security in devops
23	Nybom et al. [30]	2016	On the impact of mixing responsibilities between devs and ops
24	Olszewska and Waldén [82]	2015	DevOps meets formal modeling in high-criticality complex systems
25	Perera et al. [83]	2016	Evaluating the impact of DevOps practice in Sri Lankan software development organizations
26	Riungu-Kalliosaari et al. [34]	2016	DevOps adoption benefits and challenges in practice: A case study
27	Rowse and Cohen [84]	2021	A survey of DevOps in the South African software context
28	Salameh [85]	2019	The impact of DevOps automation, controls, and visibility practices on software continuous deployment and delivery
29	Shahin et al. [86]	2016	The intersection of continuous deployment and architecting process: Practitioners' perspectives
30	Shahin et al. [7]	2017	Beyond continuous delivery: An empirical investigation of continuous deployment challenges
31	Toh et al. [19]	2019	Adoption issues in DevOps from the perspective of continuous delivery pipeline
32	Trihinas et al. [87]	2018	Devops as a service: Pushing the boundaries of microservice adoption
33	Van Belzen et al. [88]	2019	Critical success factors of continuous practices in a DevOps context
34	Wahaballa et al. [89]	2015	Toward unified DevOps model
35	Wettinger et al. [15]	2014	Devopslang–Bridging the gap between development and operations
36	Wiedemann et al. [90]	2020	Understanding how DevOps aligns development and operations: A tripartite model of intra-IT alignment
37	Šmite et al. [91]	2021	Overcoming cultural barriers to being agile in distributed teams
38	Trigo et al. [92]	2022	DevOps adoption: Insights from a large European Telco

organization and that is necessary for every organization [93]. As most automotive companies rely on silos for the tasks they should perform, knowledge of responsibility is important [93]. According to Šmite et al. [91], to improve collaboration and communication among diverse cultural backgrounds employees from organizations, training should be provided so that the employees can focus on cultural differences and work together. They have also suggested that cultural training should be required when new members join the team. Training can help the employees to tackle such barriers.

Organizational hierarchy. Another subcategory of organizational success factors is organizational hierarchy. In Šmite et al. [91], the authors found that willingness to accept requests from superiors is a key organizational success factor. They also found that seeking an immediate manager's approval for team tasks and communication plays a vital

Table 2
The three major categories of critical success factors, as well as examples of identified individual factors. The full mapping of individual factors is given in the appendices.

Categories	Critical success factor	Examples of identified individual factors
Technical	*Performance Engineering*	Performance engineering complexity, Performance regressions effects, Seamless upgrades.
	Integration	Integration with existing tools, Mechanism for quality feedback, Continuous integration process.
	Build and Test Automation	Development and test automation, Drive change management, Verification.
	Infrastructure	Deployment and infrastructure provisioning, Design a common infrastructure, Accommodate legacy systems.
	DevOps as a service and its associated challenges	
Organizational	*Intra-organizational Collaboration and Communication*	Collaboration between dev and ops, Misalignment among departments, Collaboration for different roles.
	Organizational Hierarchy	Willingness to accept requests from superiors, Seeking immediate manager's approval for team task in defense of local superiors, Collaboration between departments that boosts communication and employee welfare.
	Strategic Planning	Plan next improvement, Dissimilar development, Quality responsiveness to business needs.
Social and Cultural	*Team Dynamics*	Team and process efficiency, Reluctance to reveal a lack of understanding, Fear of changes.
	Cultural shift	Cross-functional team, Change in organizational culture, Diversity and knowledge absorption

Fig. 4. A word cloud created from the abstracts of the selected 38 articles.

role in team progress. Alnamlah et al. [66] state that when there is an absence of a lead person on the team, then the efficiency of the team decreases. According to Diaz et al. [73], emphasizing the silos is also an essential factor that determines success.

Strategic planning. Strategic planning is an integral part of a successful system. It is very important for there should be an initiative, resources, a budget, and integration with strategic IT and business plans that align with the company's objectives [65]. According to several studies [15,32,38,72,76,82,89], continuous delivery and continuous development have become important practices for software development and should therefore be taken into account in strategic planning. They also found that DevOps practices for continuous delivery and continuous development promote fast and frequent delivery of changing features and ensure the quality of the product. Akbar et al. [60] stated that there should be a continuous sequencing of DevOps approaches and also found that internal DevOps events are necessary to introduce DevOps to the employees and empower practitioners to share their work, problems, and collaborative strategies with their peers. This helps the employees to accomplish clear goals that facilitate future improvement with lean leadership. According to both Šmite et al. [91] and Riungu et al. [34], issues such as company pressure, reluctance to adhere to non-feasible deadlines, and hesitancy to discuss failures and dissimilar development must be considered during strategic planning. Furthermore, system orchestration [60], change management [60], and reduction of the time to market [65,91] are also important to consider while engaging in strategic planning for the organization.

4.3. Social and cultural success factors

In DevOps, social and cultural aspects form part of the collaborative culture that is necessary for a company's success [22]. Collaboration culture seeks to improve activities by eliminating silos between development and operation teams [94]. Good communication and collaboration is necessary for the organization with a DevOps open culture [94]. When we talk about collaboration, we may only think about the Dev and Ops teams' engagement in the collaboration process. Basically, collaboration means more than that, since good collaboration includes relationships among the stakeholders and different clients who are both inside and outside of the organization. When a company has good collaboration, this leads the organization to achieve good progress for successful implementation of DevOps practices [95]. According to Lwakatare et al. [4], empathy between teams, working in a good environment, and supporting teams are some of the defining attributes of a better culture of collaboration in an organization.

Thus, social and cultural factors are among the most important for DevOps critical success. Under these social and cultural categories, we have found two subcategories named *team dynamics* and *cultural shift*. These factors are discussed below.

Team dynamics. From the results of our literature review, we have found that team dynamics play a vital role in DevOps success. According to Diaz et al. [73,81], team and process efficiency have some patterns that impact the organizational and cultural perspective for DevOps practices. Šmite et al. [91] have suggested that reluctance to reveal a lack of understanding in asking questions impacts team efficiency. They also claim that offshore teams (e.g., those in Sweden and India) with DevOps practices might be reluctant to voice criticism or propose alternatives to perceived directives from superiors. Akbar

et al. [60] found that some factors that influence DevOps practices in software organizations are cross-functional teams and established, skilled DevOps teams. Tsanos et al. [96] and Kolfschoten et al. [97] have explained how mutuality, trust, commitment, and mutual respect among teams impact the team efficiency for DevOps. Yoon et al. [98] conducted a survey on Korean IT organizations and found that clear role setting and methods for solving common issues in teams are critical for team dynamics. According to Wettinger et al. [15], efficient collaboration in teams has a positive impact on team dynamics. Kolfschoten et al. [97] found that understanding and respect for each other and their respective organizations speed up teams' efficiency. Nybom et al. [30] conducted a survey with 14 engineers of an organization and found several benefits for team efficiency, such as improved collaboration, trust, smoother workflow, shared responsibilities, and common ways of working. According to Claps et al. [44], team roles, team coordination, team experience, and source code control impact the success of DevOps.

Moreover, for team dynamics, a company should emphasize knowledge sharing, which is part of the collaboration process within the organization. When the developers and operations personnel work together, they should try to make product documentation accessible and understandable, so that both teams can get a sense of the whole idea. This sharing process can facilitate both teams through knowledge management [99].

Cultural shift. After reviewing the literature, we identified cultural shift as another important component of DevOps CSFs. According to Riungu et al. [34], cultural shift's impacts on DevOps adoption are bigger than technical or process issues. They found that a company should work towards a common goal to achieve the overall objectives of a project. Akbar et al. [60] stated that companies should emphasize culture rather than tools that will help with task execution [67]. Diaz et al. [72] conducted an interview of stakeholders of 11 (multinational) software-intensive companies and found that organization and employee culture impact the success of an organization. Finally, cultural norms [97] and understanding of culture [30] have also been reported as important factors for DevOps success.

According to Šmite et al. [91], culturally distinct behavior can be experienced in the organization and cause misunderstandings and prevent agile ways of working in distributed DevOps teams. The offshore teams based in India and Sweden experienced cultural barriers and differences that delayed the agile adoption approaches in collaboration with the offshore teams. According to the findings reported by Šmite et al. [91], some cultural characteristics were very rigid and unchangeable. Prior research shows that distributed teams are left behind to experience, experiment, and adjust their ways of working, as it is believed that cultural awareness is gained only from experience. According to Casey [100], the importance and necessity of cultural training were unrecognized while valuable time, endless effort, and valuable resources were wasted. Therefore it is essential for companies and distributed teams to have cross-cultural communication courses, so that existing values and communication resources can help DevOps culture emerge in the organization [101]. According to Šmite [91], cultural training is important, and the personnel who can match with the team culture can significantly impact project success [91,102].

4.4. Technical success factors

Technological success factors are based on the technological context of a company, which represents the relevant technologies that have been used by the company, as well as new technologies available in the market [103]. We have found five subcategories for the technical success factors, which we call *performance engineering*, *integration*, *build and test automation*, *infrastructure*, and *DevOps as a service*. These are discussed below.

Performance engineering. Bezemer et al. [68] described how performance can be addressed in industrial DevOps projects. According to them, the complexity of performance engineering approaches is one major barrier to DevOps success. Therefore, they suggested that performance engineering approaches need to be lightweight. They also supported the integration of performance engineering approaches with the tools of the DevOps pipeline. In the context of DevOps, the adoption of performance engineering practices is low. This is because these approaches are not designed for DevOps contexts. The existing tools and approaches that are used for DevOps settings do not integrate properly with existing performance engineering processes. According to Bezmer at al. [68], integrated solutions may help to improve estimations, allocate teams, and manage team productivity and project performance. In Chen's view [70], performance regression often slips to operations while using DevOps. Thus, techniques are needed to detect performance regression at the source code level.

Integration. According to Junior et al. [104], integrated solutions for the combined processes of DevOps help DevOps succeed. In [77], the authors state that AISD (Agile Information System Development) and closer integration of ISD with IS should be maximized for success. In Claps et al.'s perspective [44], continuous integration processes and challenges of operations and support may impact DevOp's success.

Build and test automation. According to Bird [93], automation is a process that extends software operations, through which the machines are automated to ensure faster production and delivery of products to meet users' needs, expectations, and demands. Automation has benefited the manufacturing industries by providing solutions for time-consuming activities. The involvement of DevOps will thus help companies to achieve success. According to Humble and Farley [18], four steps should be followed for a successful automation process, which include steering inefficiency, testing, deploying, and operating. Thus, automation will promote transparency, easier deployment, and quicker release of the service to the market [18].

Riungu et al. [34] reported that test automation and automatic testing technique improvement can affect the success of DevOps. Furthermore, the verification of soundness [67], specific tools for automation [67], automation techniques [15], and building and process automation [73] may impact DevOps success.

Infrastructure. According to Claps et al. [44], infrastructure is vital for software process integration. Specifically, infrastructure helps in continuous development and solves problems regarding hardware and software issues. The infrastructure allows software products to be deployed when they are needed. As stated by Luz et al. [78], development and infrastructure provisioning are needed for better execution of DevOps practices.

DevOps as a service. Microservices are essential for the future of DevOps [87,105]. The microservices approach is used for handling complex systems that require changes in high speed [105]. Microservices are thus essential because they can share the core of the operating system, which helps to ensure faster deployments in the cloud without interrupting performance. In this way, the application services can work better with good interfaces and can avoid synchronous and blocking situations whenever needed. When DevOps ideology is adopted for the microservices, then separate software teams are responsible for various aspects of the end applications. This process allows both Dev and Ops teams to develop, test, and overcome failure and try to scale production independently [87].

For continuous delivery and continuous integration of the frequent changes in the codes, it is not possible to achieve a process at once. When the steps are divided into small units, then that is feasible to develop, deploy, and manage the process. In this way, when a developer works on a microservice, they can understand, correct, and write new codes without knowing what codes have already been written by their

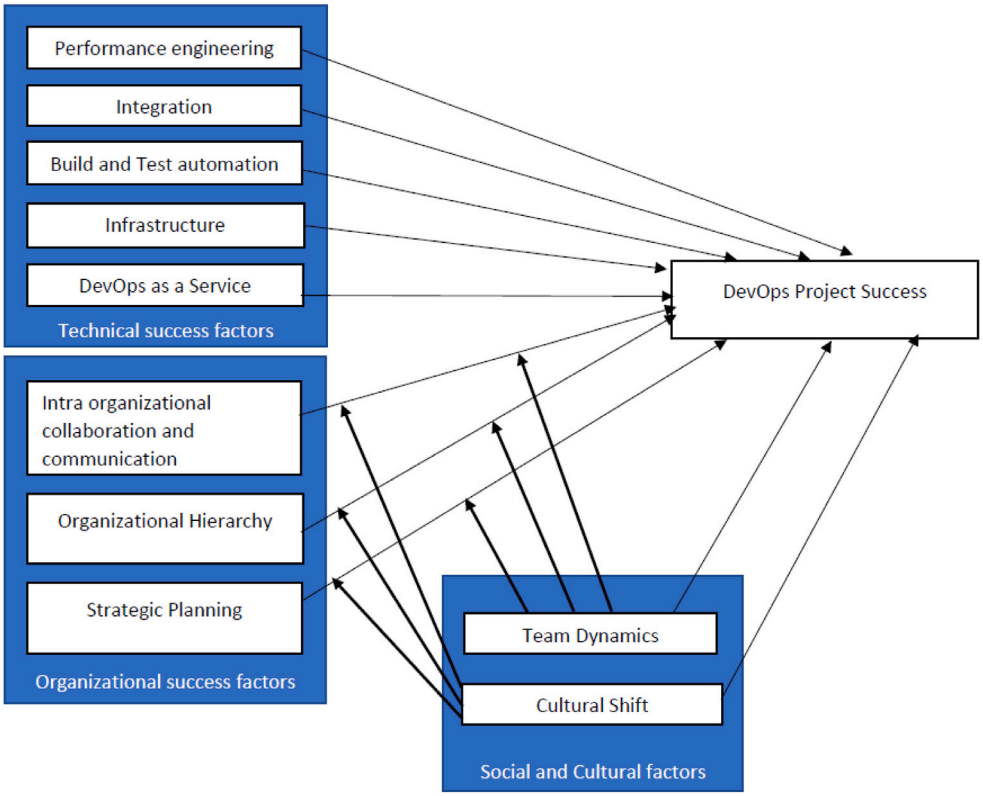


Fig. 5. The synthesized framework of critical success factors.

peers. This is because API is used for this purpose, and there is no need for data sharing, data structure, or other internal objects [87].

Microservices are thus an integral part of DevOps culture. The adoption of the microservice thus supports the teams in their rapid development and deployment of modern, distributed, and cloud-based applications. The use of microservices is a new phenomenon for the cloud computing industry. Through this process, the DevOps team works together and manages the life cycle of the application and goes through continuous releases. DevOps as a service platform thus facilitates software teams with respect to the effective use of microservices [87].

4.5. Synthesized critical success factor model

To address the lack of a comprehensive model of the CSFs of DevOps, we have developed a synthesized model from our findings. The model is built on top of the identified factors and categories. The model is shown in Fig. 5.

In this model, we suggest that technical factors and organizational factors (intra-organizational collaboration, organization hierarchy, and strategic planning) would directly impact DevOps success. These two relationships are supported by the technology-organization-environment (TOE) framework [106,107], which suggests that both technological and organizational factors directly impact the adoption of technology in an organization. These two relationships have also been empirically verified in prior information systems and management science literature [108–110]. Following these findings, we suggest that these two factors can also drive DevOps cloud-based organizations. We propose that the social and cultural factors (team dynamics and cultural shift) would moderate the effects of both technical and organizational factors on DevOps success [111,112]. The justification of such moderating effects is rooted in the differences among different cultures in terms of individualism/collectivism, power distance, uncertainty avoidance, and masculinity/femininity [113]. Prior literature on IT adoption has also empirically shown culture to be a moderator. This implies that the effects of technical and organizational factors might vary depending on the culture. In addition, social and cultural factors might directly also impact DevOps success.

However, we note that the actual relationships can be explored in future research by collecting empirical data from organizations that use DevOps. Furthermore, it might be that not all important CSFs have already been identified, as the current review is based only on peer-reviewed literature, and the model might need to be updated after the field has matured.

In addition, we hope the proposed framework could be used as a starting point to understand DevOps CSFs and how to tackle challenges in an organization. Specifically, this model can help raise awareness of DevOps practices among practitioners and DevOps professionals. Despite its usefulness, we think that empirical data through surveys and interviews with different organization professionals working with DevOps will be needed to validate this model. This will also ensure the applicability of the proposed framework. After empirical validation, and when the field matures more, new factors can be added, or some factors that will not have a significant impact on success can be removed from the framework.

5. Discussion

5.1. Key findings and observations

This study sets forth a research question to identify the CSFs presented in the extant research for successful DevOps operations in organizations. After reviewing 38 primary studies and analyzing them, we identified 10 CSFs. The factors were divided into three main categories, and their inter-relationship is hypothesized as a comprehensive model. This model is illustrated in Fig. 5.

In addition, we summarize our key observations in the following three points:

First, the ten factors identified are logically categorized into technical, organizational, and social and cultural groups. Based on the technological context, there are internal technologies along with external technologies that impact different aspects of the organization's decision-making process [103]. If we consider the organizational success factors to be the resources that play a significant role that impacts the adoption and implementation process of a new innovation [114], then DevOps entrance into the company is a new innovation in the organization. Our findings also suggest that the size of the organization and management support also plays a significant role in critical decision-making for DevOps adoption for CSFs. When compared to similar studies done with agile or plan-driven methods (c.f. [115–117]), the role of technically-oriented factors is overemphasized. Nevertheless, this might indicate a major characterizing difference of technologies between DevOps and more traditional software development process models.

Second, while DevOps is a relatively young software process model, there is already remarkable research built on it. Whereas this body of literature is clearly smaller than the extant work addressing, for instance, agile methods, it is emerging swiftly. Furthermore, the extant literature is already suggesting CSFs and drives to identify them (cf. [60]), but there is a clear lack of a comprehensive model addressing all factors and their relationships. In addition, the CSF literature on DevOps is fragmented in addressing the phenomenon through narrowly defined lenses, and there are only a few studies that take a broader view of the phenomenon. While this study aims to answer this call, further work is still needed to verify and enlarge this comprehensive understanding.

Third, some of the critical success factors often seen in other areas such as personal competencies, training, and top management support, were not present among the primary studies addressed. There are several possible explanations for their absence, ranging from their mitigated role (e.g., top management support might not be important at all after the prerequisites for DevOps have been set, as was found in the project cost estimates [118]) to the extant research being too heavily focused on DevOps-specific characteristics. Further research is needed to address this.

Fourth, the findings could be applied to different sizes of companies. Specifically, the identified success factors could be applied to DevOps cultural changes and mindsets for organizations. CSF classification could have an impact on IT applicability, assessment of the public sector, and different sizes of companies in industries. DevOps's focus is to create a company culture that breaks down the silos and barriers in the organization by putting more effort into better coordination and communication with focused team objectives.

5.2. Threats to validity

Validity considerations are required while conducting an SLR [119]. For our research, we have identified some threats to validity. These issues include search term limitations, search engine selection processes, data extraction inaccuracy, and inclusion and exclusion criteria with researcher bias. We discuss the threats as suggested in [120].

Internal validity. In order to make sure that the systematic review is repeatable, we have carefully defined the search terms and inclusion and exclusion criteria. Issues regarding search terms, search engines, and other biases can lead to an incomplete set of initial data sources. To identfy relevant studies, dfferent search terms pointing to the similar concepts were used, followed by a manual search in the references other than the initial pool. To control for threats, we relied on four databases. Based on the strategy, we could say that an adequate and inclusive search has been done for the study. The missing publication percentage will be low if there are any missing from the study. Furthermore, while applying the inclusion and exclusion criteria, the researchers' judgment might have affected the selection process. There could also have been personal bias during the process. Thus, two researchers participated in the paper selection process to remove this kind of bias.

External validity. The selection process chose only studies written in English, and those conducted in another language were excluded from the study. In addition, the study is limited by the used search engines, databases, and keywords. Thus, the question arises as to whether we have selected papers that cover all the literature on DevOps CSFs. To mitigate this issue, we have aimed to extract sufficient information from the selected studies, used the most common and largest databases and broad search terms.

For reliability, there could be a question of whether we have drawn correct conclusions through rigorous and repeatable processes. To ensure the reliability of the process, one researcher reviewed the papers, extracted the data, and created the themes. Another researcher reviewed all the extracted data and themes, which mitigated the bias in data extraction and analysis. Any disagreements were resolved by discussions between researchers. By following this procedure, the study ensured that the results of similar studies will not have huge deviations from our reported outcomes.

5.3. Further work

From our literature review, we have found some topics that may require extended studies. These topics could be considered as a future research agenda for examining CSFs of DevOps projects.

Research model development and test for CSF. Our literature review has identified several fruitful avenues for future work in this domain. Our synthesized framework can be used as the guiding framework to investigate CSFs in the future. For example, our framework could help build research models to answer possible research questions such as "What are the most important technical factors?", "What are the most important organizational factors?", and "What are the most important social-cultural factors?" Empirical data can be collected to investigate the factors, and structural equation modeling techniques can be used to test the research models to answer the described research questions.

Following a theory-guided approach for CSF. We observed that prior research lack useful frameworks when investigating DevOps success. In our literature review, we were able to identify limited papers that used a theory guided approach, though the topic of DevOps had attracted a significant amount of interest among researchers and practitioners. The organizational and social factors can be tied together with theories from organizational behavior and communication. Therefore, future research can borrow theories from these disciplines and focus on theory-guided approaches to investigate DevOps success.

Developing scales for success measurement. There are still opportunities to develop scales for measuring DevOps success and many of the factors identified in our literature review. From the prior studies, we have found that very few studies developed or validated scales for measuring factors related to DevOps success. These scales can be used in surveys for collecting quantitative data among DevOps professionals. Such data are needed to test the research models (or hypotheses) as described earlier.

Table A.3
Identified individual factors, their categorization into CSFs and division into three major categories.

Categories	Critical success factor	Extracted individual factors from the primary studies
Technical	*Performance Engineering*	Performance engineering complexity [68], Performance regressions effects [70], Lightweight approaches for performance engineering [68], Software release [32], Seamless upgrades [44]
	Integration	Smooth integration with existing tools [68], Limited mechanism for quality feedback [69], AISD and a closer integration of ISD with IS should be maximized [77], Integrate the challenges of operations and support [61], Immature system [61], Continuous integration process [44], Continuous integration and deployment [60], Continuous integration [72,76,85]
	Build and Test Automation	Test automation improvement [34,92], Drive change management [65], Verification (soundness) [67], Specific tools to automate building [78], Automatic testing techniques [60], Development and test automation [32], Process automation [90], Automation [15,83,85], Lack of process automation [73]
	Infrastructure	Deployment and infrastructure provisioning [78], Infrastructure [44], Design a common infrastructure [60], Accommodate legacy systems [60]
	DevOps as a Service and its associated challenges [87]	
Organizational	*Intra-Organizational Collaboration and Communication*	Collaboration between dev and ops [73,74,92], Misalignment among departments [72], Inflexibility of communication processes [72], Communication structures [91], Miscommunication between development and operations [72], Collaboration [84,92], Collaboration for different roles [2,74], Experienced support [84], Help from cross-organizational team [72]
	Organizational Hierarchy	Willingness to accept requests from superiors [74,91], Absence of a lead person [66], Attempt matrix organization and transparency [60], Seeking immediate manager's approval for team task in defense to local superiors [91], Collaboration between departments that boosts communication and employee welfare [34]
	Strategic Planning	Continuous delivery [15,72,82], Keep to the sequencing of the DevOps approach [60], Internal DevOps event [60], Demonstrating lean leadership behavior [60], Plan next improvement [60], Quality responsiveness to business needs [83,121], Deployment risk by identified tooling obstacle [30], Company pressure [91], Reluctant to warn about non-feasible deadlines [91], Reluctance to discuss failure [91], Dissimilar development [34], Production environments [34], System quality and development [82,121], Quality responsiveness to business needs [83], Agility to new technology [83], Use system orchestration [60], Real-time feedback [60], Limited mechanism for quality feedback [69], DevOps security pipeline [60], Reduce the time to market [72], Emphasizing the silos [73], Clear role setting [98]
Social and Cultural	*Team Dynamics*	Team and process efficiency [73], Reluctant to reveal a lack of understanding [91], Reluctant to expose problems [91], Reluctance to voice criticism or propose alternatives to perceived directives from superiors [91], Fear of changes [89], Blaming team mates [89], Efficient collaboration [15], Understanding and respect for each other and their respective organizations [97], Shared responsibilities [30], Enhance collaboration and communication [80,88], Improved collaboration [30], Common ways of working [30], Interpersonal skills [2], Team roles [44], Team coordination [44,88], Team collaboration [81], Team experience of source code control [44], Cross functional team integration [90]
	Cultural Shift	Cross-functional team [60], Established skilled DevOps teams [60], Change in organizational culture [79], Lack of guidance [61], Isolation in teams [89], Getting quick response time to business demands [72], Shift of mindset [80], Organization and employees culture [72,74], Culture [83], Cultural norms, values mutual respect, understanding, trust among teams [97], Cultural mindset [84], Cultural impacts [30], Collaboration culture [98], Emphasize culture rather than tools [60], Diversity and knowledge absorption [98]

Case studies with DevOps companies. More research conducted with different methods on DevOps success factors is required. Indeed, we were able to identify some research papers that included case studies. We have observed that the scope and interest for case studies is huge for practitioners and researchers, and many companies are emphasizing the use of DevOps and implementing the culture for the betterment of their software delivery. After going through different case studies from prior literature, we suggest that more case studies using mixed method approaches should be conducted for a better understanding of the CSFs for DevOps projects.

Empirical studies on how to tackle DevOps challenges. From our literature review, we found a very limited number of empirical studies that discuss how DevOps professionals tackle the challenges related to CSFs. Thus, future empirical research can be conducted to identify these strategies.

Comparison of DevOps practices in different countries. We found a very limited number of empirical studies that discuss how DevOps practices may differ between countries. In our comprehensive framework, we have observed that cultural factors can have an impact on DevOps success. Thus, future research can conduct in-depth studies in different countries to identify the possible differences in terms of CSFs.

Grey literature review. This systematic literature study provides only an academic view of the phenomenon. There is a gap among research areas focused on DevOps CSFs, especially between industrial and academic fields. A gray literature review could clarify the misconception of DevOps practices by analyzing, classifying, and combining the corresponding academic state-of-the-art and industrial state-of-the-art DevOps cultures. Thus, a gray literature review would provide an alternative, more practitioner-focused view of the subject.

6. Conclusions

DevOps is a phenomenon that combines tools, organizational culture, practices, and collaboration. DevOps is supporting the software

industry in achieving better performance and development processes. In this paper, we attempt to provide a clear understanding of the CSFs of DevOps. The final set of 38 papers was reviewed in detail to extract valuable and relevant information for the study. We have categorized the extracted factors into three main groups, which include technical, organizational, and social & cultural success factors. The main factors have some sub-success factors. The proposed framework is based on prior literature and provides a comprehensive view of CSFs and how they impact the success of DevOps practices in organizations. This study's findings will help researchers and practitioners to enhance their understanding of CSFs and know beforehand how to handle DevOps certain issues in organizations.

CRediT authorship contribution statement

Nasreen Azad: Conceptualization, Methodology, Writing – Original draft. **Sami Hyrynsalmi:** Conceptualization, Methodology, Writing – review & editing, Supervision.

Declaration of competing interest

No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.infsof.2023.107150.

Data availability

Data will be made available on request.

Appendix. Extracted individual factors

See Table A.3.

References

[1] L. Chen, Continuous delivery: overcoming adoption challenges, J. Syst. Softw. 128 (2017) 72–86.

[2] F.M. Erich, C. Amrit, M. Daneva, A qualitative study of devops usage in practice, J. Softw. Evol. Process 29 (2017) e1885.

[3] R. Colomo-Palacios, E. Fernandes, P. Soto-Acosta, X. Larrucea, A case analysis of enabling continuous software deployment through knowledge management, Int. J. Inf. Manage. 40 (2018) 186–189.

[4] L.E. Lwakatare, P. Kuvaja, M. Oivo, An exploratory study of devops extending the dimensions of devops with practices, ICSEA 104 (2016) 2016.

[5] H. Alahyari, R.B. Svensson, T. Gorschek, A study of value in agile software development organizations, J. Syst. Softw. 125 (2017) 271–288.

[6] K. Dikert, M. Paasivaara, C. Lassenius, Challenges and success factors for large-scale agile transformations: A systematic literature review, J. Syst. Softw. 119 (2016) 87–108.

[7] M. Shahin, M.A. Babar, M. Zahedi, L. Zhu, Beyond continuous delivery: an empirical investigation of continuous deployment challenges, in: 2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM, IEEE, 2017, pp. 111–120.

[8] E. Laukkanen, J. Itkonen, C. Lassenius, Problems, causes and solutions when adopting continuous delivery—a systematic literature review, Inf. Softw. Technol. 82 (2017) 55–79.

[9] J. Järvinen, T. Huomo, T. Mikkonen, P. Tyrväinen, From agile software development to mercury business, in: C. Lassenius, K. Smolander (Eds.), Software Business. Towards Continuous Value Delivery, Springer International Publishing, Cham, 2014, pp. 58–71.

[10] J. Highsmith, A. Cockburn, Agile software development: The business of innovation, Computer 34 (2001) 120–127.

[11] P. Abrahamsson, O. Salo, J. Ronkainen, J. Warsta, Agile software development methods: Review and analysis, 2017, ArXiv abs/1709.08439.

[12] R. Krawatzeck, B. Dinter, Agile business intelligence: Collection and classification of agile business intelligence actions by means of a catalog and a selection guide, Information Systems Management 32 (2015) 177–191.

[13] K. Petersen, C. Wohlin, A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case, J. Syst. Softw. 82 (2009) 1479–1490.

[14] I.M. Sebastian, J.W. Ross, C. Beath, M. Mocker, K.G. Moloney, N.O. Fonstad, How big old companies navigate digital transformation, in: Strategic Information Management, Routledge, 2020, pp. 133–150.

[15] J. Wettinger, U. Breitenbücher, F. Leymann, Devopslang–bridging the gap between development and operations, in: European Conference on Service-Oriented and Cloud Computing, Springer, 2014, pp. 108–122.

[16] B. Fitzgerald, K.-J. Stol, Continuous software engineering: A roadmap and Agenda, J. Syst. Softw. 123 (2017) 176–189.

[17] P. Rodríguez, A. Haghighatkhah, L.E. Lwakatare, S. Teppola, T. Suomalainen, J. Eskeli, T. Karvonen, P. Kuvaja, J.M. Verner, M. Oivo, Continuous deployment of software intensive products and services: A systematic mapping study, J. Syst. Softw. 123 (2017) 263–291.

[18] J. Humble, D. Farley, Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation, Pearson Education, 2010.

[19] M.Z. Toh, S. Sahibuddin, M.N. Mahrin, Adoption issues in devops from the perspective of continuous delivery pipeline, in: Proceedings of the 2019 8th International Conference on Software and Computer Applications, 2019, pp. 173–177.

[20] J. Ram, D. Corkindale, M.-L. Wu, Implementation critical success factors (csfs) for erp: Do they contribute to implementation success and post-implementation performance? Int. J. Prod. Econ. 144 (2013) 157–174.

[21] M. Gall, F. Pigni, Taking devops mainstream: a critical review and conceptual framework, Eur. J. Inform. Syst. (2021) 1–20, http://dx.doi.org/10.1080/0960085X.2021.1997100.

[22] N. Azad, S. Hyrynsalmi, What are critical success factors of devops projects? a systematic literature review, in: X. Wang, A. Martini, A. Nguyen-Duc, V. Stray (Eds.), Software Business, Springer International Publishing, Cham, 2021, pp. 221–237.

[23] P. Debois, Agile infrastructure and operations: how infra-gile are you? in: Agile 2008 Conference, IEEE, 2008, pp. 202–207.

[24] O. Gotel, D. Leip, Agile software development meets corporate deployment procedures: stretching the agile envelope, in: International Conference on Extreme Programming and Agile Processes in Software Engineering, Springer, 2007, pp. 24–27.

[25] J. Iden, B. Tessem, T. Päivärinta, Problems in the interplay of development and it operations in system development projects: A delphi study of Norwegian it experts, Inf. Softw. Technol. 53 (2011) 394–406.

[26] M. Sacks, Devops principles for successful web sites, in: Pro Website Development and Operations, Springer, 2012, pp. 1–14.

[27] I. technology glossary, Gartner it glossary, 2022.

[28] J. Smeds, K. Nybom, I. Porres, Devops: a definition and perceived adoption impediments, in: International Conference on Agile Software Development, Springer, 2015, pp. 166–177.

[29] J. Roche, Adopting devops practices in quality assurance, Commun. ACM 56 (2013) 38–43.

[30] K. Nybom, J. Smeds, I. Porres, On the impact of mixing responsibilities between devs and ops, in: International Conference on Agile Software Development, Springer, 2016, pp. 131–143.

[31] N. Azad, S. Hyrynsalmi, Devops challenges in organizations: Through professional lens, in: Software Business. 13th International Conference, ICSOB 2022, Bolzano, Italy, November (2022) 8-11 Proceedings, Springer Nature, 2022, pp. 1–16.

[32] M. Callanan, A. Spillane, Devops: making it easy to do the right thing, IEEE Softw. 33 (2016) 53–59.

[33] R. Jabbari, N. bin Ali, K. Petersen, B. Tanveer, What is devops? a systematic mapping study on definitions and practices, in: Proceedings of the Scientific Workshop Proceedings of XP2016, 2016, pp. 1–11.

[34] L. Riungu-Kalliosaari, S. Mäkinen, L.E. Lwakatare, J. Tiihonen, T. Männistö, Devops adoption benefits and challenges in practice: a case study, in: International Conference on Product-Focused Software Process Improvement, Springer, 2016, pp. 590–597.

[35] R.d.S. Barros, et al., DevOps Technologies for Tomorrow (Ph.D. thesis), 2016.

[36] L. Bass, R. Jeffery, H. Wada, I. Weber, L. Zhu, Eliciting operations requirements for applications, in: 2013 1st International Workshop on Release Engineering, RELENG, IEEE, 2013, pp. 5–8.

[37] C.A. Cois, J. Yankel, A. Connell, Modern devops: optimizing software development through effective system interactions, in: 2014 IEEE international professional communication conference, IPCC, IEEE, 2014, pp. 1–7.

[38] A. Dyck, R. Penners, H. Lichter, Towards definitions for release engineering and devops, in: 2015 IEEE/ACM 3rd International Workshop on Release Engineering, IEEE, 2015, p. 3.

[39] A. Alnafessah, A.U. Gias, R. Wang, L. Zhu, G. Casale, A. Filieri, Quality-aware devops research: Where do we stand? IEEE Access 9 (2021) 44476–44489.

[40] L. Bass, I. Weber, L. Zhu, DevOps: A Software Architect's Perspective, Addison-Wesley Professional, 2015.

[41] Arvind, Devops lifecycle:everything you need to know about devops lifecycle phases, 2022.

[42] M. Skelton, C. O'Dell, Continuous Delivery with Windows And. NET, O'Reilly Media, 2016.

[43] M. van Belzen, J. Trienekens, R. Kusters, Critical success factors of continuous practices in a devops context, in: A. Siarheyeva, C. Barry, M. Lang, H. Linger, C. Schneider (Eds.), Information Systems Development: Information Systems Beyond 2020, ISD2019 Proceedings, ISEN Yncréa Méditerranée, 2019, pp. 1–12.
[44] G.G. Claps, R.B. Svensson, A. Aurum, On the journey to continuous deployment: Technical and social challenges along the way, Inf. Softw. Technol. 57 (2015) 21–31.
[45] L.E. Lwakatare, P. Kuvaja, M. Oivo, Relationship of devops to agile, lean and continuous deployment, in: P. Abrahamsson, A. Jedlitschka, A. Nguyen Duc, M. Felderer, S. Amasaki, T. Mikkonen (Eds.), Product-Focused Software Process Improvement, Springer International Publishing, Cham, 2016, pp. 399–415.
[46] P. Zimmerer, Strategy for continuous testing in idevops, in: Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings, 2018, pp. 532–533.
[47] B. Fitzgerald, K.-J. Stol, Continuous software engineering and beyond: trends and challenges, in: Proceedings of the 1st International Workshop on Rapid Continuous Software Engineering, 2014, pp. 1–9.
[48] R. Vonk, A Study into Critical Success Factors During the Adoption and Implementation of Continuous Delivery and Continuous Deployment in a DevOps Context (Master's thesis), Open University of the Netherlands, 2020.
[49] M. Fowler, M. Foemmel, Continuous integration, 2006.
[50] D. Stahl, J. Bosch, Automated software integration flows in industry: A multiple-case study, in: Companion Proceedings of the 36th International Conference on Software Engineering, 2014, pp. 54–63.
[51] T. Schlossnagle, Monitoring in a devops world, Commun. ACM 61 (2018) 58–61.
[52] R.A. Dickinson, C.R. Ferguson, S. Sircar, Critical success factors and small business, Am. J. Small Bus. 8 (1984) 49–57, http://dx.doi.org/10.1177/104225878400800309.
[53] J.F. Rockart, Chief executives define their own data needs, Harv. Bus. Rev. 57 (1979) 81–93.
[54] K. Grunert, C. Ellegaard, The Concept of Key Success Factors: Theory and Method, WorkingPaper, MAPP - Centre for Research on Customer Relations in the Food Sector, 1992.
[55] C.V. Bullen, J.F. Rockart, A Primer on Critical Success Factors, Working Papers 1220-81. Report (Alfred P. Sloan School of Management. Center for Information Systems Research) ; No. 69, Massachusetts Institute of Technology (MIT), Sloan School of Management, 1981, https://EconPapers.repec.org/RePEc:mit:sloanp:1988.
[56] J.K. Leidecker, A.V. Bruno, Identifying and using critical success factors, Long Range Plann. 17 (1984) 23–32.
[57] T. Chow, D.-B. Cao, A survey study of critical success factors in agile software projects, J. Syst. Softw. 81 (2008) 961–971, http://dx.doi.org/10.1016/j.jss.2007.08.020, agile Product Line Engineering.
[58] W. Hussain, T. Clear, S. MacDonell, Emerging trends for global devops: a New Zealand perspective, in: 2017 IEEE 12th International Conference on Global Software Engineering, ICGSE, IEEE, 2017, pp. 21–30.
[59] N. Kerzazi, B. Adams, Who needs release and devops engineers, and why? in: Proceedings of the International Workshop on Continuous Software Evolution and Delivery, 2016, pp. 77–83.
[60] M.A. Akbar, S. Mahmood, M. Shafiq, A. Alsanad, A.A.-A. Alsanad, A. Gumaei, Identification and prioritization of devops success factors using fuzzy-ahp approach, Soft Comput. (2020) 1–25.
[61] M. Muñoz, M. Negrete, J. Mejía, Proposal to avoid issues in the devops implementation: A systematic literature review, in: World Conference on Information Systems and Technologies, Springer, 2019, pp. 666–677.
[62] M. Van Belzen, D. DeKruiff, J.J. Trienekens, Success factors of collaboration in the context of devops, in: Proceedings of the 12th IADIS International Conference Information Systems 2019, IS 2019, 2019, pp. 26–34.
[63] B.A. Kitchenham, S. Charters, Guidelines for Performing Systematic Literature Reviews in Software Engineering. Version 2.3, EBSE Technical Report EBSE-2007-01, Keele University, Keele, Staffs, United Kingdom, 2007.
[64] B. Kitchenham, P. Brereton, A systematic review of systematic review process research in software engineering, Inf. Softw. Technol. 55 (2013) 2049–2075.
[65] S. Abdelkebir, Y. Maleh, M. Belaissaoui, An agile framework for its management in organizations: a case study based on devops, in: Proceedings of the 2nd International Conference on Computing and Wireless Communication Systems, 2017, pp. 1–8.
[66] B. Alnamlah, S. Alshathry, N. Alkassim, N. Jamail, The necessity of a lead person to monitor development stages of the devops pipeline, Indonesian J. Electr. Eng. Comput. Sci. 21 (2021) 348, http://dx.doi.org/10.11591/ijeecs.v21.i1.pp348-353.
[67] W. Ben Mesmia, M. Escheikh, K. Barkaoui, Devops workflow verification and duration prediction using non-Markovian Stochastic Petri nets, J. Softw. Evol. Process 33 (2021) e2329.
[68] C.-P. Bezemer, S. Eismann, V. Ferme, J. Grohmann, R. Heinrich, P. Jamshidi, W. Shang, A. van Hoorn, M. Villavicencio, J. Walter, et al., How is performance addressed in devops? in: Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering, 2019, pp. 45–50.
[69] B. Chen, Improving the software logging practices in devops, in: 2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings, ICSE-Companion, IEEE, 2019, pp. 194–197.
[70] J. Chen, Performance regression detection in devops, in: 2020 IEEE/ACM 42nd International Conference on Software Engineering: Companion Proceedings, ICSE-Companion, IEEE, 2020, pp. 206–209.
[71] J. Díaz, R. Almaraz, J. Pérez, J. Garbajosa, Devops in practice: an exploratory case study, in: Proceedings of the 19th International Conference on Agile Software Development: Companion, 2018, pp. 1–3.
[72] J. Díaz, J.E. Perez, A. Yague, A. Villegas, A. de Antona, Devops in practice–a preliminary analysis of two multinational companies, in: International Conference on Product-Focused Software Process Improvement, Springer, 2019, pp. 323–330.
[73] J. Díaz, D. López-Fernández, J. Pérez, Á. González-Prieto, Why are many businesses instilling a devops culture into their organization? Empir. Softw. Eng. 26 (2021) 1–50.
[74] N. Azad, Understanding devops critical success factors and organizational practices, in: 2022 IEEE/ACM International Workshop on Software-Intensive Business, IWSiB, IEEE, 2022, pp. 83–90.
[75] S. Rafi, M.A. Akbar, A.A. AlSanad, L. AlSuwaidan, H. Abdulaziz AL-ALShaikh, H.S. AlSagri, Decision-making taxonomy of devops success factors using preference ranking organization method of enrichment evaluation, Math. Probl. Eng. (2022) 2022.
[76] I. Karamitsos, S. Albarhami, C. Apostolopoulos, Applying devops practices of continuous automation for machine learning, Information 11 (2020) 363.
[77] M. Hüttermann, C. Rosenkranz, Devops: Walking the shadowy bridge from development success to information systems success, in: European Conference on Information Systems: Human Values Crisis in a Digitalizing World. ECIS 2021 Research Papers, 2019, pp. 1–9.
[78] W.P. Luz, G. Pinto, R. Bonifácio, Building a collaborative culture: a grounded theory of well succeeded devops adoption in practice, in: Proceedings of the 12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, 2018, pp. 1–10.
[79] C. Marnewick, J. Langerman, Devops and organisational performance: the fallacy of chasing maturity, IEEE Softw. (2020).
[80] K. Maroukian, S.R. Gulliver, Leading devops practice and principle adoption, in: 9th International Conference on Information Technology Convergence and Services, 2020.
[81] V. Mohan, L.B. Othmane, Secdevops: Is it a marketing buzzword?-mapping research on security in devops, in: 2016 11th International Conference on Availability, Reliability and Security, ARES, IEEE, 2016, pp. 542–547.
[82] M. Olszewska, M. Waldén, Devops meets formal modelling in high-criticality complex systems, in: Proceedings of the 1st International Workshop on Quality-Aware DevOps, 2015, pp. 7–12.
[83] P. Perera, M. Bandara, I. Perera, Evaluating the impact of devops practice in Sri Lankan software development organizations, in: 2016 Sixteenth International Conference on Advances in ICT for Emerging Regions, ICTer, IEEE, 2016, pp. 281–287.
[84] M. Rowse, J. Cohen, A survey of devops in the South African software context, in: 54th Hawaii International Conference on System Sciences, HICSS 2021, Kauai, Hawaii, USA, January 5, 2021, ScholarSpace, 2021, pp. 1–10, http://hdl.handle.net/10125/71435.
[85] H. Salameh, The impact of devops automation, controls, and visibility practices on software continuous deployment and delivery, in: Proceedings of the 2nd International Conference on Research in Management and Economics, 2019, pp. 22–46.
[86] M. Shahin, M.A. Babar, L. Zhu, The intersection of continuous deployment and architecting process: Practitioners' perspectives, in: Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, ESEM 2016, Ciudad Real, Spain, September (2016) 8-9, ACM, 2016, http://dx.doi.org/10.1145/2961111.2962587, 44:1–44:10.
[87] D. Trihinas, A. Tryfonos, M.D. Dikaiakos, G. Pallis, Devops as a service: Pushing the boundaries of microservice adoption, IEEE Internet Comput. 22 (2018) 65–71.
[88] M. Van Belzen, J. Trienekens, R. Kusters, Critical success factors of continuous practices in a devops context, 2019.
[89] A. Wahaballa, O. Wahballa, M. Abdellatief, H. Xiong, Z. Qin, Toward unified devops model, in: 2015 6th IEEE International Conference on Software Engineering and Service Science, ICSESS, IEEE, 2015, pp. 211–214.
[90] A. Wiedemann, M. Wiesche, H. Gewald, H. Krcmar, Understanding how devops aligns development and operations: a tripartite model of intra-it alignment, Eur. J. Inform. Syst. 29 (2020) 458–473.
[91] D. Šmite, N.B. Moe, J. Gonzalez-Huerta, Overcoming cultural barriers to being agile in distributed teams, Inf. Softw. Technol. 138 (2021) 106612.
[92] A. Trigo, J. Varajão, L. Sousa, Devops adoption: Insights from a large European Telco, Cogent Eng. 9 (2022) 2083474.
[93] J. Bird, DevOpsSec: Delivering Secure Software Through Continuous Delivery, O'Reilly Media, 2016.
[94] P.D. Collins, J. Hage, F.M. Hull, Organizational and technological predictors of change in automaticity, Acad. Manag. J. 31 (1988) 512–543.

[95] L. Lwakatare, Devops Adoption and Implementation in Software Development Practice Concept, Practices, Benefits and Challenges (Ph. D. dissertation), 2017.
[96] C.S. Tsanos, K.G. Zografos, A. Harrison, Developing a conceptual model for examining the supply chain relationships between behavioural antecedents of collaboration, integration and performance, Int. J. Logist. Manag. (2014).
[97] G.L. Kolfschoten, G.-J. de Vreede, R.O. Briggs, H.G. Sol, Collaboration 'engineerability', Group Decis. Negot. 19 (2010) 301–321.
[98] C. Yoon, K. Lee, B. Yoon, O. Toulan, Typology and success factors of collaboration for sustainable growth in the it service industry, Sustainability 9 (2017).
[99] R.d. Feijter, S. Overbeek, R.v. Vliet, E. Jagroep, S. Brinkkemper, Devops competences and maturity for software producing organizations, in: Enterprise, Business-Process and Information Systems Modeling, Springer, 2018, pp. 244–259.
[100] V. Casey, Leveraging or exploiting cultural difference?, in: 2009 Fourth IEEE International Conference on Global Software Engineering, IEEE, 2009, pp. 8–17.
[101] M.M. Lehman, Uncertainty in computer application and its control through the engineering of software, J. Softw. Maint. Res. Pract. 1 (1989) 3–27.
[102] N.B. Moe, V. Stray, M.R. Goplen, Studying onboarding in distributed software teams: a case study and guidelines, in: Proceedings of the Evaluation and Assessment in Software Engineering, 2020, pp. 150–159.
[103] J. Baker, The technology–organization–environment framework, Inform. Syst. Theory (2012) 231–245.
[104] P.S.S. Júnior, M.P. Barcellos, R. de Almeida Falbo, J.P.A. Almeida, From a scrum reference ontology to the integration of applications for data-driven software development, Inf. Softw. Technol. 136 (2021) 106570.
[105] L. Chen, Microservices: architecting for continuous delivery and devops, in: 2018 IEEE International Conference on Software Architecture, ICSA, IEEE, 2018, pp. 39–397.
[106] L.G. Tornatzky, M. Fleischer, A.K. Chakrabarti, Processes of technological innovation, in: Lexington Books, 1990.
[107] M. Rogers, Everett, Diffusion of Innovations. Vol. 12, New York, 1995.
[108] J.Y. Thong, An integrated model of information systems adoption in small businesses, J. Manage. Inf. Syst. 15 (1999) 187–214.
[109] K.K. Kuan, P.Y. Chau, A perception-based model for edi adoption in small businesses using a technology–organization–environment framework, Inform. Manag. 38 (2001) 507–521.
[110] P.Y. Chau, K.Y. Tam, Factors affecting the adoption of open systems: an exploratory study, MIS Q. (1997) 1–24.
[111] Z. Zhang, E. Xia, J. Huang, et al., Impact of the moderating effect of national culture on adoption intention in wearable health care devices: Meta-analysis, JMIR MHealth UHealth 10 (2022) e30960.
[112] A. Tarhini, K. Hone, X. Liu, T. Tarhini, Examining the moderating effect of individual-level cultural values on users' acceptance of e-learning in developing countries: a structural equation modeling of an extended technology acceptance model, Interact. Learn. Environ. 25 (2017) 306–328.
[113] G. Hofstede, Culture's Consequences: Comparing Values, Behaviors, Institutions and Organizations Across Nations, Sage Publications, 2001.
[114] T. Oliveira, M. Thomas, M. Espadanal, Assessing the determinants of cloud computing adoption: An analysis of the manufacturing and services sectors, Inform. Manag. 51 (2014) 497–510.
[115] A. Ahimbisibwe, R.Y. Cavana, U. Daellenbach, A contingency fit model of critical success factors for software development projects: A comparison of agile and traditional plan-based methodologies, J. Enterp. Inform. Manag. 28 (2015) 7–33, http://dx.doi.org/10.1108/JEIM-08-2013-0060.
[116] A. Aldahmash, A.M. Gravell, Y. Howard, A review on the critical success factors of agile software development, in: J. Stolfa, S. Stolfa, R.V. O'Connor, R. Messnarz (Eds.), Systems, Software and Services Process Improvement, Springer International Publishing, Cham, 2017, pp. 504–512.
[117] C. Tam, E.J. da Costa Moura, T. Oliveira, J. Varajão, The factors influencing the success of on-going agile software development projects, Int. J. Project Manag. 38 (2020) 165–176, http://dx.doi.org/10.1016/j.ijproman.2020.02.001.
[118] J. Rahikkala, S. Hyrynsalmi, V. Leppänen, I. Porres, The role of organisational phenomena in software cost estimation: An empirical study of supporting and hindering factors, E-Informatica Softw. Eng. J. 12 (2018) 167–198, http://dx.doi.org/10.5277/e-Inf180101.
[119] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson, Systematic mapping studies in software engineering, in: 12th International Conference on Evaluation and Assessment in Software Engineering, Vol. 12, EASE, 2008, pp. 1–10.
[120] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, A. Wesslén, Planning, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 89–116, http://dx.doi.org/10.1007/978-3-642-29044-2_8.
[121] M. Hüttermann, DevOps for Developers, A Press, 2012.