Hindawi
Mathematical Problems in Engineering
Volume 2022, Article ID 2600160, 15 pages
https://doi.org/10.1155/2022/2600160

Research Article

Decision-Making Taxonomy of DevOps Success Factors Using Preference Ranking Organization Method of Enrichment Evaluation

Saima Rafi,[1] **Muhammad Azeem Akbar**,[2] **Abeer Abdulaziz AlSanad**,[3] **Lulwah AlSuwaidan**,[3] **Halah Abdulaziz AL-ALShaikh**,[3] **and Hatoon S AlSagri**[3]

[1]*University of Murcia, Department of Informatics and Systems, Murcia 30100, Spain*
[2]*Lappeenranta University of Technology, Department of Information Technology, Lappeenranta 53851, Finland*
[3]*Imam Mohammad Ibn Saud Islamic University, Information Systems Department, Riyadh 11432, Saudi Arabia*

Correspondence should be addressed to Saima Rafi; saeem112@gmail.com, Muhammad Azeem Akbar; azeem.akbar@ymail.com, and Abeer Abdulaziz AlSanad; aaasanad@imamu.edu.sa

Received 30 October 2021; Accepted 17 December 2021; Published 10 January 2022

Academic Editor: Naeem Jan

Due to multitudes factors like rapid change in technology, customer needs, and business trends, the software organizations are facing pressure to deliver quality software on time. To address this concern, the software industry is continually looking the solution to improve processing timeline. Thus, the Development and Operations (DevOps) has gained a wide popularity in recent era, and several organizations are adopting it, to leverage its perceived benefits. However, companies are facing several problems while executing the DevOps practices. The objective of this work is to identify the DevOps success factors that will help in DevOps process improvement. To accomplish this research firstly, a systematic literature review is conducted to identify the factors having positive influence on DevOps. Secondly, success factors were mapped with DevOps principles, i.e., culture, automation, measurement, and sharing. Thirdly, the identified success factors and their mapping were further verified with industry experts via questionnaire survey. In the last step, the PROMETHEE-II method has been adopted to prioritize and investigate logical relationship of success factors concerning their criticality for DevOps process. This study's outcomes portray the taxonomy of the success factors, which help the experts design the new strategies that are effective for DevOps process improvement.

1. Introduction

The multitude nature of software organizations like rapid change in technology [1], increasing business competence, and market demands [2] has changed the overall environment of software industry. Many organizations are facing common problems: continuous deployment and delivery of software. The customers nowadays expect fast delivery and frequent response on their product. To fulfil this gap, frequent releases are important for customers to provide feedback on continuous bases [3]. Agile, in software development, is the common solution adopted by several companies to reduce development time and deployment cost and increase customer satisfaction and success rate of software project. However, the focus of agile paradigm is on collaboration between the development teams, ignoring the operations teams. The tasks performed by operations teams such as deployment, management, customer performance, and support systems site [4] will lag behind if gaps between operation and development teams are not merged.

To improve the communication, coordination, and integration paths between development and operations teams, a paradigm known as DevOps has been adopted [5]. The word DevOps originated from two words developers "Dev" and operations "Ops" [1]. DevOps paradigm is adopted in various organizations to extend the collaboration between developers and operations in personal. DevOps is defined as "a cluster of practices intended for smooth interactions

between developers and operations teams to work as a single unit" [4]. Overall, the activities included in DevOps paradigm are an environment usually adopted for continuous development and delivery process in software organizations to increase service quality [1]. Several companies like Facebook, Amazon, Flickr, and Netflix are working in DevOps paradigm to increase performance cycle [4]. However, DevOps is still a challenging process to be adopted. The reason behind this concern is that there are a heap of information, definitions, frameworks, practices, and courses related to DevOps process improvement, lacking guidance to structure and organize such massive information, for proper DevOps adoption. To do so, they need proper structured software development paradigm.

The literature shows that numerous researches have been conducted on the topic considering the tools [6], practices [7], attributes, and the factors influencing DevOps implementation [8] in software organizations. Several of the available studies in literature have adopted empirical approach to collect data regarding DevOps definition, tools, continuous delivery pipeline, practices, and influencing factors. However, fewer studies present logical relationship between factors influencing DevOps paradigm. Thus, the main research concern is that we are known with factors, practices, and tools of DevOps, but there is no detailed holistic view that shows the logical relationship and prioritization between factors influencing DevOps implementation. This motivated us to extend our preliminary findings published in Ease-2020 [9], in which we explored the list of success factors (SFs) reported by the researchers. We are extending our study [9] by conducting an empirical study with real practitioners (in industry) and by logically examining the relationship between the success factors and the DevOps principles, i.e., "culture, automation, measurement, and sharing (CAMS)" [10]. The results of this study come up with a taxonomy that presents logical relationship and ranking; and it will assist in enhancing the decision-making capability of DevOps experts while considering particular success factor (to their mapped criterion) for successful DevOps implementation. This categorization of success factors will also help DevOps experts focus on the critical areas of DevOps paradigm, which needs further improvement.

The objectives of this study are (1) to explore and identify the success factors from state of the art, (2) to empirically verify the identified success factors with industry experts, (3) to map the success factors with CAMS criteria (principles of DevOps), and (4) to develop a taxonomy based on ranking of the success factors, based on logical relationship and CAMS criteria. To address these objectives, we have designed research questions (RQs).

(RQ1) What are the success factors of DevOps reported in state of the art and state of the practices?

(RQ2) How to enhance decision-making capabilities of DevOps experts for considering the particular success factor?

The structure of this paper after introduction consists of background and motivation in section 2. Section 3 is about methodology. Results, discussion and summary, and limitations are explained in Sections 4,5 and 6. Study implications and conclusion and future work are briefly explained in the last two Sections 7 and 8 of this paper.

2. Background and Motivation

The DevOps is a vague term and is defined in various context in literature. Peuraniemi [11] defined DevOps as an environment in which developers and operations teams work in a group form. Dyck et al. [12] state DevOps as an approach that stresses empathy and cross-functional collaboration between and within the teams in software organizations, in order to operate resilient systems and accelerate delivery of changes. Erich et al. [13] have explained DevOps in terms of "conceptual framework that supports team of developers and operations in software development organizations". According to Virmani [2], DevOps covers all the aspects that aid in the delivery of fast, optimized, and high quality software. There are four main criteria of DevOps that help in better adoption of and implementation of DevOps, i.e., culture, automation, measurement, and sharing (CAMS). The focus of DevOps is to build bridges for smooth interaction between developers and operations teams [10]. There are some other studies that highlighted DevOps significance in various aspects. For example, Macarthy and Bass [14] worked on building taxonomy of DevOps in practices, Gokarna and Singh [15] empirically investigated the historical background of DevOps, Mishra and Otaiwi [16] discussed in their research DevOps software quality to maintain continuous environment, and Rafi et al. [17] discussed the data quality aspects of software application while adopting DevOps.

Lwakatare et al. [4] have identified DevOps from monitoring perspective. They claimed that, to monitor system logs and certain tools performance, DevOps paradigm will help two teams collaborate in such a way that all details will be monitored in a continuous manner. Peuraniemi [11] discovered "Infrastructure as Code (IaC) as a main component of DevOps as it helps in automatically managing and configurations during software development process." Farroha and Farroha [1] stated that the DevOps main agenda is to provide software product with high quality, agility technology to make pressing smooth, human-based factors, bridging the loop holes through coordination between developers and operations team. Semds et al. [18] performed literature review to state DevOps in terms of enablers and capabilities. In their study, interviews were conducted that analyze the organizational impediments barriers in DevOps implementation. Rafi et al. [8] proposed a readiness-model for the effective adoption of DevOps in software development organization. The readiness-model proposed was based on facts collected from state of the art and state of the practices. Similarly, Leite et al. [19] presented key concerns of DevOps in form of conceptual map, from the viewpoint of team managers, researchers, and engineers. We have adopted the same approach as used by Leite et al. [19] to identify DevOps success factors (SFs) for effective implementation of DevOps process. Furthermore, we have prioritized the success factors and developed a taxonomy

that will enhance decision-making capability of DevOps experts while selecting particular success factor.

Despite the demand of DevOps in software organizations, there are various studies on tools, definition, concepts, factors, practices, and characterization of DevOps, but none of them discussed conceptual mapping and relationship between success factors in detail. This research gap motivated us to highlight the logical relationship between success factors and develop a taxonomy that will help DevOps experts in designing strategies for better implementation of DevOps. To conduct the research, we have extended our previous study on identification of success factors for DevOps implementation published in EASE′20 conference [9]. We have mapped the identified SFs into CAMS criteria (principles of DevOps) [10] to develop a taxonomy. Moreover, we have also analyzed the logical relationship between success factors and ranked them by introducing PROMETHEE-II method [20] in domain of DevOps. This ranking and logical relationship will help DevOps experts enhance their decision-making capabilities of DevOps for considering particular success factor.

There are various approaches for measuring ranks in various domains of technology. For example, "TOPSIS, ELECTRE, WSM, AHP, PROMETHEE, etc." [21] are also approaches to measure ranks. However, PROMETHEE is an approach that measures rank by pairwise comparison of alternatives, finding the strength of one alternative to another alternative [22]. This approach has been used in different fields of science by researchers, where decision making is required [21]. The results generated by this approach are easy to understand and consistent. For example, Siahaan and Mersan [22] used PROMETHEE approach to select best student in college based on criteria, i.e., skills, performance in class, grading, attendance, etc. To evaluate the quality of railway services provided to passenger, Liu and Guan [23] applied PROMETHEE technique. Using fuzzy triangular scale, the linguistic terms were transformed to numbers, and based on evaluation, service quality of railway passengers was ranked. Zhao et al. [24] have upgraded PROMETHEE approach for the assessment of incident management plans, i.e., earthquake, floods, etc.

3. Research Methodology

To fulfill the study objectives, the following research map is adopted (Figure 1). Firstly, systematic literature review (SLR) was adopted to identify the SFs of DevOps implementation in software organization. Secondly, we performed mapping of identified success factors with CAMS criteria (principles of DevOps) [10]. Thirdly, to validate the mapping scheme and factors relationship with CAMS criteria, the questionnaire survey was conducted. In last step, we have used PROMETHEE-II method [20] to rank the SFs concerning their significance for DevOps implementation.

3.1. Phase 1. (RQ: state of the art) We have applied systematic literature review technique in our previous study [9] using scenario of development of research questions, search strategy, study selection, and data synthesis to identify the factors having positive influence on DevOps implementation. This step-by-step approach is useful for robust outcomes compared to other informal techniques of data extraction from literature. Kitchenham and Charters [25] guidelines were followed, which include the following steps: "(i) planning the review, (ii) conducting the review, and (iii) reporting the review." A total of 69 studies were shortlisted after applying QA process, and 16 SFs were identified. The list of the identified success factors is shown in Section 4. We have briefly explained all factors in our previous study [9].

3.2. Phase 2: (RQ2)

3.2.1. Mapping of Success Factors. Humble and Molesky [10] explained four basic categories, i.e., DevOps principle (culture, automation, measurement, and sharing (CAMS)). We classified the identified success factors into these categories. Rafi et al. [17] adopted the same technique in their research to categorize the identified factors into 4 basic principles of DevOps. By following the same concept, the SFs (identified) were categorized into CAMS criteria. The details about CAMS criteria are explained below:

3.2.2. Culture. According to Humble and Molesky [10], culture is defined as role of people over tools and processes. To maintain culture in DevOps, the operations must collaborate with development team. The organization should schedule meetings in which both operations and development teams are present. They should share their ideas and point of view with each other to maintain continuous flow.

3.2.3. Automation. To obtain frequent feedbacks and software delivery on time, automation of built, testing, and deployment process are essential. To perform this task, the teams need deployment pipeline to be automated. If development team made any change in the software, this change will be handed over to operations team for implementation after performing series of testing in the pipeline. After the approval of new change, the team will be ready to deploy the software product towards production units.

3.2.4. Measurement. In measurement, monitoring of high- and low-grade business metrics is performed. The elements involved in high-level metrics are revenue, time, transactions, cost, etc. In low-level metrics, the people are observed. To improve performance, people skills, behavior, and capabilities cannot be measured; therefore, the right decision must be made while selecting the project team. The key metrics (high and low) must be visible to the team involved in development process, so that their performance is measured.

3.2.5. Sharing. The sharing principle of DevOps includes sharing of knowledge, resources, tools, and techniques, management of infrastructure, and environments for

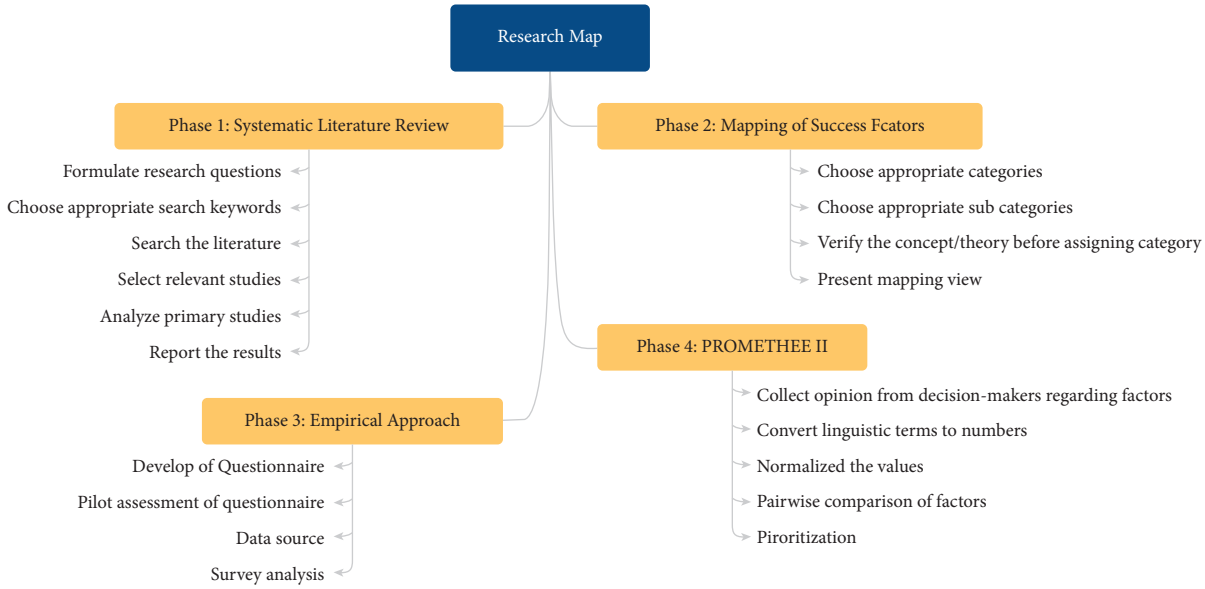

Figure 1: Research design.

development and operations teams. To release a valuable software product, two teams must collaborate with each other in all phases of development until the release of software.

To perform mapping, we have followed the guidelines and coding scheme of Grounded Theory (GT) approach [26]. The SFs are grouped to CAMS category. The 4 key steps were performed for mapping process that includes code, subcategories, categories, and theory/framework. Based on the understanding of SFs, all the authors of this research mapped the factors into related category of DevOps principles (CAMS). The mapping scheme is shown in Figure 2.

To remove personal biases during mapping, we have performed interrater reliability test [27]. The three external experts having knowledge about DevOps and empirical software engineering were invited to perform this task. The experts were from Nanjing University, China, and Software organization in UK. The experts followed all the steps of mapping again and grouped the identified SFs according to their knowledge. The responses collected from study authors and external experts were further evaluated by calculating the nonparametric Kendall's coefficient of concordance (W) [27]. The value of W near to 1 shows agreement, and the value of W near to 0 represents disagreement on the grouping of SFs (performed by both authors of this research and external experts). The outcomes ($W = 0.91$) show the significant agreement between the teams.

3.3. Phase 3: (RQ1: State of Practices &RQ2). The objective of questionnaire survey is to validate the mapping scheme and to determine the logical relationship of SFs with DevOps principles (CAMS). The steps adopted to perform questionnaire survey are presented in Figure 1. This technique will help the researchers gather information from real industry of DevOps environment. Akbar et al. [28] followed the same approach to collect expert's opinion from large population. They developed the questionnaire to determine the existence of identified factors in real industry. Akbar et al. used this approach in two other studies while investigating success factors of requirement change management [29] and while investigating challenges of requirement change management [30]. Khan et.al [31] also used questionnaire survey approach to investigate the barriers of software process improvement in software organizations.

3.3.1. Questionnaire Development. The questionnaire was designed by using the platform of Google form. The sample questionnaire consists of four sections: (1) biographical information, (2) the closed-ended questions that consist of DevOps SFs identified from literature along with mapping categories, (3) to analyze the relationship of DevOps SFs with CAMS criteria, and 4) this section consists of open ended questions to gather additional information to practitioners related to DevOps implementation. The Likert scale was used to collect responses from practitioners. The Likert scale consists of responses, i.e., positive = Extremely Agree (EA) and Agree (A), negative = Extremely Disagree (ED) and Disagree (D), and Neutral (N). The positive response indicates the number of participants who agreed on mapping scheme of DevOps SFs. However, the negative response shows the percentage of participants who did not consider the mapping scheme effective for DevOps implementation. The neutral category indicates the participants' feedback who have a neutral opinion about the DevOps SFs.

3.3.2. Pilot Assessment of Questionnaire. The developed questionnaire was initially assessed concerning to the understand-ability and scalability of designed questions and the structure of the questionnaire. To perform pilot assessment, we emailed our questionnaire sample to three

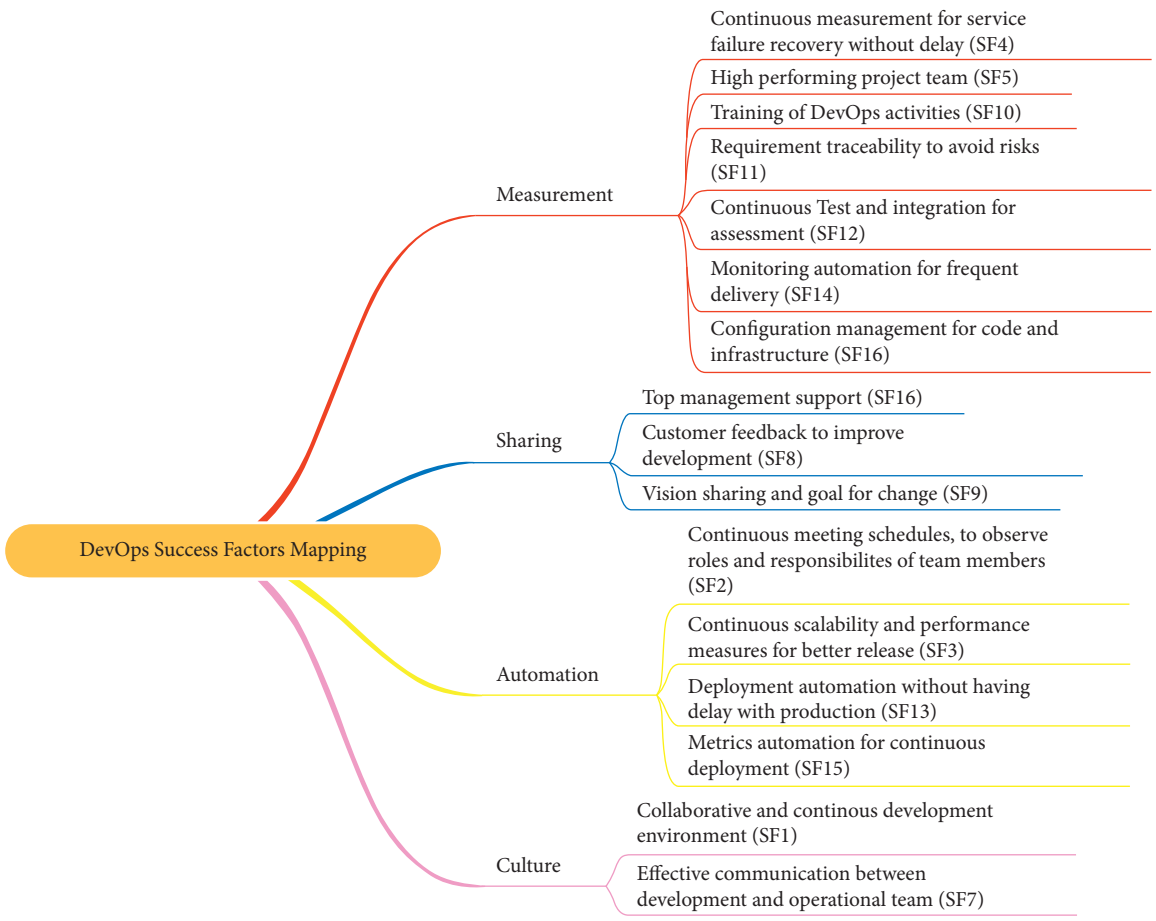

Figure 2: Mapping of SFs with DevOps principles (CAMS).

external experts (two from Soft. Tech Spain and one from King Fahd University, Saudi Arabia). The requested experts suggested some modifications related to questionnaire design, e.g., to add Likert scale before section 2 and to add comments point in section 4 (to gather more open view about successful implementation of DevOps from participants). We redesigned our questionnaire according to the recommendation of experts. The final version of questionnaire after approval of experts is presented in Appendix B.

3.3.3. Data Source. The objective of this study to validate the mapping scheme of DevOps SFs with CAMS criteria [13] and analyze the weightage of each factor with respect to CAMS criteria. The data source is essential to gather responses from targeted population. Therefore, for validation of the mapping scheme we have used is Research-Gate, Emails, and LinkedIn profiles. To spread questionnaire, we have used snowball approach [32], which is a cost effective method to gather responses from targeted population [33]. During data collection phase (i.e., January 2021 to February 2021), we have collected total 100 complete responses. The details about participants responses are presented in Section 4.

3.3.4. Survey Data Analysis. The frequency distribution approach will help in analyzing qualitative and quantitative data. According to [34], "this is an effective technique that assists in measuring the ordinal and nominal data within group (variables) and across the group." This technique is statistically used to compare the SFs and their importance for DevOps implementation process. Niazi et al. [35] used the survey approach to determine the importance of project management in global software development. Akbar et al. [36] used the survey approach to investigate the heavy weight and light weight methodologies on the point star model. Sloane et al. [37] applied the case study and survey approach for the assessment of decision support system in clinical engineering.

3.4. Phase 4: (RQ2). The PROMETHEE "Preference Ranking Organization Method of Enrichment Evaluation" approach was first presented by Brans et al. [20] in 1985. There are several versions of PROMETHEE available, for example, PROMETHEE I, II, III, IV, V, cluster, etc. The nature of a research problem decides the application of the approach. Considering advantages of PROMETHEE technique, it is easy to apply with low complexity rate. This approach has treated many successful applications in various fields for decision analysis. For example, few methods related to PROMETHEE-II are used in emergency management area, where we have to deal with multicriteria decision-making

(MCDM) problems, as it is difficult to manage timeliness requirements in emergency management for so many steps of traditional algorithm and the large number of calculation and comparisons [20]. This method is also used in the field of chemistry [38] and health care [39], where a detailed analysis on logical relationship between elements is required. The logistic management and information technology are also using PROMETHEE-II approach, because "it is interactive and is able to classify and order alternatives, which are complex and difficult to compare" [22]. The other advantages of this method are simplicity, stability, and clarity [40]. The PROMETHEE-II principles are based on pairwise comparison of alternatives with each criterion. Behzadian et al. [41] stated that, in PROMETHEE-II, we evaluated the alternatives based on different criteria (which have to be maximized or minimized). During the comparative study, each criterion should be able to distinguish the alternatives, regardless of how the alternatives behave under other criteria. PROMETHEE is useful in providing complete detail of ranking alternatives, but like other MCDM approaches, "decision makers have to assign weights to each criteria and should have knowhow about outranking relationship between different alternatives and fuzzy variable term of scale" [42].

The seven steps (Figure 3) required performing PROMETHEE-II approach are discussed as follows:

Step 1. The decision matrix is normalized using beneficial and nonbeneficial criteria:

$$D_{\mathrm{ij}} = \frac{\left[M_{\mathrm{ij}} - \min M_{\mathrm{ij}}\right]}{\left[\max\left(M_{\mathrm{ij}}\right) - \min\left(M_{\mathrm{ij}}\right)\right]} \text{ (Beneficial Criteria)}, \quad (1)$$

where $i = 1, 2, 3, \ldots, \mathrm{n}$ and $j = 1, 2, 3, \ldots, \mathrm{m}$

$$D_{\mathrm{ij}} = \frac{\left[\max M_{\mathrm{ij}} - M_{\mathrm{ij}}\right]}{\left[\max\left(M_{\mathrm{ij}}\right) - \min\left(M_{\mathrm{ij}}\right)\right]} \text{ (Non Beneficial Criteria)}, \quad (2)$$

where M_{ij} is a decision-maker's evaluation of ith alternative with respect to jth criterion.

Step 2. "The difference of ith alternative as compared with other ones are determined in this step" [42]. This means that pairwise comparison of each alternative with criteria should be assessed.

Step 3. "In this step, we choose and calculate the preference function $\mathrm{P_j}$ (i, i′)" [42].

According to Brans et al. [43], there are 6 various types of preference functions (ranging from 0 to 1). Figure 4 illustrates these preference functions. The PROMETHEE approach induces a preference function to describe the decision-maker's preference difference between pairs of alternatives on each criterion. It is possible to choose a different function for each criterion. For this study, we have chosen type-1 preference function. This function requires no parameters and thresholds. The type-1 preference function is

$$P_j\left(i, i'\right) = 0 \text{ if } \quad \mathrm{Dij} \le \mathrm{Di}' j, \quad (3)$$

$$P_j\left(i, i'\right) = 1 \quad \mathrm{Dij} \le \mathrm{Di}' j. \quad (4)$$

Step 4. In this step, we will calculate aggregate preference function by combining weights.

$$\pi\left(i, i'\right) = \frac{\sum_{j=1}^{m} P_{\mathrm{ij}}\left(i, i'\right) w_j}{\sum_{j=1}^{m} w_j}, \quad (5)$$

where w_j is the weight of relative importance given to ($\mathrm{j^{th}}$) criterion.

Step 5. "We determine the outranking flow (negative and positive) for each alternative compared with (p-1) alternatives" [43] as shown in Figure 4, and the formulas used to calculated leaving and outranking flow are

$$\text{the entering flow} : \varphi^{+}(i) = \frac{1}{n-1} \sum_{i'-1}^{N} \pi\left(i', i\right), \quad \left(i \ne i'\right), \quad (6)$$

$$\text{the leaving flow:} \ \varphi^{-}(i) = \frac{1}{n-1} \sum_{i'-1}^{N} \pi\left(i, i'\right), \ \left(i \ne i'\right), \quad (7)$$

where N represents the number of alternatives. "The entering flow measures the weakness of alternatives and leaving flow measures the strength of alternatives" [20].

Step 6. In this step, we calculate the net outranking flow of all alternatives.

$$\text{The net flow outranking} : \varphi(i) = \varphi^{+}(i) - \varphi^{-}(i). \quad (8)$$

Step 7. "To calculate the ranking of alternatives consider values of net outranking flow $\varphi(i)$. The highest the $\varphi(i)$ value, the best the alternative" [43]. The PROMETHEE-II also avoids incomparability between alternatives.

4. Results

4.1. Identification of Success Factors (RQ1). A total of 69 studies were selected after applying all steps of tollgate approach during systematic literature review, and sixteen success factors were identified. The list of the SFs along with percentage wise impact of SFs in DevOps paradigm is presented in Table 1 and Appendix A shows quality evaluation of selected studies in detail.

According to Table 1, the factors "Effective communication between development and operational team" SF7, "Top management support" SF6, and "Customer feedback to improve development" SF8 are the most important factors, which show positive impact on DevOps paradigm. Considering the significance of these factors would help in effective DevOps implementation in software industry. All identified factors were briefly discussed in our previous study [9].

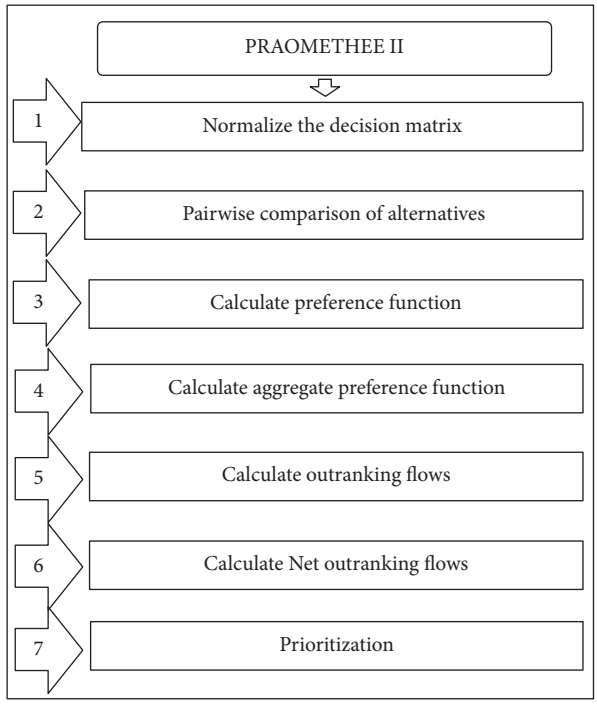

Figure 3: PROMETHEE-II steps.

4.2. Mapping of DevOps Success Factors (RQ2). Further, we performed mapping of identified SFs into 4 categories of DevOps principles, i.e., culture, automation, measurement, and sharing [10]. Research team and three external researchers, as explained in Section 3, perform the mapping. The aim of this criteria wise subdivision is to perform PROMETHEE-II analysis for logical relationship and ranking of SFs based on their importance with DevOps process improvement. To verify the findings of mapping, we have applied interrater reliability analysis and questionnaire survey. The demographic details of survey participants are listed in Appendix C. Figure 2 shows the final version of DevOps success factors mapping. The majority of survey participants agreed on the SFs mapping with DevOps principles (CAMS) Table 2. The frequency analysis results in Table 2 show that most of the success factors and their mapped category are >55%. Furthermore, results show that "culture" and "automation" with 91% are the highest ranked category according to survey respondents and SF8 "customer feedback to improve development" with 90% being the high ranked SF for DevOps implementation in software organization.

4.3. Results of PROMETHEE-II Approach (RQ2). To analyze the logical relationship and ranks of the identified SFs of DevOps, we have adopted PROMETHEE-II approach [20]. This approach enhances decision-making capabilities of DevOps experts for making decision to consider particular success factor (based on CAMS criteria) for DevOps implementation. To perform this task, the questionnaire survey was conducted as mentioned in above section. The linguistic values assigned by participants of survey (as weightage) were converted into numbers. Table 3 presents the weighted categories with pairwise percentage of SFs for DevOps. The percentage of each SF is determined by using a Likert scale explained in Section 3.

The steps performed for PROMETHEE-II analysis are as follows.

Step 1:To determine normalized decision, matrices (1) and (2) were used. Table 4 shows the constructed normalized decision matric. The criterion "automation" is selected as a nonbeneficial criterion and automation, measurement, and sharing as a beneficial criterion. This selection of cost and beneficial criteria was done depending upon nature of criteria selected for analysis. In this study, "automation" having attributes, i.e., cost, time, effort, etc., must be reduced while implementing DevOps in software organization.

Step 2: To determine the difference in categories and success factors, pairwise difference is calculated. For example, the pairwise difference of SF1 with respect to DevOps principles is calculated in Table 5. The same method has been adopted to calculate the difference of other SFs (Appendix D).

Step 3: To calculate the preference function, "type-1 function" is selected, as no parameters are required. This selection of preference function was made depending upon the nature of problem. We replaced the calculated pairwise difference values considering (3) and (4). For example, the preference function values of SF1 are presented in Table 6.

Step 4: Using (5) aggregate preference function is calculated. The results of SF1 are shown in Table 7. The results of all the other SFs are presented in Appendix D.

Step 5: We determined the positive outranking flow and negative outranking flow for each success factor compared with $(p-1)$ factors (SFs) using (6) and (7). Table 8 presents all values of positive and negative outranking flow.

Step 6: (8) is used to calculate net outranking flow as shown in Table 9.

Step 7: We ranked the SFs considering values of net outranking flow $\varphi(i)$. The highest the $\varphi(i)$ value, the best the success factor (Table 9). The graphical presentation of ranking is shown in Figure 5.

According to the ranking of SFs in Table 9 and Figure 5, "customer feedback to improve development" SF8 is the most critical SF. Therefore, while dealing with continuous development, practitioners should collect customer feedback continuously to deliver software product on time. The versions of software were updated according to customer's feedback without effecting the other tasks [4]. "Configuration management for code and infrastructure" SF16 ranked second most important SF in terms of DevOps implementation. The percentage results of SF16 calculated from SLR 17% show that this factor is not discussed in detail in literature and needs further research. However, the real industry practitioners and research authors of this study

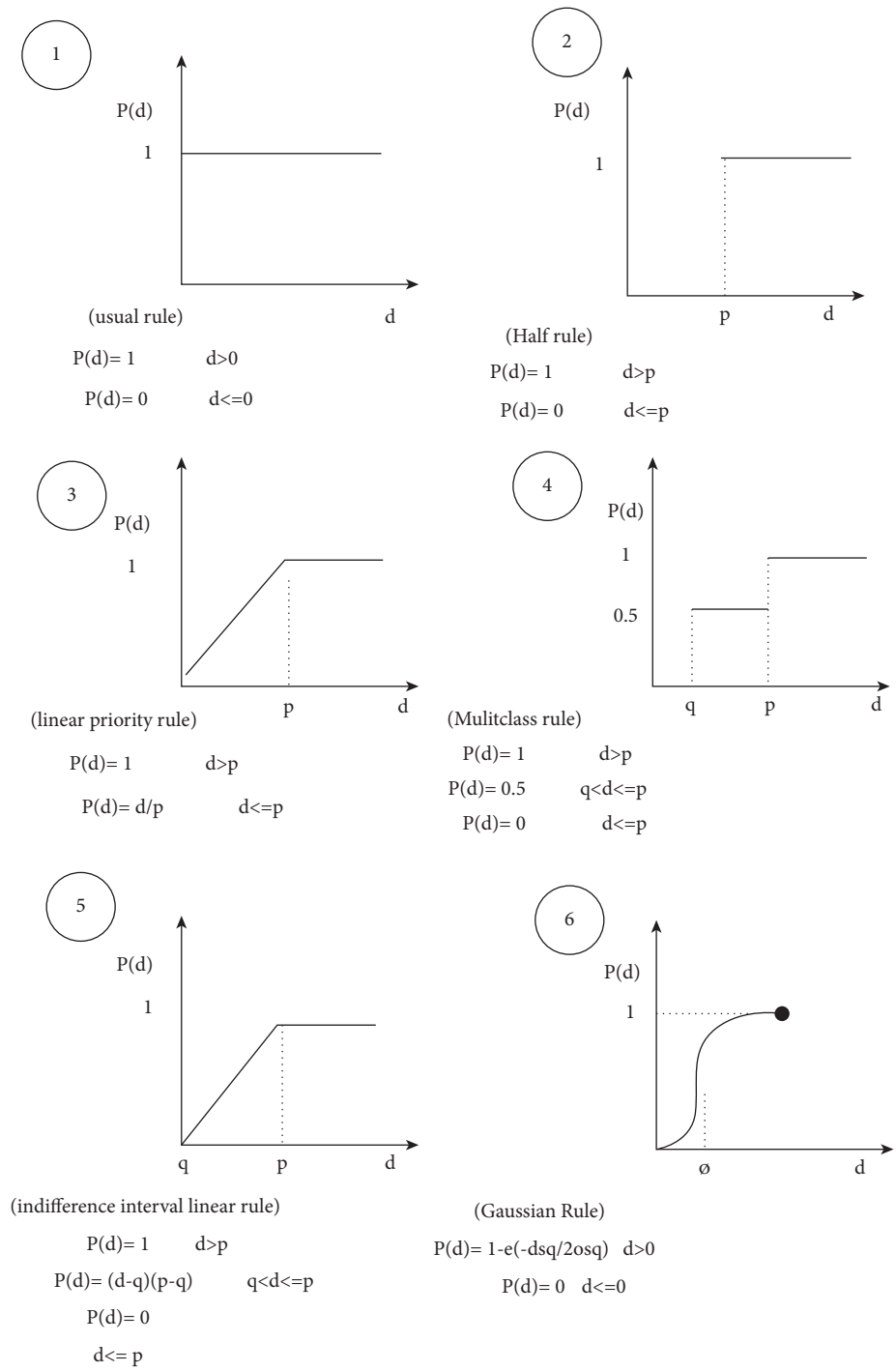

Figure 4: Preference functions.

consider this factor as an important factor to make positive impact on DevOps implementation. "Effective communication between development and operational team" SF7 is the third most important SF for DevOps implementation. As communication is the backbone of DevOps paradigm [11], without effective communication between teams, the DevOps implementation is not possible. "Continuous Test and integration for assessment" SF12 is the fourth most significant SFs for DevOps implementation. The DevOps paradigm requires continuous testing [44] and integration [45] to assess the performance of software product during development phase. This integration for assessment will resolve several issues related to testing and security [46]. The other high ranked SFs are "Continuous meeting schedules, to observe roles and responsibilities of team members" SF2, "Continuous scalability and performance measures for better release" SF3, and "Collaborative and continuous development environment" SF1 playing important role in

Table 1: List of identified SFs [9].

Identified DevOps success factors	Percentage (%)
"Collaborative and continuous development environment (SF1)"	51
"Continuous meeting schedules, to observe roles and responsibilities of team members (SF2)"	70
"Continuous scalability and performance measures for better release (SF3)"	68
"Continuous measurement for service failure recovery without delay (SF4)"	33
"High performing project team (SF5)"	58
"Top management support (SF6)"	75
"Effective communication between development and operational team (SF7)"	90
"Customer feedback to improve development (SF8)"	87
"Vision sharing and goal for change (SF9)"	26
"Training of DevOps activities (SF10)"	85
"Requirement traceability to avoid risks (SF11)"	43
"Continuous test and integration for assessment (SF12)"	55
"Deployment automation without having delay with production (SF13)"	36
"Monitoring automation for frequent delivery (SF14)"	42
"Metrics automation for continuous deployment (SF15)"	49
"Configuration management for code and infrastructure (SF16)"	17

Table 2: Questionnaire response of survey participants.

S.#	List of challenges	Number of responses ($N = 100$)							
		Positive			Negative			Neutral	
		E.A	A	%	D	E.D	%	N	%
C	Culture	41	50	91	0	0	0	10	10
1	"Collaborative and continuous development environment (SF1)"	30	40	70	5	5	10	20	20
2	"Effective communication between development and operational team (SF7)"	40	40	80	0	5	5	15	15
A	Automation	50	41	91	0	1	0	9	9
3	"Continuous meeting schedules, to observe roles and responsibilities of team members (SF2)"	30	38	68	0	3	3	29	29
4	"Continuous scalability and performance measures for better release (SF3)"	20	40	60	3	5	8	32	32
5	"Deployment automation without having delay with production (SF13)"	40	40	80	0	5	5	15	15
6	"Metrics automation for continuous deployment (SF15)"	31	34	65	5	5	10	25	25
M	Measurement	40	32	72	2	2	4	24	24
7	"Continuous measurement for service failure recovery without delay (SF4)"	20	40	60	10	1	11	29	29
8	"High performing project team (SF5)"	28	32	60	5	10	15	25	25
9	"Training of DevOps activities (SF10)"	30	40	70	10	0	10	20	20
10	"Requirement traceability to avoid risks (SF11)"	30	41	71	5	6	11	28	28
11	"Continuous test and integration for assessment (SF12)"	30	40	70	10	0	10	20	20
12	"Monitoring automation for frequent delivery (SF14)"	35	35	70	0	10	10	20	20
13	"Configuration management for code and infrastructure (SF16)"	31	34	65	5	5	10	25	25
S	Sharing	30	40	70	6	5	11	19	19
14	"Top management support (SF6)"	30	26	56	4	10	14	30	30
15	"Customer feedback to improve development (SF8)"	40	50	90	1	0	1	9	9
16	"Proper vision and goal for change (SF9)"	39	36	75	5	0	5	20	20

terms of DevOps implementation and better performance measure in software organization.

4.4. Ranking Based Taxonomy of DevOps Success Factors. Furthermore, we have determined the ranking of SFs according to CAMS criteria to develop a holistic taxonomy. The DevOps principles, i.e., CAMS, are briefly explained in Section 3. We have used the formal approach for mapping of SFs with CAMS criteria (section 3), applied in other research as well. The developed taxonomy of SFs and DevOps principles will enhance decision-making capabilities of DevOps experts to consider particular success factor for DevOps implementation. Figure 6 shows the overall ranking "OR" and criterion wise ranking "CR" of success factors, respectively. According to the ranking, "Effective communication between development and operational team" SF7 is ranked as third in OR of pairwise comparison but ranks first in CR in "culture" criterion of DevOps principles, similarly, "Vision sharing and goal for change" SF 9, OR = 8 and CR = 2, which indicates that SF9 has significant value within the mapped criterion, i.e., "sharing." This ranking of SFs will assist the practitioners in improving their decision-making approach for the consideration of particular success factor within the criterion and as a whole.

5. Discussion and Summary

The objective of this research is to identify SFs that can affect DevOps implementation in positive manner. To address this research objective, firstly, we have conducted a SLR study to explore DevOps SFs in available literature. Secondly, the mapping of success factors into CAMS criteria [10] was performed with study authors and three external researchers (DevOps experts). Thirdly, to verify the mapping scheme of DevOps SFs, we have conducted a questionnaire survey.

Table 3: Percentage of SFs for DevOps with weighted categories.

Principles	Culture	Automation	Measurement	Sharing
Weights	0.35	0.35	0.25	0.15
Percentage of success factors				
SF1	70	63	73	73
SF2	80	73	70	81
SF3	81	75	69	83
SF4	67	79	62	60
SF5	76	70	60	60
SF6	69	50	61	52
SF7	89	91	90	70
SF8	100	83	71	81
SF9	79	56	67	70
SF10	91	80	63	78
SF11	54	76	69	59
SF12	97	71	64	64
SF13	59	83	60	70
SF14	54	80	71	65
SF15	76	71	53	60
SF16	87	71	91	54
Max value	100	91	91	83
Min value	54	50	53	52

Table 4: Normalized decision-matrix.

Principles	Culture	Automation	Measurement	Sharing
Weighted values	0.35	0.35	0.25	0.15
Normalized decision-matrix				
SF1	0.35	0.68	0.53	0.68
SF2	0.57	0.44	0.45	0.94
SF3	0.59	0.39	0.42	1.00
SF4	0.28	0.29	0.24	0.26
SF5	0.48	0.51	0.18	0.26
SF6	0.33	1.00	0.21	0.00
SF7	0.76	0.00	0.97	0.58
SF8	1.00	0.20	0.47	0.94
SF9	0.54	0.85	0.37	0.58
SF10	0.80	0.27	0.26	0.84
SF11	0.00	0.37	0.42	0.23
SF12	0.93	0.49	0.29	0.39
SF13	0.11	0.20	0.18	0.58
SF14	0.00	0.27	0.47	0.42
SF15	0.48	0.49	0.00	0.26
SF16	0.72	0.49	1.00	0.06

Finally, the PROMETHEE-II method [20] was applied to analyze the logical relationship and rank each SF with respect to their significance for DevOps implementation.

(RQ1) What are the success factors of DevOps reported in state of the art and state of the practices?

To answer this research question, we have conducted a SLR study to explore DevOps SFs available in literature. The 16 success factors were identified presenting the key areas where the DevOps experts must focus on for better implementation of DevOps. The SFs were further mapped into principles of DevOps (CAMS). The identified DevOps SFs and their mapping scheme with CAMS are presented in Section 4. The SFs and their mapping were empirically investigated by questionnaire approach. A total 100 responses were collected, and the results show that the identified SFs have positive influence on DevOps implementation. The demographic details of participants are given in Appendix C.

Table 5: Pairwise difference of identified SF1.

Principles	Culture	Automation	Measurement	Sharing
Pairwise difference of SF1				
D(SF1-SF2)	−0.22	0.24	0.08	−0.26
D(SF1-SF3)	−0.24	0.29	0.11	−0.32
D(SF1-SF4)	0.07	0.39	0.29	0.42
D(SF1-SF5)	−0.13	0.17	0.35	0.42
D(SF1-SF6)	0.02	−0.32	0.32	0.68
D(SF1-SF7)	−0.41	0.68	−0.44	0.10
D(SF1-SF8)	−0.65	0.48	0.06	−0.26
D(SF1-SF9)	−0.19	−0.17	0.16	0.10
D(SF1-SF10)	−0.45	0.41	0.27	−0.16
D(SF1-SF11)	0.35	0.31	0.11	0.45
D(SF1-SF12)	−0.58	0.19	0.24	0.29
D(SF1-SF13)	−0.24	0.48	0.35	0.10
D(SF1-SF14)	0.35	0.41	0.06	0.26
D(SF1-SF15)	−0.13	0.19	0.53	0.42
D(SF1-SF16)	0.37	0.19	−0.47	0.62

Table 6: Preference function values of SF1.

Principles	Culture	Automation	Measurement	Sharing
Preference function values of SF1				
D(SF1-SF2)	0.00	0.24	0.08	0.00
D(SF1-SF3)	0.00	0.29	0.11	0.00
D(SF1-SF4)	0.07	0.39	0.29	0.42
D(SF1-SF5)	0.00	0.17	0.35	0.42
D(SF1-SF6)	0.02	0.00	0.32	0.68
D(SF1-SF7)	0.00	0.68	0.00	0.10
D(SF1-SF8)	0.00	0.48	0.06	0.00
D(SF1-SF9)	0.00	0.00	0.16	0.10
D(SF1-SF10)	0.00	0.41	0.27	0.00
D(SF1-SF11)	0.35	0.31	0.11	0.45
D(SF1-SF12)	0.00	0.19	0.24	0.29
D(SF1-SF13)	0.24	0.48	0.35	0.10
D(SF1-SF14)	0.35	0.41	0.06	0.26
D(SF1-SF15)	0.00	0.19	0.53	0.42
D(SF1-SF16)	0.00	0.19	0.00	0.62

(RQ2) How to enhance decision-making capabilities of DevOps experts for considering the particular success factor.

We have applied PROMETHEE-II approach systematically to answer this research question. The questionnaire survey was performed (as explained in Section 3 and Appendix B) to collect data from practitioners (DevOps experts) about relationship of SFs. The pairwise comparison of each SF with DevOps principles was performed. The SFs were prioritized based on their net outranking flow value. The greater the net outranking flow is, the highest the rank of SF is compared with other alternatives (SFs). This approach is effective in understanding logical relationship of SF with DevOps principles (CAMS). This approach provides overall pairwise comparison ranking that will assist DevOps experts in focusing on the areas that need further practices. The results are discussed in Table 9. The results show that "Customer feedback to improve development" SF 8 is the most important SF while dealing with DevOps

Table 7: Aggregate preference function values.

Principles	Culture	Automation	Measurement	Sharing	Total
Aggregate preference function values of SF1					
D(SF1-SF2)	0.00	0.08	0.02	0.00	0.10
D(SF1-SF3)	0.00	0.10	0.03	0.00	0.13
D(SF1-SF4)	0.02	0.14	0.07	0.06	0.30
D(SF1-SF5)	0.00	0.06	0.09	0.06	0.21
D(SF1-SF6)	0.01	0.00	0.08	0.10	0.19
D(SF1-SF7)	0.00	0.24	0.00	0.01	0.25
D(SF1-SF8)	0.00	0.17	0.01	0.00	0.18
D(SF1-SF9)	0.00	0.00	0.04	0.01	0.06
D(SF1-SF10)	0.00	0.14	0.07	0.00	0.21
D(SF1-SF11)	0.12	0.11	0.03	0.07	0.33
D(SF1-SF12)	0.00	0.07	0.06	0.04	0.17
D(SF1-SF13)	0.08	0.17	0.09	0.01	0.36
D(SF1-SF14)	0.12	0.14	0.01	0.04	0.32
D(SF1-SF15)	0.00	0.07	0.13	0.06	0.26
D(SF1-SF16)	0.00	0.07	0.00	0.09	0.16

Table 8: Outranking flow of SFs for DevOps (positive and negative).

Factors	SF1	SF2	SF3	SF4	SF5	SF6	SF7	SF8	SF9	SF10	SF11	SF12	SF13	SF14	SF15	SF16	φ^+
SF1	0	0.1	0.13	0.3	0.21	0.19	0.25	0.18	0.06	0.21	0.33	0.17	0.36	0.32	0.26	0.16	0.22
SF2	0.12	0	0.02	0.31	0.2	0.29	0.21	0.09	0.08	0.12	0.34	0.12	0.37	0.34	0.25	0.13	0.20
SF3	0.13	0.02	0	0.3	0.21	0.29	0.2	0.08	0.09	0.11	0.33	0.12	0.36	0.34	0.26	0.14	0.20
SF4	0	0	0	0	0.04	0.07	0.1	0.03	0	0.01	0.13	0	0.11	0.11	0.08	0.05	0.05
SF5	0.05	0.02	0.04	0.15	0	0.09	0.18	0.11	0	0.08	0.22	0.01	0.24	0.25	0.05	0.04	0.10
SF6	0.11	0.2	0.21	0.26	0.17	0	0.35	0.28	0.05	0.26	0.39	0.18	0.38	0.42	0.18	0.18	0.24
SF7	0.26	0.2	0.2	0.4	0.34	0.43	0	0.12	0.23	0.18	0.46	0.2	0.42	0.41	0.39	0.09	0.29
SF8	0.27	0.16	0.16	0.41	0.36	0.44	0.21	0	0.24	0.14	0.47	0.15	0.44	0.43	0.4	0.23	0.30
SF9	0.13	0.14	0.16	0.37	0.23	0.2	0.3	0.23	0	0.23	0.41	0.18	0.43	0.42	0.29	0.2	0.26
SF10	0.18	0.08	0.07	0.27	0.22	0.3	0.15	0.03	0.13	0	0.37	0.07	0.33	0.34	0.26	0.15	0.20
SF11	0	0	0	0.07	0.06	0.09	0.13	0.06	0.01	0.07	0	0.03	0.12	0.04	0.11	0.02	0.05
SF12	0.2	0.15	0.15	0.33	0.2	0.29	0.23	0.1	0.14	0.13	0.39	0	0.42	0.4	0.25	0.12	0.23
SF13	0	0	0	0.05	0.05	0.09	0.07	0	0	0	0.09	0.03	0	0.06	0.09	0.08	0.04
SF14	0	0.01	0.01	0.08	0.1	0.13	0.09	0.03	0.03	0.05	0.04	0.05	0.1	0	0.14	0.05	0.06
SF15	0.05	0.02	0.03	0.14	0	0.09	0.17	0.1	0	0.08	0.22	0	0.23	0.25	0	0.03	0.09
SF16	0.25	0.21	0.23	0.41	0.29	0.34	0.18	0.23	0.22	0.26	0.44	0.18	0.52	0.46	0.33	0	0.30
φ^-	0.12	0.09	0.09	0.26	0.18	0.22	0.19	0.11	0.09	0.13	0.31	0.10	0.32	0.31	0.22	0.11	

Table 9: Net outranking flow and ranks of SFs.

DevOps success factors	Leaving flow	Entering flow	Out ranking flow	Ranking
"Collaborative and continuous development environment (SF1)"	0.22	0.12	0.10	7
"Continuous meeting schedules, to observe roles and responsibilities of team members (SF2)"	0.20	0.09	0.11	6
"Continuous scalability and performance measures for better release (SF3)"	0.20	0.08	0.12	5
"Continuous measurement for service failure recovery without delay (SF4)"	0.05	0.26	−0.21	13
"High performing project team (SF5)"	0.10	0.18	−0.08	11
"Top management support (SF6)"	0.24	0.22	0.02	10
"Effective communication between development and operational team (SF7)"	0.29	0.2	0.17	3
"Customer feedback to improve development (SF8)"	0.30	0.11	0.19	1
"Vision sharing and goal for change (SF9)"	0.26	0.09	0.09	8
"Training of DevOps activities (SF10)"	0.20	0.13	0.07	9
"Requirement traceability to avoid risks (SF11)"	0.05	0.31	−0.26	15
"Continuous test and integration for assessment (SF12)"	0.23	0.1	0.13	4
"Deployment automation without having delay with production (SF13)"	0.04	0.32	−0.28	16
"Monitoring automation for frequent delivery (SF14)"	0.06	0.31	−0.25	14
"Metrics automation for continuous deployment (SF15)"	0.09	0.22	−0.13	12
"Configuration management for code and infrastructure (SF16)"	0.30	0.12	0.18	2

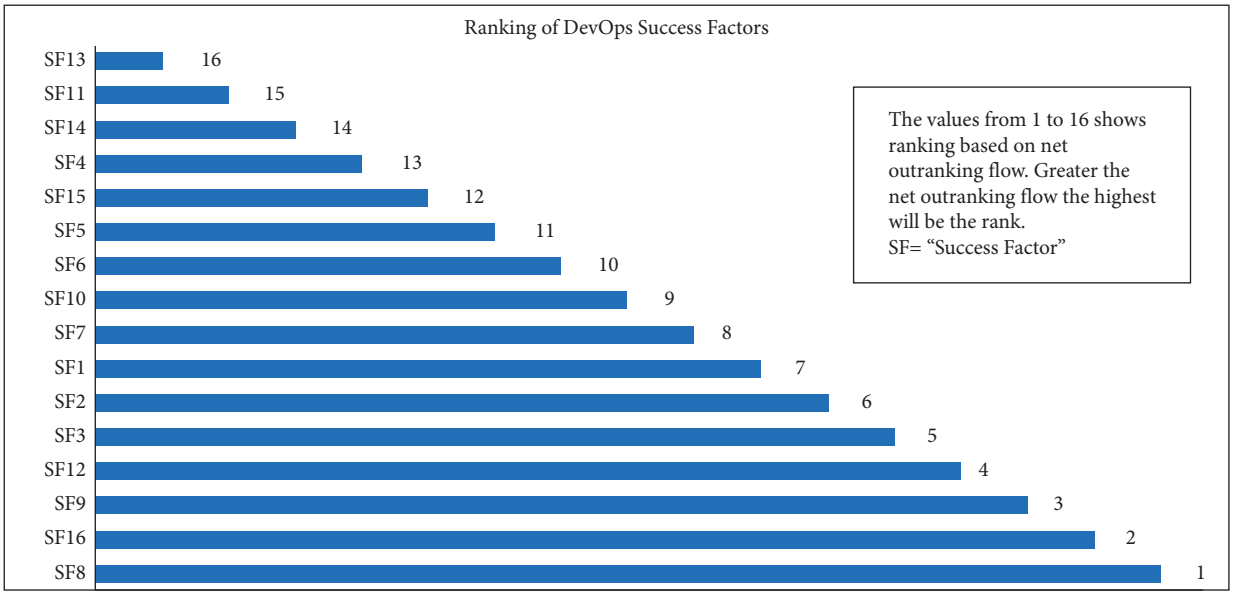

Figure 5: Graphical representation of SFs overall ranking.

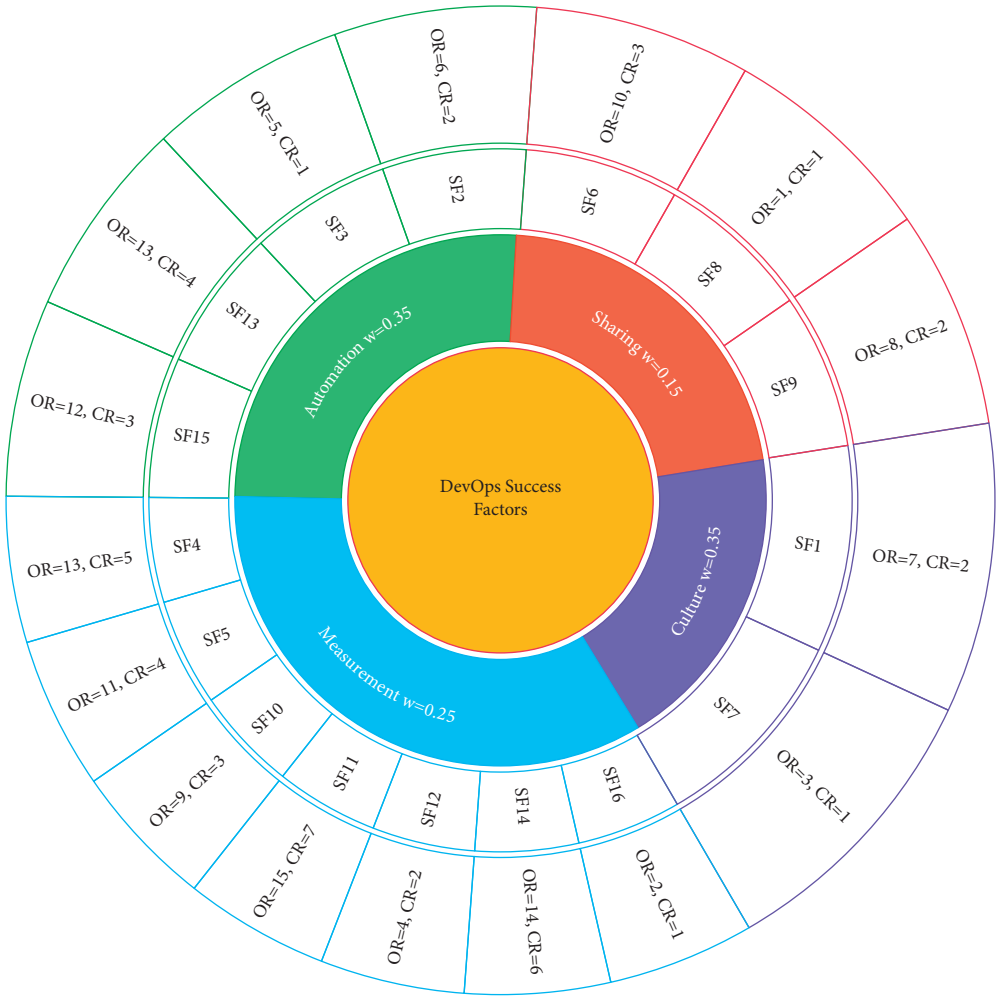

Figure 6: Taxonomy of SFs for DevOps based on overall ranking (OR) and criterion ranking (CR).

implementation. Furthermore, "Configuration management for code and infrastructure" SF16 and "Effective communication between development and operational team" SF7 are the second and third highest ranked factors that can help in progressive implementation of DevOps. Figure 5 shows the graphical representation of SFs according to their ranking.

In addition, we have developed the taxonomy of DevOps SFs based on CAMS criteria (principles of DevOps) [10] (Section 3). The questionnaire survey conducted to validate the ranking and relationship of SFs with mapped category strongly emphasized that all the categories are important to make DevOps process successful. The participants of survey ranked "culture" and "automation" $w = 0.35$ as the highest ranked categories as DevOps overall commitment is about collaboration between teams and continuous deployment in order to deliver software product in manageable time. We further noted that "measurement" and "sharing" ($w = 0.25$, $w = 0.15$) are ranked as second and third ranked categories of DevOps principles for successful implementation of DevOps.

The overall taxonomy of DevOps SFs with their mapped criterion and ranking is presented Figure 6. From Figure 6, it is clear that some factors have overall ranking low but are considered important in their mapped criterion. For example, "Effective communication between development and operational team" SF7 overall ranking is "OR = 3" but in "culture" category, it is ranked as "CR = 1" showing that significance of SF7 is more with respect to mapped category. This taxonomy provides holistic view to DevOps experts about DevOps SFs by focusing on SFs with respect to their overall and criterion ranking. The DevOps experts after knowing the significance of factors can apply new practices to consider these SFs in their organization for better implementation of DevOps.

5.1. Threats to Validity. A potential threat towards the study findings is the biasness in the mapping of success factors in DevOps principles (CAMS criteria). In mapping process, the first two authors of this study perform the mapping procedure via coding scheme (labelling, categories, subcategories, and theory). The initial mapping is verified by authors nos. 3 and 4. Furthermore, we applied the interrater reliable test with external and according to the value of $W = 0.91$; the mapping process is consistent.

Secondly, the sample size $n = 100$ for questionnaire survey study might not be strong enough to justify the results of empirical study. However, based on other research studies of other software engineering domain, the 100 response is a representative set for generalization of study results.

The third threat is about scheduling time with participants for survey analysis. To avoid this threat, we have given them a long time span of more than two months. We also provided our e-mail ID and contact details to the participants in case they need further discussion.

5.2. Study Implications. The ultimate objective of this study is to explore the factors that could positively influence the DevOps process execution, reported by state-of-the-art and state-of-practices. Further, this study addresses the multicriteria decision-making problem using PROMETHEE-II method. The explored success factors and their prioritization based on PROMETHEE-II approach are useful for academic researchers to the development of new and effective strategies by considering the most important influencing areas of DevOps process.

The findings of this study also have practical implication as the study explored and prioritized the important success factors of DevOps paradigm. The practitioners need to consider the highest priority success factors for the successful execution of DevOps activities.

6. Conclusion and Future Directions

Presently, the software market demanded high quality product with continuous delivery in order to satisfy customer with their services. DevOps can increase the communication, reliability, and trust between developers and operations teams and can help software organizations in yielding better outcomes. Therefore, to make DevOps implementation successful for continuous development and delivery of software product, in this study, 16 SFs of DevOps were identified using SLR approach. The identified SFs will assist the DevOps experts towards better implementation of DevOps in their organization.

The identified factors were further mapped into the DevOps basic principles (CAMS) and verified from industrial practitioners by conducting questionnaire survey. We gathered 100 complete responses, which address that the identified SFs are significant for DevOps implementation process. In addition, the PROMETHEE-II technique is used to analyze the logical relationship and ranks of the factors, and their corresponding categories with respect to their significance towards DevOps implementation. The ranking taxonomy was developed, which will help DevOps experts focus on the crucial areas that needs more practices. "Customer feedback to improve development" SF 8 and "Configuration management for code and infrastructure" SF16 are the most important success factors (based on their ranking) that must be considered while working on DevOps implementation. However, "Configuration management for code and infrastructure" SF16 with percentage of 17% shows that less researchers have explored this factor. Therefore, DevOps needs better practices and strategies to improve its implementation in software organization. In future, we will investigate the best practices of DevOps that will help fulfil the requirements of particular success factor. In addition, we will also explore management for code area to improve overall infrastructure of DevOps.

Appendix

Appendix A

Primary studies selected for identification of DevOps success factors.

Link: https://tinyurl.com/y2pfjqst.

Appendix B

Questionnaire sample.
Link: https://tinyurl.com/4jp9kmb4.

Appendix C

Demographic Detail of participants.
Link: https://tinyurl.com/3uvzvcyf.

Appendix D

PROMETHEE analysis of DevOps success factors.
Link: https://tinyurl.com/4j3zmzp3.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors extend their appreciation to the Deanship of Scientific Research at Imam Mohammad Ibn Saud Islamic University for funding this work through research group no. RG-21-07-03.

References

[1] B. S. Farroha and D. L. Farroha, "A framework for managing mission needs, compliance, and trust in the DevOps environment," in *Proceedings of the 2014 IEEE Military Communications Conference*, IEEE, Baltimore, MD, USA, 6 October 2014.

[2] M. Virmani, "Understanding DevOps & bridging the gap from continuous integration to continuous delivery," in *Proceedings of the Fifth International Conference on the Innovative Computing Technology (INTECH 2015)*, IEEE, Galcia, Spain, 20 May 2015.

[3] V. Andrikopoulos, A. Reuter, S. Gómez Sáez, and F. Leymann, "A GENTL approach for cloud application topologies," in *Proceedings of the European Conference on Service-Oriented and Cloud Computing*, pp. 148–159, Springer, Manchester, UK, 2 September 2014.

[4] L. E. Lwakatare, P. Kuvaja, and M. Oivo, "Dimensions of devops," in *Proceedings of the International conference on agile software development*, pp. 212–217, Springer, Helsinki, Finland, 25 May 2015.

[5] J. Waller, N. C. Ehmke, and W. Hasselbring, "Including performance benchmarks into continuous integration to enable DevOps," *ACM SIGSOFT - Software Engineering Notes*, vol. 40, no. 2, pp. 1–4, 2015.

[6] F. Erich, C. Amrit, and M. Daneva, "A mapping study on cooperation between information system development and operations," in *Proceedings of the International Conference on Product-Focused Software Process Improvement*, pp. 277–280, Springer, Helsinki, Finland, 10 December 2014.

[7] S. K. Bang, S. Chung, Y. Choh, and M. Dupuis, "A grounded theory analysis of modern web applications: knowledge, skills, and abilities for DevOps," in *Proceedings of the 2nd annual conference on Research in information technology*, pp. 61-62, Orlando Florida USA, 10 October 2013.

[8] S. Rafi, W. Yu, M. A. Akbar, S. Mahmood, A. Alsanad, and A. Gumaei, *Readiness model for DevOps implementation in software organizations. Journal of Software: Evolution and Process*, vol. 33, no. 4, p. e2323, 2021.

[9] S. Rafi, W. Yu, and M. A. Akbar, "RMDevOps: a road map for improvement in DevOps activities in context of software organizations," in *Proceedings of the Evaluation and Assessment in Software Engineering*, pp. 413–418, Trondheim Norway, 15 April 2020.

[10] J. Humble and J. Molesky, "Why enterprises must adopt devops to enable continuous delivery," *Cutter IT Journal*, vol. 24, no. 8, p. 6, 2011.

[11] T. Peuraniemi, "DevOps, value-driven principles, methodologies and tools," *Data-and Value-Driven Software Engineering with Deep Customer Insight*, vol. 43, 2017.

[12] A. Dyck, R. Penners, and H. Lichter, "Towards definitions for release engineering and DevOps," in *Proceedings of the 2015 IEEE/ACM 3rd International Workshop on Release Engineering*, IEEE, Florence, Italy, 30 July 2015.

[13] F. Erich, C. Amrit, and M. Daneva, "Cooperation between information system development and operations: a literature review," in *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, ACM, Torino Italy, 18 September 2014.

[14] R. W. Macarthy and J. M. Bass, "An empirical taxonomy of DevOps in practice," in *Proceedings of the 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, pp. 221–228, IEEE, Portoroz, Slovenia, 26 August 2020.

[15] M. Gokarna and R. Singh, "DevOps: a historical review and future works," in *Proceedings of the 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, pp. 366–371, IEEE, Greater Noida, India, 19 February 2021.

[16] A. Mishra and Z. Otaiwi, "DevOps and software quality: a systematic mapping," *Computer Science Review*, vol. 38, Article ID 100308, 2020.

[17] S. Rafi, W. Yu, M. A. Akbar, A. Alsanad, and A. Gumaei, "Multicriteria based decision making of DevOps data quality assessment challenges using fuzzy TOPSIS," *IEEE Access*, vol. 8, pp. 46958–46980, 2020.

[18] J. Smeds, K. Nybom, and I. Porres, "DevOps: a definition and perceived adoption impediments," in *Proceedings of the International conference on agile software development*, Springer, Helsinki, Finland, 25 May 2015.

[19] L. Leite, C. Rocha, F. Kon, D. Milojicic, and P. Meirelles, "A survey of DevOps concepts and challenges," *ACM Computing Surveys*, vol. 52, no. 6, pp. 1–35, 2019.

[20] J. P. Brans, P. Vincke, and B. Mareschal, "How to select and how to rank projects: the PROMETHEE method," *European Journal of Operational Research*, vol. 24, no. 2, pp. 228–238, 1986.

[21] V. Balali, B. Zahraie, and A. Roozbahani, "A comparison of AHP and PROMETHEE family decision making methods for selection of building structural system," *American Journal of Civil Engineering and Architecture*, vol. 2, no. 5, pp. 149–159, 2014.

[22] A. P. U. Siahaan and M. Mesran, *Best Student Selection Using Extended PROMETHEE-II Method*, osf.io, 2017.

[23] P. Liu and Z. Guan, "Evaluation research on the quality of the railway passenger service based on the linguistic variables and

the improved PROMETHEE-II method," *Journal of Computers*, vol. 4, no. 3, pp. 265–270, 2009.

[24] H. Zhao, Y. Peng, and W. Li, "Revised PROMETHEE II for improving efficiency in emergency response," *Procedia Computer Science*, vol. 17, pp. 181–188, 2013.

[25] B. Kitchenham and S. Charters, "Guidelines for performing systematic literature reviews in software engineering," *Cités*, vol. 1, 2007.

[26] W. A. Babchuk, "Glaser or Strauss? Grounded theory and adult education," in *Proceedings of the 15th Annual Midwest Research-to-Practice Conference in Adult, Continuing, and Community Education*, ERIC, Lincoln, Nebraska, 17 October 1996.

[27] K. A. Hallgren, "Computing inter-rater reliability for observational data: an overview and tutorial," *Tutorials in Quantitative Methods for Psychology*, vol. 8, no. 1, pp. 23–34, 2012.

[28] M. A. Akbar, M. Shafiq, T. Kamal, M. T. Riaz, and M. K. Shad, "An empirical study investigation of task allocation process barriers in the context of offshore software development outsourcing: an organization size based analysis," *International Journal of Computing and Digital Systems*, vol. 8, no. 04, pp. 343–350, 2019.

[29] A. Yagüe, J. Garbajosa, J. Díaz, and E. González, *An exploratory study in communication in Agile Global Software Development. Computer Standards & Interfaces*, vol. 48, pp. 184–197, 2016.

[30] J. D. Herbsleb, D. J. Paulish, and M. Bass, "Global software development at siemens: experience from nine projects," in *Proceedings of the 27th international conference on Software engineering*, pp. 524–533, IEEE, St. Louis, MO, USA, 15 May 2005.

[31] A. A. Khan, J. Keung, M. Niazi, S. Hussain, and A. Ahmad, "Systematic literature review and empirical investigation of barriers to process improvement in global software development: client-vendor perspective," *Information and Software Technology*, vol. 87, pp. 180–205, 2017.

[32] S. Easterbrook, J. Singer, M.-A. Storey, and D. Damian, "Selecting empirical methods for software engineering research," in *Guide to Advanced Empirical Software Engineering*pp. 285–311, Springer, London, England, 2008.

[33] M. Shameem, R. R. Kumar, C. Kumar, B. Chandra, and A. A. Khan, "Prioritizing challenges of agile process in distributed software development environment using analytic hierarchy process," *Journal of Software: Evolution and Process*, vol. 30, no. 11, p. e1979, 2018.

[34] T. Yaghoobi, "Prioritizing key success factors of software projects using fuzzy AHP," *Journal of Software: Evolution and process*, vol. 30, no. 1, p. e1891, 2018.

[35] M. Niazi, S. Mahmood, M. Alshayeb, A. M. Qureshi, K. Faisal, and N. Cerpa, "Toward successful project management in global software development," *International Journal of Project Management*, vol. 34, no. 8, pp. 1553–1567, 2016.

[36] M. A. Akbar, J. Sang, A. A. Khan et al., "Statistical analysis of the effects of heavyweight and lightweight methodologies on the six-pointed star model," *IEEE Access*, vol. 6, pp. 8066–8079, 2018.

[37] E. Sloane, M. J. Liberatore, R. L. Nydick, W. Luo, and Q. B. Chung, "Clinical engineering technology assessment decision support: a case study using the analytic hierarchy process (AHP)," in *Proceedings of the Second Joint 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society][Engineering in Medicine and Biology*, IEEE, Houston, TX, USA, 23 October 2002.

[38] K. Finstad, "Response interpolation and scale sensitivity: evidence against 5-point scales," *Journal of usability studies*, vol. 5, no. 3, pp. 104–110, 2010.

[39] T. M. Amaral and A. P. C. Costa, "Improving decision-making and management of hospital resources: an application of the PROMETHEE II method in an Emergency Department," *Operations Research for Health Care*, vol. 3, no. 1, pp. 1–6, 2014.

[40] I. Veza, S. Celar, and I. Peronja, "Competences-based comparison and ranking of industrial enterprises using PROMETHEE method," *Procedia Engineering*, vol. 100, pp. 445–449, 2015.

[41] M. Behzadian, R. B. Kazemzadeh, A. Albadvi, and M. Aghdasi, "PROMETHEE: a comprehensive literature review on methodologies and applications," *European Journal of Operational Research*, vol. 200, no. 1, pp. 198–215, 2010.

[42] G. Kabra, A. Ramesh, and K. Arshinder, "Identification and prioritization of coordination barriers in humanitarian supply chain management," *International Journal of Disaster Risk Reduction*, vol. 13, pp. 128–138, 2015.

[43] J. P. Brans, B. Mareschal, and V. Promethee, "Promethee V: mcdm problems with segmentation constraints," *INFOR: Information Systems and Operational Research*, vol. 30, no. 2, pp. 85–96, 1992.

[44] S. Rafi, W. Yu, and M. A. Akbar, "Towards a hypothetical framework to secure DevOps adoption: grounded theory approach," in *Proceedings of the Evaluation and Assessment in Software Engineering*, pp. 457–462, ACM, Trondheim Norway, 15 April 2020.

[45] S. N. Mullaguru, "Changing scenario of testing paradigms using DevOps–A comparative study with classical models," *Global Journal of Computer Science and Technology*, vol. 15, 2015.

[46] H. Myrbakken and R. Colomo-Palacios, "DevSecOps: a multivocal literature review," in *Proceedings of the International Conference on Software Process Improvement and Capability Determination*, Springer, Palma de Mallorca, Spain, 4 October 2017.