# Current DevOps Teaching Techniques: A Systematic Literature Review

Pedro S. C. Garcia
DECOM - UFOP
Ouro Preto, Brazil
pedro.clair@aluno.edu.ufop.br

José Ferreira
DCC - UFMG
Belo Horizonte, Brazil
josenicuri3@gmail.com

Matheus Gonçalves
DCC - UFMG
Belo Horizonte, Brazil
matheusgpm@gmail.com

Tiago Carneiro
DECOM - UFOP
Ouro Preto, Brazil
tiago@ufop.edu.br

Eduardo Figueiredo
DCC - UFMG
Belo Horizonte, Brazil
figueiredo.dcc2@gmail.com

Igor M. Pereira
DESCI - UFOP
Ouro Preto, Brazil
igormuzetti@ufop.edu.br

## ABSTRACT

DevOps is a set of practices that deals with the coordination between development and operations, in the context of teams, with the main objective of ensuring continuous integration and delivery, quality, and reliability. DevOps, which builds upon modern processes, is currently evolving. So, finding specific techniques for teaching it is crucial. This study aims to identify in the software engineering literature the existence of studies that have used techniques to teach DevOps in the last five years. To achieve this, a systematic literature review was conducted to identify teaching techniques applied in the DevOps contexts, as well as the associated benefits and challenges. From 27 papers with 40 cited techniques, we found that the most common teaching techniques are Project-Based Learning followed by Collaborative Learning and the Flipped Classroom, where 70% can be considered active methods. The benefits include increased student engagement, simulation of real industry experience, and improved practical and technical skills. However, challenges arise in validating student performance, keeping up with the rapidly changing DevOps content, and finding sufficient time to teach during a course.

## CCS CONCEPTS

• **Social and professional topics → Computing education**; • **Software and its engineering → Software creation and management**.

## KEYWORDS

DevOps, Education, Teaching Techniques

## 1 INTRODUCTION

DevOps is a set of practices that deals with the coordination between development and operations, in the context of teams, with the main objective of ensuring continuous integration and delivery, quality, and reliability [12, 13, 24]. In this context, it can be considered an evolution of the agile movement [2, 5–7, 22]. This, in turn, was born in 2001, to improve the waterfall model delivery process [29]. More recent studies on the evolution of DevOps link its good performance to the business needs of industrial software [6, 26]. It can be said that DevOps uses a set of software engineering activities together with support tools, adapting the way teams develop software systems [16, 27].

Figure 1 presents DevOps' incorporation into the management system by a tool. In addition, waterfall and agile models also can be implemented. Independent of the specific method adopted, the fundamental stages encompassing requirements analysis (PLAN in Figure 1), construction (BUILD in Figure 1), testing (TEST in Figure 1, and maintenance (DEPLOY, OPERATE, and OBSERVE in Figure 1) are universally applied in software engineering [8, 32, 33]. These stages serve as foundational pillars within computer science courses that integrate software engineering subjects into their syllabi. Consequently, DevOps tools, concepts, practices, values, and principles are intricately interconnected with and relevant to such courses, enriching the pedagogical landscape by aligning with established software engineering frameworks and methodologies.

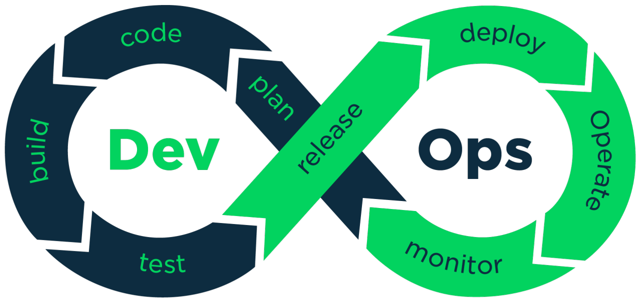

**Figure 1: Suggested implementation of the DevOps flow in the context of software engineering [11].**

So, this study employed an SLR to identify effective teaching techniques for DevOps education. Our study utilized three online search engines. Initially, we extracted 201 papers, and through the snowballing process, we incorporated approximately 240 papers. This comprehensive approach yielded 27 papers referencing a total of 40 distinct teaching techniques. Project-based learning, collaborative learning, and the flipped classroom emerged as the most prevalent approaches, all emphasizing active learning methodologies. These techniques prioritize practical experience and teamwork, fostering a comprehensive understanding of DevOps principles. By implementing such methods, educators can equip the next generation of developers with the essential skills to navigate the challenges of modern software development [31].

In this context, the goal of this study was to identify recent literature (within the last 5 years) related to the software engineering course that employed educational techniques to teach DevOps (practice, values, principles, and tools).

For this purpose, a systematic literature review (SLR) was conducted. Section 2 showed a background of the concepts necessary for understanding the paper; Section 3 related the methods used, followed by Section 4 with results and discussions. In sequence, Section 5 is the threats of viability, and finally, Section 6 the relateds work and 7 conclusion of the paper followed by acknowledgments and bibliographic references.

## 2 BACKGROUND

In this section, we delve into the concepts, principles, values and tool of DevOps and teaching techniques.

### 2.1 DevOps

DevOps aims to establish a mindset that focuses on a closer collaboration between teams by setting the common goal to develop high-quality software systems and operate resilient systems [12]. More recently [24] defines that DevOps is a collaborative and multidisciplinary effort within an organization to automate continuous delivery of new software versions while guaranteeing their correctness and reliability.

In a practice away, the term "DevOps" refers to the automated integration of software development and IT teams, which is achieved through the implementation of a set of defined practices and tools [4]. This represents a significant cultural shift in the way in which software development and IT operations collaborate throughout the software development lifecycle. The approach emphasizes communication, collaboration, integration, and automation [20], to improve the speed and quality of software delivery.

Since the advent of DevOps in 2007-2008, organizations have continued to adopt this approach at an increasing rate [3]. There are gaps, such as the lack of industrial support during product configuration [28]. The key elements of DevOps include continuous integration (CI), continuous delivery (CD), infrastructure as code (IaC), and automated testing, for example in this case, to reduce bad smells [14]. By integrating these practices, organizations can accelerate the development and deployment of software while maintaining a high level of quality. The evolution of DevOps has been facilitated by advancements in technology, including cloud computing, containerization, and microservices architecture. These technologies provide the necessary infrastructure and tools to implement DevOps practices effectively.

Furthermore, the DevOps approach places significant emphasis on the importance of feedback loops and continuous improvement. By collecting and analyzing feedback from both the development and operations teams, organizations can identify areas for enhancement and refine their processes iteratively. This iterative approach fosters a culture of continuous improvement, whereby teams are encouraged to experiment, learn from failures, and implement changes accordingly. The feedback-driven model not only results in the accelerated delivery of software but also ensures that the delivered software aligns with the evolving needs and expectations of users.

### 2.2 Teaching Techniques

In this section, we discuss the methods that encompass several teaching techniques used in DevOps courses.

**Active Method:** Alter the emphasis from a passive reception of information to an active engagement in the learning process [21]. Such techniques require the performance of practical activities that necessitate critical thinking, problem-solving, and collaboration with others[21]. This hands-on approach fosters a more profound comprehension of the subject matter and the capacity to apply knowledge in authentic contexts [21]. Based on the definition extracted by [13], we can cite as examples these three teaching techniques: "Project-based Learning (PjBL)", Collaborative Learning, and Flipped Classroom.

**Delivery Method:** The term "delivery method" encompasses the various approaches employed to convey information and facilitate learning [1]. These methods serve as the vehicles through which knowledge is transported from instructors to learners, thereby shaping the learning experience and impacting its effectiveness. [1].

**Others** These are specific approaches that might not fall into the other categories. Look for mentions of "Agile process" or "Gamification" in the resume section. The definitions of the main techniques can be seen in Table 1.

## 3 STUDY DESIGN

This section describes the process for conducting a systematic literature review on the teaching of DevOps in a software engineering context. A systematic literature review is a method which sets out a series of steps to methodically organize the review [10]. Systematic reviews aim to present a fair evaluation of a research topic by using a trustworthy, rigorous, and auditable methodology [10, 23].

Section 3.1 presents the research questions, Section 3.2 outlines the search strategy for data collection, Section 3.3 establishes the selection criteria for inclusion and exclusion, Section 3.5 delineates the data analysis methods, Section 3.4 details the process of data extraction, and Section 3.6 concludes the methodology by reporting the findings.

### 3.1 Research Question

The first step of this systematic literature review aims to identify techniques for teaching DevOps principles and practices within software engineering education. The review period

**Table 1: Teaching techniques definitions and method classification**

| Technique | Definition | Method |
|---|---|---|
| Project-Based Learning | It focuses on a project in which the students work on a concrete task. | Active |
| Flipped Classroom | Activities traditionally conducted in the classroom become home activities (vice versa). | |
| Lecture | It is the traditional teaching method in educational institutions in which the lecturer directly instructs the students. | Delivery |
| Labs | It involves accomplishing practical tasks exploring a software engineering topic usually conducted in dedicated rooms equipped with computers for each student. | |
| Agile Process | Use of agile activities during course execution like sprints and scrum planning. | Other |

encompasses the past five years (2019-2023) to capture the latest advancements in DevOps education. We investigate the following research questions:

- **RQ1:** What are the current techniques for teaching DevOps in the software engineering context?

This question guides the core investigation into identifying pedagogical approaches for integrating DevOps concepts and practices into software engineering curricula.

- **RQ2:** What are the benefits of these techniques?

By exploring the advantages associated with these teaching techniques, we aimed to understand how they contributed to student learning outcomes and overall effectiveness in DevOps education.

- **RQ3:** What are the challenges of using these techniques?

Examining the potential challenges faced when implementing these techniques may provide valuable insights for educators and curriculum developers. Understanding these challenges also allows for the development of strategies to mitigate them and optimize the teaching of DevOps within software engineering education.

## 3.2 Search Strategy

The second step is to develop a comprehensive search strategy to identify relevant studies. Figure 3 shows the process as a whole in the flowchart. At the start of the work (or pre-study) there was a return from the Check-up for the String, in case we decided to make any adjustments. Approximately 3 tests were carried out on 2 search engines from our list, where we empirically verified that the string in the paper proved to be better.

The search strategy includes a combination of keywords and Boolean operators. The search terms used in this study result in following string:

```
(("software engineering" OR "software development")
AND devops AND education)
```

The following databases were used to identify studies: ACM Digital Library, IEEE Xplore, and Science Direct, this represents the ***"Search"*** in Figure 3. The ***"Check-up"*** step in Figure 3 consists of an initial reading of the paper, taking into account the title and abstract, thus classifying them for the next step.

Finally, an interaction of a ***"Snowball"*** was realized, where the backward snowballing represents the references of the paper and the forward represents the citation. Google Scholar was used to verify the number of citations each paper received.

## 3.3 Select Criteria

The third step of this study is to screen the studies based on eligibility criteria. The eligibility criteria should be clearly defined and should include the following:

- Studies published in the English language
- Studies published between 2019-2023
- Studies that focus on DevOps Education in a software engineering context

On the other hands, the exclude criteria are:

- Lack of specific techniques in teaching.
- The absence of specific concepts, practices, values, or principles related to DevOps.
- Any paper not available in full-text.

## 3.4 Data Extraction

The fifth step was to apply the string in the search engines. Table 2 displays the databases used and the number of documents extracted when the initial string was applied.

**Table 2: Results from the search string**

| Sourcer Engine | Quantity |
|---|---|
| ACM Digital Library | 25 |
| IEE Explore | 60 |
| Science Direct | 126 |

Figure 3 illustrates the workflow followed in this study. Initially, thirty-one papers were pre-selected using an automated method that assessed the presence of the keywords

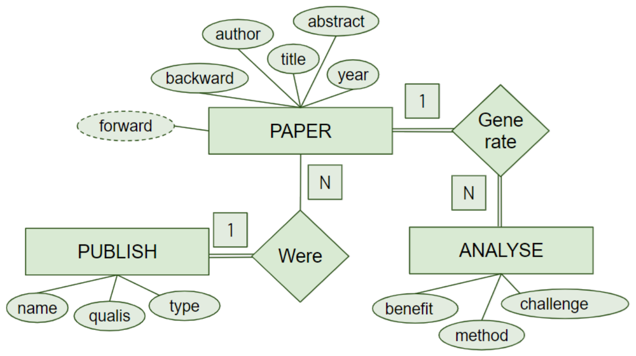

**Figure 2: Entity-Relationship Model to Data Extraction, Data Synthesis and Quality**

"software," "DevOps," and "education" in the title or abstract of each paper.

Subsequently, inclusion and exclusion criteria were applied. The process generally went as follows: The first author analyzed the articles and shortlisted them, the other co-authors reviewed and questioned the first author and discussions took place until a group consensus was reached.

It resulted in ten papers for the initial stage of the Snowball process. This process expanded the dataset by adding twenty additional papers sourced from 248 references and 90 citations. During the Snowball sampling process, one paper was excluded from the study due to inadequate or incomplete referencing. Two papers were also systematic literature reviews; however, they were disregarded as primary studies for analysis but are included in the related works. The paper"s list are listed in the table 3

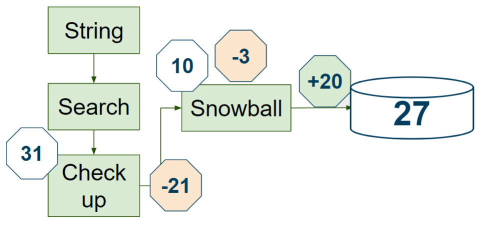

**Figure 3: The process followed to extract the papers for analysis**

## 3.5 Data Analysis

The fourth step involves extracting pertinent data from the chosen studies. In Figure 2, we observe the attributes slated for analysis, categorized into three entities.

Under "PAPER", we find essential information such as Author(s), Abstract, Year, and Title, serving as foundational details of the paper. Following this, "ANALYSIS" encompasses Benefits, Technique(s), and Challenges, representing individual insights. It"s noteworthy that a paper may undergo multiple analyses, reflecting the diverse teaching techniques it incorporates.

Lastly, "PUBLISH" entails Name, Qualis, and Type, gauging the quality of selected studies based on their publication.

## 3.6 Reporting

Finally, the data was summarized using an appropriate analytical approach. The themes that emerge was used to answer the research questions. The findings of the systematic literature review was reported based on the qualitative and quantitative approach.

## 4 RESULTS AND DISCUSSIONS

The papers' publication dates were extracted and presented in Figure 6 to understand the evolution over the years. From 2019 to 2023, there were 7, 2, 6, 5, and 7 publications, respectively. This indicates a growing trend over this period.

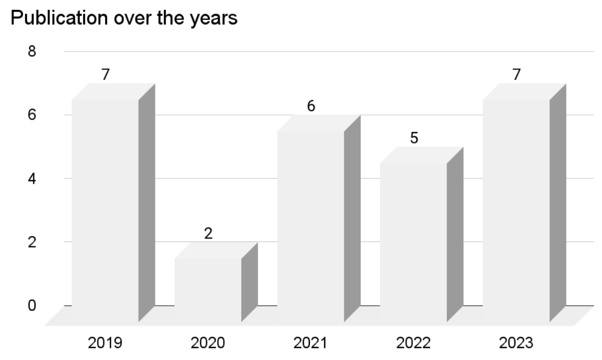

**Figure 4: The timeline of publications considering the last 5 years (2019-2023)**

These publications were classified as teaching techniques. So, we respond to the first research questions posed in our study.

**RQ1:** What are the current techniques for teaching DevOps in the software engineering context?

Figure 6 illustrates the main techniques used, with the four most frequently mentioned being Project-based Learning (PjBL) with 10 mentions, Collaborative Learning with 5 mentions, Flipped Classroom with 4, Agile Process and Labs with 3. This result was the starting point for our second and third research questions, where our focus was on the three main techniques.

**RQ2:** What are the benefits of these techniques?

Project-Based Learning (PjBL) was the most used technique. It is stated that "In the first project follow-up session each group presents the chosen product", which introduces the concept of the technique at the beginning of the learning

**Table 3: DevOps Education papers with Publication Year**

| No. | paper Title | Year | Venue |
|---|---|---|---|
| A1 | Analysing the SWECOM Standard for Designing a DevOps Education Programme | 2019 | FISEE |
| A2 | Design of a (yet another?) devops course | 2019 | DEVOPS |
| A3 | Devops-preparing students for professional practice | 2019 | FIE |
| A4 | Grounded Theory for DevOps Education | 2019 | ICSME |
| A5 | Industry-academy collaboration in teaching DevOps and continuous delivery to software engineering students: towards improved industrial relevance in higher education | 2019 | ICSE |
| A6 | Teaching DevOps and cloud based software engineering in university curricula | 2019 | e-Science |
| A7 | Teaching DevOps in Academia and Industry: Reflections and Vision | 2019 | DEVOPS |
| A8 | Threading DevOps practices through a university software engineering programme. | 2020 | CSEE&T |
| A9 | Understanding devops education with grounded theory. | 2020 | ICSE |
| A10 | Analyzing DevOps Teaching Strategies: An Initial Study | 2021 | SBES |
| A11 | Combining Agile and DevOps to Improve Students" Tech and Non-tech Skills. | 2021 | CSEDU |
| A12 | DevOps Research-Based Teaching Using Qualitative Research and Inter-Coder Agreement | 2021 | TSE |
| A13 | Qualifying Software Engineers Undergraduates in DevOps - Challenges of Introducing Technical and Non-technical Concepts in a Project-oriented Course | 2021 | ICSE |
| A14 | Shifting traditional undergraduate software engineering instruction to a DevOps focus | 2021 | JCSC |
| A15 | Teaching DevOps: a tale of two universities. | 2021 | SPLASH |
| A16 | Achievement unlocked: a case study on gamifying DevOps practices in industry | 2022 | ESEC/FSE |
| A17 | DevOps education: an interview study of challenges and recommendations | 2022 | ICSE |
| A18 | Exploring the benefits of combining DevOps and agile | 2022 | Future Internet |
| A19 | Teaching Guide for Beginnings in DevOps and Continuous Delivery in AWS Focused on the Society 5.0 Skillset. | 2022 | RITA |
| A20 | The Complications and Solutions of Using DevOps in the decipline | 2022 | JISIT |
| A21 | A Serious Game with Which to Introduce Users to the World of DevOps | 2023 | CSEDU |
| A22 | Leveraging Teaching Methods to Overcome Challenges in DevOps Education. | 2023 | ZKGI |
| A23 | Overcoming Challenges in DevOps Education through Teaching Methods | 2023 | ICSE |
| A24 | Preparing Students for Software Production with DevOps: A Graduate Course Approach | 2023 | CCSC |
| A25 | Software Engineering Education in the DevOps Era: Experiences and Recommendations | 2023 | CIBSE |
| A26 | Teaching DevOps and Software Engineering Practices Using an Automated Programming Assessment System | 2023 | JC |
| A27 | Unveiling the Teaching Methods Adopted in DevOps Courses | 2023 | SBQS |

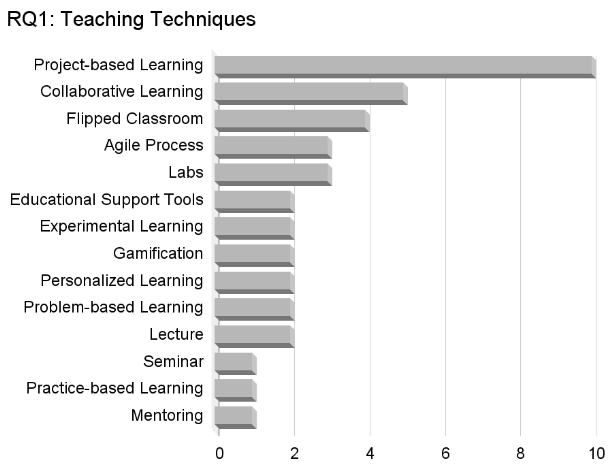

**Figure 5: The 40 teaching techniques cities by papers**

process [A7]. The benefits of this approach, based on the analyses conducted, are categorized as follows:

**Student Engagement:** PjBL enhances student engagement. Project-based learning tends to be more engaging than traditional methods, as students have the opportunity to apply their knowledge in practical and meaningful situations [A10]. The teacher provides the student with an audience to present the final work of the project, which increases motivation.

**Personalized Learning and Individualized Support:** PjBL allows students to work on projects tailored to their skills and interests, promoting personalized learning [A23]. In addition, instructors can provide individualized support as needed, assisting students in their learning process.

**Preparation for the Workforce:** PjBL provides a platform for the development of practical skills such as problem-solving, communication, teamwork, and critical thinking. Projects often require collaboration among students, preparing them for the collaborative work environment found in the IT industry [A22].

**Encouragement of Research and Innovation:** PjBL often encourages students to seek out new technologies and approaches [A2]. This exposes them to new ideas and encourages them to be innovative in their problem-solving approaches.

In summary, these outcome is particularly relevant in the context of teaching DevOps, where experience with projects is essential for understanding the principles and practices of DevOps. Project-based learning provides a dynamic and engaging approach that not only fosters knowledge acquisition but also cultivates practical skills, thus preparing students for success in the workforce. By engaging in project-based learning activities, students not only gain theoretical knowledge but also develop the practical skills necessary for implementing DevOps processes effectively in professional settings.

Otherwise, collaborative learning stands out as a promising teaching method for DevOps courses, offering general benefits for students, educators, and the labor market. The results of the study indicate that collaborative learning contributes to:

**Enhancing learning:** The collaborative environment promotes experimentation, reflection and the development of practical skills, complemented by lectures aligned with the project's objectives and covering the theoretical basis of DevOps.

**Meet industry demand:** The collaborative approach prepares students for real DevOps practices, focusing on code quality, effective communication, and team collaboration, essential skills for meeting industry demands.

**Improve course planning:** The research provides valuable insights for DevOps educators, improving course planning and the selection of effective teaching methods.

**Increase knowledge retention:** Collaborative learning facilitates knowledge retention and understanding of complex concepts, preparing students for real-world challenges.

**Improve the student experience:** The collaborative approach increases student involvement and participation in the learning process, making the experience more meaningful and rewarding.

The flipped classroom model emerges as a promising pedagogical approach within software engineering education, particularly in the context of DevOps, offering multifaceted benefits for students, and instructors. Our empirical findings indicate that the implementation of the flipped classroom yields the following advantages for teaching DevOps.

**Enhanced Software Engineering Instruction:** The flipped classroom empowers students to autonomously delve into theoretical underpinnings, thereby liberating valuable class time for practical, collaborative endeavors. This paradigm shift fosters deeper comprehension and application of knowledge.

**Augmented Student Engagement:** The interactive nature and emphasis on teamwork inherent in the flipped classroom paradigm cultivate dynamic and participatory learning, thereby stimulating active student involvement in the educational process.

**Elevated Practical Proficiency:** Exposure to hands-on experiences utilizing industry-standard tools such as Git for source code management and Continuous Integration practices equips students with indispensable competencies requisite for success within the labor market.

**Streamlined Infrastructure Deployment and Uniformity:** The flipped classroom model streamlines the swift deployment and uniformity of infrastructure configurations, thereby preparing students to navigate the exigencies of the perpetually evolving digital landscape.

**RQ3:** What are the challenges of these techniques?

The implementation of Project-Based Learning (PjBL) techniques also presents a series of challenges that need to be considered.

**Validation of Results:** Evaluating student performance and validating results obtained through PjBL can be challenging, especially because assessment may be more subjective than in traditional methods. Shift from the traditional teaching paradigm: Adopting PjBL requires a significant shift in how teaching is conceived and conducted, which may encounter resistance from educators and institutions accustomed to traditional methods.

**Constant updating of content:** The field of DevOps is constantly evolving, requiring continuous updating of course content to remain relevant and up-to-date. Difficulty for students to assimilate learning: Some students may struggle to adapt to the project-based learning approach, especially if they are accustomed to more passive teaching methods.

**Time management:** PjBL can require more time and effort from both students and educators, which can be challenging to manage, especially in courses with high workload.

**Selection and integration of appropriate tools:** Choosing and configuring the right tools for projects can be complex and require specialized technical knowledge, both from educators and students.

**Difficulty in finding realistic projects:** Finding projects that are relevant and realistic for students can be challenging, especially in a field like DevOps that involves a wide range of skills and knowledge. Difference between academic knowledge and market knowledge: Aligning course objectives with job market demands can be complicated, especially if there is a disconnect between the academic knowledge taught and the skills required by companies.

**Supervision and monitoring of students:** Supervising and monitoring students' progress throughout projects can be labor-intensive and require a significant time investment from educators.

**Integration of DevOps into software engineering programs:** Integrating DevOps into existing software engineering programs may require a significant curriculum overhaul and a program-level approach, not just isolated courses.

**Blending technical DevOps practices with human-centered design principles:** Integrating technical DevOps practices with human-centered design principles can be challenging due to the different approaches and focuses of these disciplines.

These challenges highlight the complexity involved in successfully implementing project-based learning techniques, especially in a dynamic field like DevOps. It is important for educators and institutions to be aware of these challenges and develop strategies to effectively address them.

This research also underscores the potential of collaborative learning as a promising teaching method for DevOps courses, while also acknowledging the challenges that must be overcome for its effective implementation.

**Lack of Research:** The research on teaching methods employing collaborative learning in DevOps Education remains limited. This indicates that educators may not be aware of available and recommended teaching approaches, and how to effectively apply them.

**Assessment:** Assessing student comprehension following collaborative learning activities can pose a challenge. The complex nature of collaborative work hinders individual measurement of learning and the allocation of credit to students.

**Teaching DevOps Culture and Concepts:** Instructing students on the culture and concepts of DevOps, particularly those without industrial experience, can be challenging even in a collaborative environment. Additional support may be necessary for students to grasp the practical application of DevOps principles.

**Multidisciplinary Skills:** DevOps instruction necessitates multidisciplinary skills and mastery of complex execution architectures. This may impede collaborative learning, as students may possess varying levels of experience and knowledge in relevant areas.

**Evolving Technologies:** DevOps and continuous delivery technologies are continually evolving, rendering knowledge acquisition an ongoing process. This can pose a challenge for students and educators in a collaborative setting.

**Industrial Relevance:** Maintaining industrial relevance, instructional development, and operational activities in a collaborative environment can be challenging. Educators must stay abreast of the latest industry trends and technologies to ensure students are learning the most relevant skills.

**Tools:** Inadequate tools for practical activities may hinder student engagement and collaboration. It is crucial for educators to have access to tools that facilitate collaborative learning and the practical application of DevOps concepts.

**Resources:** Establishing a comprehensive DevOps process in a classroom setting may be challenging due to resource constraints. Educators may need to be innovative and find ways to simulate real-world DevOps scenarios with limited resources.

**Time:** Limited time to effectively teach DevOps can be an even greater challenge in a collaborative environment. Educators must be efficient and utilize classroom time effectively to ensure students grasp the most important concepts.

Although the implementation of collaborative learning in DevOps education presents obstacles, its potential to enhance the teaching process is undeniable. By overcoming these challenges, we can build more dynamic and efficient learning environments.

Finally, the research highlights the potential of the flipped classroom as a promising teaching method for DevOps courses, while also recognizing the challenges that need to be overcome for its effective implementation.

**Tool Selection:** Choosing appropriate tools for the flipped classroom environment is crucial to ensure the method's success. This requires educators to be familiar with available tools and select those that best meet students' needs and course objectives.

**Integration of Theory and Practice:** The flipped classroom requires careful integration of theory with practice. Students should have the opportunity to apply learned concepts in practical activities and receive timely feedback to solidify their learning.

**Content Standardization:** Lack of standardization in DevOps curricula and teaching models complicates the implementation of the flipped classroom. Standardized materials and resources need to be developed that can be easily adapted to different teaching contexts. Incorporation of Practices and Tools: Effective integration of DevOps practices and tools in the flipped classroom requires careful planning and adequate training for educators.

**Clear and Well-Defined Tasks:** Vague task formulation can lead to significant variation in student submissions, making assessment and feedback challenging.

**Student Engagement:** Classroom discussions may be seen as repetitive by some students, leading to demotivation and disinterest. It's important to find ways to keep students engaged and active in the learning process.

**Mandatory Participation:** Mandatory classroom participation may conflict with students' individual learning styles. Finding ways to encourage participation without making it coercive is necessary. Tool Updates: The constant evolution of DevOps tools requires educators to stay updated, which can be a challenge. Institutions need to provide support and training to educators to help them keep up with the latest trends.

**Program-Level Approach:** Inclusion of DevOps topics requires a program-level approach, not just a course. This calls for collaboration among different departments and curriculum coordination to ensure teaching coherence and effectiveness.

Despite the challenges, the flipped classroom offers significant potential to enhance DevOps education. By overcoming these challenges, we can create more engaging and effective learning environments that prepare students for success in the constantly evolving digital industry.

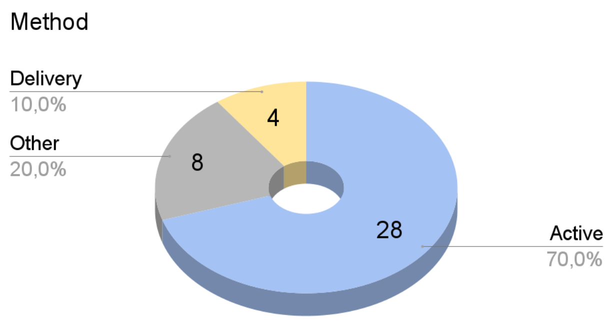

**Figure 6: Active methodology represents 70% of the techniques applied to teach DevOps.**

Finally, Figure 6 indicates that active methodologies are most recommended in the DevOps context, representing 70 percent of the total.

## 5 THREATS OF VIABILITY

This section examines potential threats to the study's validity within the qualitative research framework established by [34]. We considered common validity threats and corresponding control strategies to minimize associated risks.

**Transferability** concerns the generalizability of results to other contexts. We developed a detailed protocol based on the works of [10] and [23], which is rigorously outlined in our methods section. Rigorous search strategies were employed to prevent overlooking relevant papers or inadvertently including irrelevant studies. Three prominent databases known for their extensive collections of scientific publications were consulted. Additionally, snowballing techniques, encompassing both backward and forward searching, were implemented to capture any papers that might not have been identified through the initial search string.

**Credibility:** Credibility refers to the confidence that the research findings accurately reflect the participants' experiences [19]. We had a group of four researchers reviewing the data analyzed by the first researcher during the execution of the study where two of them are Ph.D students. Finally, two doctors supervised and reviewed the structure of the project as a whole.

**Confirmability:** It refers to the degree to which other researchers can verify the findings. We show the evidence for each identified teaching method by quoting participants. We share our data sheets that served as the basis for crafting this paper [17].

## 6 RELATED WORK

Two reviews contribute to our findings, with the first being the research conducted by Fernandes and colleagues delineates the temporal evolution of publications spanning the period from 2014 to 2019 [15], revealing a discernible trend of escalating publication rates. So, the interest in teaching DevOps has been growing since 2014. Grotta and Prado's work, published in 2022, reveals that 72.7% of its database is from 2019 to 2021, supporting the growth of DevOps education. The study also emphasizes PjBL as the primary teaching method, serving as the foundation for restructuring the courses.

In the work by Bruel et al., [9], an encapsulation is provided of the discourse surrounding experience reports on DevOps pedagogy, as discussed during a panel session at the DevOps'2018 conference. The authors synthesized the diverse experiences shared by panel participants into a cohesive paper. While the primary objective of the panel may not have been specifically focused on this aspect, the paper effectively encapsulates numerous challenges and recommendations gleaned from the reported experiences. These insights, delineated in the aforementioned paper [9], significantly informed the outcomes of our systematic review, as it was designated as the principal source for our investigation.

Two literature reviews identified during the data collection process were utilized for this section. One is a compilation of challenges and recommendations in DevOps for education, focusing on computing and software development [15]. This paper addresses 73 challenges and 85 recommendations drawn from 18 sources. The other work pertains to didactic transposition in education through DevOps. The paper's primary outcomes include identifying career development benefits, student grades, and communication enhancements [18]. It highlights the recent emergence of DevOps and its connection to Project-based Learning (PjBL) [30].

Pang et al.[25] conducted a qualitative inquiry aimed at elucidating the landscape of challenges inherent in DevOps pedagogy. Employing a multifaceted approach, the study scrutinized university course curricula and engaged professionals in the field through interviews and questionnaires. It's essential to note that this investigation diverges from conventional analyses typically found in peer-reviewed academic literature. Rather, it focuses on a targeted exploration of DevOps methodologies and techniques employed within educational contexts.

## 7 CONCLUSION

This systematic literature review aimed to identify studies related to software engineering that have utilized educational techniques to teach DevOps within the last five years. It was found that the most cited technique was Project-Based Learning, demonstrating a practical and engaging approach to DevOps learning. Additionally, 70% of the identified techniques were categorized as active methodologies, indicating a growing emphasis on active student participation in the learning process.

These findings underscore the importance of dynamic and practical educational approaches in equipping the next generation with essential skills to address software development challenges professionally. By adopting active and student-centered methods, we can better prepare future IT professionals to navigate increasingly complex and dynamic development environments. Ultimately, investing in DevOps education not only strengthens students' knowledge base but also contributes to advancing the software industry as a whole, fostering innovation and technical excellence.

This systematic review has established a foundational understanding of contemporary trends in DevOps education within software engineering. It highlights three crucial areas for future research that can advance upon these findings. Firstly, there is a need to evaluate learning outcomes associated with prevalent teaching techniques identified in this study. Future research can involve implementing these techniques in controlled environments and assessing their impact on student learning. By measuring technical skills, problem-solving abilities, and teamwork proficiency in DevOps contexts, educators can discern the most effective methods for achieving specific learning objectives.

Moreover, while the review examined broad categories of teaching techniques, future research should delve deeper into contextualization. This entails analyzing how these techniques are adapted for various student groups, program durations, and specific DevOps skill sets. For instance, investigating how project-based learning can be tailored for introductory versus advanced DevOps courses could yield valuable insights for educators seeking to optimize their teaching methodologies.

Furthermore, given the rapid evolution of DevOps with emerging technologies and tools, future research should explore how these advancements influence teaching approaches. This could involve investigating the integration of simulations, cloud-based platforms, or automation tools into DevOps curricula to ensure students are equipped with the latest industry-relevant skills and knowledge. By addressing these areas of inquiry, future research can enrich the understanding of effective DevOps education, providing educators with practical and targeted guidance to prepare software developers for success in an ever-changing industry landscape.

## ACKNOWLEDGMENTS

This research received partial funding from Brazilian research institutions: CAPES, CNPq/FAPEMIG(PROFIX-JD 155774/2023-9), and FAPEMIG (Grants PPM-00651-17 and BPD-00460-22).

## REFERENCES

[1] Ali Alammary, Angela Carbone, and Judy Sheard. 2016. Blended learning in higher education: Delivery methods selection.

[2] Fernando Almeida, Jorge Simões, and Sérgio Lopes. 2022. Exploring the benefits of combining devops and agile. *Future Internet* 14, 2, 63.

[3] Atlassian. 2024. Atlassian Survey 2020 - DevOps Trends. Disponível em: https://www.atlassian.com/whitepapers/devops-survey-2020. Acessado em: 06/05/2024.

[4] Atlassian. 2024. O ciclo de vida de DevOps. Disponível em: https://www.atlassian.com/br/devops. Acessado em: 24/04/2024.

[5] Len Bass, Ingo Weber, and Liming Zhu. 2015. DevOps: A software architect's perspective. Addison-Wesley Professional.

[6] Evgeny Bobrov, Antonio Bucchiarone, Alfredo Capozucca, Nicolas Guelfi, Manuel Mazzara, and Sergey Masyagin. 2020. Teaching DevOps in academia and industry: reflections and vision. In *Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment: Second International Workshop, DEVOPS 2019, Château de Villebrumier, France, May 6–8, 2019, Revised Selected Papers 2*. Springer, 1–14.

[7] Evgeny Bobrov, Antonio Bucchiarone, Alfredo Capozucca, Nicolas Guelfi, Manuel Mazzara, Alexandr Naumchev, and Larisa Safina. 2020. Devops and its philosophy: Education matters! *Microservices: Science and Engineering*, 349–361.

[8] Pierre Bourque. 2020. The SWEBOK Guide—More Than 20 Years down the Road. In *2020 IEEE 32nd Conference on Software Engineering Education and Training (CSEE&T)*. IEEE, 1–2.

[9] Jean-Michel Bruel and Miguel Jiménez. 2019. DevOps' 18 education panel: Teaching feedback and challenges. In *Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment: First International Workshop, DEVOPS 2018, Chateau de Villebrumier, France, March 5-6, 2018, Revised Selected Papers 1*. Springer, 221–226.

[10] Angela Carrera-Rivera, William Ochoa, Felix Larrinaga, and Ganix Lasa. 2022. How-to conduct a systematic literature review: A quick guide for computer science research. *MethodsX* 9, 101895.

[11] Devopedia. 2022. DEVOPEDIA - For developers - By developers. https://devopedia.org/devops [Accessed 2024-01-05.)].

[12] Andrej Dyck, Ralf Penners, and Horst Lichter. 2015. Towards definitions for release engineering and DevOps. In *2015 IEEE/ACM 3rd International Workshop on Release Engineering*. IEEE, 3–3.

[13] Samuel Ferino, Marcelo Fernandes, Elder Cirilo, Lucas Agnez, Bruno Batista, Uirá Kulesza, Eduardo Aranha, and Christoph Treude. 2023. Overcoming Challenges in DevOps Education through Teaching Method. In *2023 IEEE/ACM 45th International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*. IEEE, 166–178.

[14] Eduardo Fernandes, Johnatan Oliveira, Gustavo Vale, Thanis Paiva, and Eduardo Figueiredo. 2010. A Review-based Comparative Study of Bad Smell Detection Tools. (2010).

[15] Marcelo Fernandes, Samuel Ferino, Uirá Kulesza, and Eduardo Aranha. 2020. Challenges and recommendations in devops education: A systematic literature review. In *Proceedings of the XXXIV Brazilian Symposium on Software Engineering*. 648–657.

[16] Nicole Forsgren, Jez Humble, and Gene Kim. 2018. Accelerate: The science of lean software and devops: Building and scaling high performing technology organizations. IT Revolution.

[17] GitHub. 2024. Analise para banco de dados e texto. https://github.com/PedroClair/2024_SBES [Accessed 2024-05-23.)].

[18] Alexandre Grotta and Edmir Parada Vasques Prado. 2022. Devops didactic transposition in is higher education: A systematic literature review.

[19] Egon G Guba. 1981. Criteria for assessing the trustworthiness of naturalistic inquiries. *Ectj* 29, 2, 75–91.

[20] Joonas Hamunen et al. 2016. Challenges in adopting a Devops approach to software development and operations.

[21] Susanna Hartikainen, Heta Rintala, Laura Pylväs, and Petri Nokelainen. 2019. The Concept of Active Learning and the Measurement of Learning Outcomes: A Review of Research in Engineering Higher Education. *Education Sciences* 9, 4. https://doi.org/10.3390/educsci9040276

[22] Gene Kim, Jez Humble, Patrick Debois, John Willis, and Nicole Forsgren. 2021. The DevOps handbook: How to create world-class agility, reliability, & security in technology organizations. IT Revolution.

[23] Barbara Kitchenham, Stuart Charters, et al. 2007. Guidelines for performing systematic literature reviews in software engineering. UK.

[24] Leonardo Leite, Carla Rocha, Fabio Kon, Dejan Milojicic, and Paulo Meirelles. 2019. A survey of DevOps concepts and challenges. *ACM Computing Surveys (CSUR)* 52, 6, 1–35.

[25] Candy Pang, Abram Hindle, and Denilson Barbosa. 2020. Understanding devops education with grounded theory. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering: Software Engineering Education and Training*. 107–118.

[26] Igor Muzetti Pereira, Tiago Carneiro, and Eduardo Figueiredo. 2021. A systematic review on the use of devops in internet of things software systems. In *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. 1569–1571.

[27] Igor Muzetti Pereira, Tiago Garcia de Senna Carneiro, and Eduardo Figueiredo. 2021. Investigating continuous delivery on iot systems. In *Proceedings of the XX Brazilian Symposium on Software Quality*. 1–10.

[28] Juliana Alves Pereira, Kattiana Constantino, and Eduardo Figueiredo. 2014. A systematic literature review of software product line management tools. In *Software Reuse for Dynamic Systems in the Cloud and Beyond: 14th International Conference on Software Reuse, ICSR 2015, Miami, FL, USA, January 4-6, 2015. Proceedings 14*. Springer, 73–89.

[29] Mary Poppendieck and Tom Poppendieck. 2003. Lean software development: An agile toolkit: An agile toolkit. Addison-Wesley.

[30] Maurício Souza, Renata Moreira, and Eduardo Figueiredo. 2019. Students perception on the use of project-based learning in software engineering education. In *Proceedings of the XXXIII Brazilian Symposium on Software Engineering*. 537–546.

[31] Mauricio R de A Souza, Lucas Veado, Renata Teles Moreira, Eduardo Figueiredo, and Heitor Costa. 2018. A systematic mapping study on game-related methods for software engineering education. *Information and software technology* 95 (2018), 201–218.

[32] Marco Tulio Valente. 2020. Engenharia de software moderna. *Princípios e Práticas para Desenvolvimento de Software com Produtividade* 1, 24.

[33] Pornpit Wongthongtham, Elizabeth Chang, Tharam Dillon, and Ian Sommerville. 2008. Development of a software engineering ontology for multisite software development. *IEEE Transactions on knowledge and Data Engineering* 21, 8, 1205–1217.

[34] Xin Zhou, Yuqin Jin, He Zhang, Shanshan Li, and Xin Huang. 2016. A map of threats to validity of systematic literature reviews in software engineering. In *2016 23rd Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 153–160.

Received 10 May 2024; revised 18 June 2024; accepted 5 June 2024