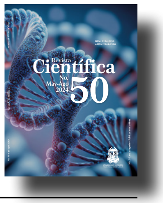
UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Revista Científica

http://revistas.udistrital.edu.co/ojs/index.php/revcie/index



Identification of Characteristics in Very Small Entities to Implement DevOps: Determining Strengths and Opportunities for Improvement

Identificación de características en entidades muy pequeñas para implementar DevOps: Determinación de fortalezas y oportunidades de mejora

Identificando características em entidades muito pequenas para implementar DevOps: Determinando pontos fortes e oportunidades de melhoria

Manuel-Alejandro Pastrana-Pardo[1]
Hugo-Armando Ordoñez-Erazo[2]
Carlos-Alberto Cobos-Lozada[3]

Received: 15th of January of 2024

Accepted: 26th of April of 2024

How to cite: Pastrana-Pardo, M. A., Ordoñez-Erazo, H. A. y Cobos-Lozada, C. A. (2024). Identification of Characteristics in Very Small Entities to Implement DevOps: Determining Strengths and Opportunities for Improvement. *Revista Científica*, *50*(2), 1-17. https://doi.org/10.14483/23448350.22071

Abstract

Using frameworks such as Scrum and DevOps has transformed companies' software development process, increased their quality, facilitated early feedback, and automated repetitive activities through tools to reduce reprocessing, monitor various elements, and enable faster production deployment. However, their implementation requires significant effort, as these frameworks clearly express what can be done but not how. Consequently, discovering which practices, techniques, and tools can be implemented becomes a matter of trial and error, which is very costly for companies of limited size, such as those called *very small entities* (VSEs), consisting of up to 25 employees according to the ISO/IEC 29110 norm. Therefore, this article proposes a set of questions that help determine the current state of companies regarding the implementation of the practices suggested by DevOps, helping to identify which specific aspects can be improved. Additionally, the proposal was evaluated use by seven companies to determine the feasibility of its use by seven companies to determine its feasibility. The application of the questions highlighted the importance of understanding a company's strengths and opportunities for improvement to design a successful implementation guide for the practices mentioned above.

Keywords: characterization of very small entities; DevOps; software development practices; software engineering; software quality assurance.

[1] Ph. D. (c) Institución Universitaria Antonio José Camacho (Cali-Valle del Cauca, Colombia). mapastrana@admon.uniajc.edu.co
[2] Ph. D. Universidad del Cauca (Popayán-Cauca, Colombia). hugoordonez@unicauca.edu.co
[3] Ph. D. Universidad del Cauca (Popayán-Cauca, Colombia). ccobos@unicauca.edu.co

Resumen

El uso de marcos de trabajo como Scrum y DevOps ha transformado el proceso de desarrollo de *software* de las empresas, incrementando su calidad y facilitando la retroalimentación temprana y la automatización de actividades repetitivas a través de herramientas para reducir el reproceso, monitorear diversos elementos y facilitar una puesta en producción más rápida. Sin embargo, su implementación requiere un esfuerzo significativo, pues los marcos de trabajo expresan claramente qué se puede hacer, pero no cómo. Por ende, descubrir qué prácticas, técnicas y herramientas se pueden implementar se convierte en un asunto de prueba y error, lo que resulta muy costoso para empresas de tamaño limitado, como las denominadas *empresas muy pequeñas* (EMP), conformadas hasta por 25 empleados según la norma ISO/IEC29110. Por lo anterior, este artículo propone un conjunto de interrogantes que ayudan a determinar el estado actual de las empresas frente a la implementación de las prácticas sugeridas por DevOps, ayudando a identificar qué aspectos específicos se pueden mejorar. Adicionalmente, la propuesta fue evaluada en siete empresas, en aras de determinar la viabilidad de su uso. La aplicación de los interrogantes evidenció la importancia de comprender las fortalezas y las oportunidades de mejora en una empresa para diseñar una guía de implementación exitosa de las prácticas mencionadas.

Palabras clave: aseguramiento de la calidad del *software*; caracterización de empresas muy pequeñas; DevOps; ingeniería de software; prácticas de desarrollo de software.

Resumo

O uso de frameworks como scrum e DevOps tem transformado o processo de desenvolvimento de software das empresas em uma empresa, aumentando sua qualidade, obtendo feedback antecipado e automatizando atividades repetitivas através de ferramentas para reduzir o retrabalho, monitorar diversos elementos e facilitar a entrega da produção com mais rapidez. No entanto, sua implementação requer um esforço significativo, pois os frameworks expressam claramente o que pode ser implementado, mas não como, deixando para um processo de tentativa e erro descobrir quais práticas, técnicas e ferramentas podem ser implementadas, o que é muito custoso para empresas de porte limitado como as chamadas empresas muito pequenas ou EMPs. composto por até 25 funcionários de acordo com a norma ISO/IEC29110. Portanto, este artigo propõe um conjunto de questões que ajudam a determinar o estado atual das empresas em relação à implementação das práticas sugeridas pelo DevOps, proporcionando a identificação de aspectos específicos que podem ser melhorados. Adicionalmente, a proposta foi avaliada em 7 empresas para determinar sua possibilidade de uso. A aplicação das questões mostra a importância de compreender os pontos fortes e as oportunidades de melhoria que permitem o desenho de um guia para a implementação bem-sucedida das práticas sugeridas.

Palavras-chaves: caracterização de empresas muito pequenas; DevOps; engenharia de software; garantia de qualidade de software; práticas de desenvolvimento de software.

INTRODUCTION

Throughout the evolution of the software development industry, there has been a constant search for the implementation of practices that allow improving the quality of deliverables, reducing production times, and reprocessing, ensuring better profitability of the projects carried out (Martínez-Fernández *et al.*, 2019). Thus, these companies must undertake continuous improvement processes, in some cases assisted by significant investments, through consultants and specialized companies that provide them information on their current state, identifying what they are doing well and what opportunities for improvement they have (Lwakatare *et al.*, 2019). This process is viable if the companies have a financial muscle capable of supporting the transformation, for which their size is critical.

A large part of the industry comprises companies classified as *very small entities* (VSEs) with no more than 25 employees (ISO/IEC JTC 1/SC 7/WG 24, 2019). In response to this, some frameworks such as DevOps have emerged, which, according to Jabbari *et al.* (2016), propose a set of practices to achieve specific goals, thereby enabling mechanisms for continuous improvement and creating an organizational culture where quality permeates the entire development process. However, frameworks allow for determining what can be implemented but do not clarify how to do it, and the exploration process can be costly. This implies the need for guidelines to identify the steps a company has taken to adopt DevOps and to determine which aspects require improvement to transform it gradually (McCarthy *et al.*, 2015).

To implement software development practices that allow the transformation of these companies' landscapes, it is necessary to have a mechanism capable of characterizing their current state by exploring some fundamental elements. This includes development practices, operations, collaboration and communication between project participants, the tests and quality controls in place, monitoring processes, logs of products developed and deployed (either during the warranty period or in the form of products or services), security practices, organizational culture, the tools used, activity automation, and the metrics used to measure product quality and acceptance.

This article proposes a method for characterizing software development companies while providing insights into their status, the progress made in implementing DevOps practices, and potential areas for improvement. The next section outlines the methods used. Afterwards, the results are detailed, and the final section presents a discussion and proposals for future work.

METHODS

This quantitative research used structured questions to collect data (Lethbridge *et al.*, 2005). This was done in the following phases: 1) identifying the topics to be evaluated; 2) organizing the questionnaire in several categories (*i.e.*, development process, operations, collaboration and communication, testing and quality assurance, monitoring and logging, security practices, culture and organization, tooling and automation, and performance and metrics); 3) characterization; 4) results analysis.

Identifying the Topics to be Evaluated

DevOps is a set of practices that combines software development (Dev) and IT operations (Ops) to enhance collaboration, efficiency, and quality throughout the software development lifecycle (Badshah *et al.*, 2020). It focuses on breaking down traditional silos between development and operations teams, enabling a more streamlined and automated approach to software delivery and infrastructure management. It can naturally be integrated with other frameworks like Scrum (Pastrana Pardo *et al.*, 2022). To understand the state of the software development practices implemented by a company, it is necessary to break down into topics according to the DevOps suggestions, not only to organize the information but also to facilitate measuring (Dai *et al.*, 2021; Akbar *et al.*, 2022; Mumbarkar & Prasad, 2022). According to Sousa *et al.* (2019), these topics are:

Development. DevOps recommends evaluating code management practices, collaboration tools, and continuous integration from a development perspective, analyzing automated build processes and code review mechanisms.

Operations. DevOps focuses on understanding a company's infrastructure management and configuration practices, analyzing how the work is integrated, and deploying software. Practices such as continuous integration (CI), continuous deployment (CD), and their combination (continuous delivery) are relevant in deployment automation aimed at improving operations.

Collaboration and Communication. DevOps emphasizes strong collaboration and communication between developers, operations teams, and other stakeholders involved in the software development process. This includes breaking down barriers, fostering a culture of shared responsibility, and promoting effective communication channels.

Testing and Quality Assurance. It is critical to continuous improvement to understand how companies carry out their tests, the quality filters and practices they have implemented, and what they do with the resulting information. Testing helps to understand the current state of products and identify frequent errors in the process to eliminate them.

Monitoring and Logging. DevOps encourages the use of monitoring and logging tools to gain insights into system behavior, identify performance bottlenecks, and detect and troubleshoot issues. Monitoring enables a proactive response to incidents and helps to ensure system stability and reliability.

Security Practices. This extension of DevOps integrates security practices throughout the software development process. It involves incorporating security considerations into every development stage, implementing security automation, and fostering a culture of shared responsibility regarding security.

Culture and Organization. DevOps is more than just practices and tools; it is a mindset and a way of working that must be embedded in the organizational culture. In identifying potential challenges related to silos and resistance to change, it is necessary for the company to understand which practices are ready for implementation and which still need preparation.

Tooling and Automation. DevOps uses configuration management tools to maintain consistent configurations across environments, aligning development, testing, and production. Assessing a tool's suitability and automation levels is crucial for a successful implementation and an analysis of the development lifecycle.

Performance and Metrics. Another DevOps suggestion is to evaluate performance metrics and key performance indicators (KPIs). Identifying which metrics are generated during the software development process, how they help companies improve, how they are reported, and any gaps in monitoring and reporting practices that can help create solid improvement opportunities.

By understanding their strengths and areas for improvement, companies can effectively plan and implement DevOps practices tailored to their unique needs Vadapalli (2018).

Assessing the Development Process

According to Vadapalli (2018) DevOps should evaluate a company's code management practices to understand how much control the development team exerts over the evolution of the software products they develop or support. Table 1 presents a group of questions, their response type, and their suggested grouping to determine the degree of adoption of DevOps practices.

Table 1. *Questions for assessing the development process*

Category	Question
Version control	1. Do development teams use a version control system (*e.g.*, Git) to manage code changes? (Yes/no question) 2. Are branches and pull requests utilized effectively for code review and collaboration? (Yes/no question) 3. Are there established processes for resolving conflicts and merging code changes? (Yes/no question)
Continuous integration (CI)	4. Is a CI server or pipeline in place to automatically build and validate code changes? (Yes/no question) 5. How frequently are code changes integrated into the main branch? (Open question) 6. Are automated builds triggered for every code commit? (Yes/no question)
Code quality and analysis	7. Do you have static code analysis tools integrated into the development process? (Yes/no question) 8. Is code quality monitored regularly (*e.g.*, code coverage, code complexity)? (Yes/no question) 9. Are there established coding standards and guidelines followed by the development teams? (Yes/no question)
Automated testing	10. How extensively are automated tests used within the development process? (Scaled question) 11. Are unit, integration, and end-to-end tests part of the development workflow? (Yes/no question) 12. Can developers access a comprehensive test suite to validate code changes? (Yes/no question)
Feedback and iteration	13. Is there a feedback loop between development teams and stakeholders (*e.g.*, product owners and users)? (Yes/no question) 14. How frequently do you gather feedback on features or enhancements during the development cycle? (Scaled question) 15. Are there mechanisms to incorporate feedback and iterate on the development process? (Yes/no question)

Evaluating Operations

According to Artac *et al*. (2017), for DevOps, fully understanding a company's infrastructure management and configuration practices in different environments (development, testing, and production) is critical to reducing the gap between development and operations. Table 2 presents a set of questions, and their response type are suggested to identify the degree of adoption of these practices.

Table 2. *Questions for evaluating operations*

Category	Question
Infrastructure automation	1. To what extent is infrastructure provisioning automated using tools like Ansible, Terraform, or CloudFormation? (Scale from 1 to 5) 2. Are there standard templates and configurations for different infrastructure components (*e.g.*, servers, databases)? (Yes/no question) 3. Is there a version control system for managing infrastructure code? (Yes/no question)
Deployment and release management	4. How frequently are deployments to production environments performed? (Open question) 5. Is there a well-defined and automated deployment process for applications and services? (Yes/no question) 6. Are there strategies to minimize downtime during deployments (*e.g.*, blue-green deployments, canary releases)? (Yes/no question)
Monitoring and incident management	7. How comprehensive is your monitoring system for tracking the health and performance of your infrastructure and applications? (Scale from 1 to 5) 8. Do you have proactive monitoring to detect and resolve issues before they impact end-users? (Yes/no question) 9. Are there established processes for incident management, including incident response, escalation, and post-incident analysis? (Yes/no question)
Configuration management and secret management	10. Do you have a centralized configuration management system (*e.g.*, Puppet, Chef, or Ansible) to manage server configurations? (Yes/no question) 11. How are sensitive credentials and secrets managed within your infrastructure (*e.g.*, using a secrets management tool)? (Open question) 12. Are policies and controls in place to ensure secure configuration and access management practices? (Yes/no question)
Continuous improvement and feedback loop	13. Is there a culture of continuous improvement and learning within the operations teams? (Yes/no question) 14. Are there mechanisms for collecting feedback from stakeholders, developers, and users to improve the operational processes? (Yes/no question) 15. Do you have regular retrospectives or post-incident reviews to identify areas for improvement and implement changes? (Yes/no question)

Analyzing Collaboration and Communication

As mentioned before, collaborative work allows for a fluid communication channel that guarantees constant interaction between all project stakeholders (Hemon *et al.*, 2020). This collaborative work helps to keep the product vision clear and unified and generates continuous feedback, which in turn helps to keep the project on track and improve various aspects on time and at controlled costs (Hemon *et al.*, 2019). The questions in Table 3 are suggested to understand the degree to which a company has adopted these practices. They are grouped by category and outlined by their response type.

Table 3. *Questions for analyzing collaboration and communication*

Category	Question
Team communication and collaboration	1. How effectively do teams communicate and collaborate across different departments (*e.g.*, development, operations, QA)? (Scale from 1 to 5) 2. Are there established channels and tools for seamless communication and information sharing? (Yes/no question) 3. Are there regular meetings or stand-ups to discuss progress, challenges, and dependencies? (Yes/no question)
Cross-functional collaboration	4. To what extent do teams work together across the entire software development lifecycle (*e.g.*, requirements gathering, development, testing, deployment)? (Scale from 1 to 5) 5. Are there joint planning and decision-making processes involving representatives from different teams? (Yes/no question) 6. Do teams share ownership and responsibility for the success of the software delivery process? (Yes/no question)
Documentation and knowledge sharing	7. Is there a culture of documenting processes, procedures, and best practices within your organization? (Yes/no question) 8. Are there centralized repositories or knowledge bases to store and share documentation? (Yes/no question) 9. Do teams actively contribute to and utilize resources for efficient collaboration and knowledge transfer? (Yes/no question)
Feedback loops and continuous improvement	10. Are there mechanisms for providing feedback on processes and practices related to collaboration and communication? (Yes/no question) 11. How are lessons learned from incidents, retrospectives, and feedback sessions incorporated into process improvements? (Open question) 12. Is there a focus on continuous improvement to enhance collaboration and communication across teams? (Yes/no question)
Automation of collaboration tools and workflows	13. To what extent are collaboration tools and workflows automated to streamline communication and coordination? (Scale from 1 to 5) 14. Are there integrations between different tools used by various teams (*e.g.*, issue tracking, chat, version control)? (Yes/no question) 15. How well are these tools utilized to ensure visibility, transparency, and real-time updates on project status? (Scale from 1 to 5)

Assessing Testing and Quality Assurance Procedures

Quality can be understood as the set of methods, techniques, and tools that make it possible to control various situations that may jeopardize the operation of a product (Pressman & Maxim, 2015). In this vein, several quality filters are required within the development process to ensure both corrective and preventive implementation (Pastrana-Pardo *et al.*, 2022). To identify the degree of adoption of the practices suggested by DevOps in this regard, Table 4 presents a set of questions and their response type.

Table 4. *Testing and quality assurance questions*

Category	Question
Test automation	1. To what extent is test automation implemented in your software development lifecycle? (Scale from 1 to 5) 2. How comprehensive is your automated testing strategy regarding unit, integration, and end-to-end tests? (Scale from 1 to 5) 3. Are frameworks or tools in place for efficient test automation and execution? (Yes/no question)
Continuous testing	4. Do you have a well-defined strategy for continuous testing throughout the development pipeline? (Yes/no question) 5. Are tests executed automatically as part of the CI/CD process? (Yes/no question) 6. Is there a focus on shifting testing left to catch issues earlier in the development cycle? (Yes/no question)
Test environments and data management	7. How effectively are test environments provisioned and managed using infrastructure-as-code practices? (Scale from 1 to 5) 8. Do you have strategies for creating and managing realistic test data sets? (Yes/no question) 9. Are there mechanisms to ensure the integrity and security of test environments and data? (Yes/no question)
Test coverage and metrics	10. How well do you measure and monitor test coverage across different levels of testing (unit, integration, *etc.*)? (Scale from 1 to 5) 11. Are there established metrics to assess the effectiveness of your testing efforts (*e.g.*, code coverage, defect leakage)? (Yes/no question) 12. Do you have visibility into testing progress and test results through dashboards or reporting tools? (Yes/no question)
Collaborative testing and feedback	13. How well do development, QA, and operations teams collaborate during testing? (Scale from 1 to 5) 14. Are there established processes for providing feedback on test results, defects, and test environment issues? (Yes/no question) 15. Do you have mechanisms to capture and address end-user or stakeholder feedback? (Yes/no question)

Analyzing Monitoring and Logging

Monitoring systems enable the early detection of issues such as performance bottlenecks, infrastructure failures, or application errors. Thus, teams can identify problems before they impact end-users, allowing for proactive troubleshooting, mitigations, scaling plans, or even the optimization of relevant aspects for development, testing, or operation (Zhou & Mohindra, 2015). Additionally, when an issue occurs, having access to comprehensive monitoring data and detailed logs helps to understand the root cause, isolate the problem, and expedite the resolution process (Chen, 2019). To understand the adoption of these practices, the following questions are suggested (Table 5).

Table 5. *Questions for monitoring and logging evaluations*

Category	Question
Monitoring tooling and infrastructure	1. What monitoring tools and technologies are in place to track the health and performance of your infrastructure and applications? (Open question) 2. How comprehensive and scalable is your monitoring infrastructure? (Scale from 1 to 5) 3. Are there automated processes to set up monitoring for new components or services? (Yes/no question)
Alerting and incident response	4. Do you have well-defined alerting mechanisms to notify relevant teams about incidents or anomalies? (Yes/no question) 5. Are alerts configured based on meaningful thresholds and actionable metrics? (Yes/no question) 6. How efficiently are incidents responded to and resolved using monitoring data? (Scale from 1 to 5)
Log management and analysis	7. How do you collect, store, and manage logs generated by your applications and infrastructure? (Open question) 8. Are there centralized log management tools or services in place? (Yes/no question) 9. Do you utilize log analysis tools to gain insights into application behavior and troubleshoot issues? (Yes/no question)
Performance monitoring and optimization	10. How effectively do you monitor and analyze application performance metrics (*e.g.*, response time, resource utilization)? (Scale from 1 to 5) 11. Do you have performance baselines or benchmarks to compare against and identify performance bottlenecks? (Yes/no question) 12. Are there established processes for performance optimization based on monitoring data? (Yes/no question)
Dashboards and reporting	13. How well do you visualize and present monitoring data through dashboards or reporting tools? (Scale from 1 to 5) 14. Are there customized dashboards tailored to different stakeholders (*e.g.*, development teams, operations teams, management)? (Yes/no question) 15. Do you regularly review and analyze the data presented in these dashboards to drive improvements? (Yes/no question)

Assessing Security Practices

Security practice assessments help to ensure that security considerations are integrated into the DevOps lifecycle from the early stages. Organizations can build secure applications and infrastructure by embedding security into the development, deployment, and operations processes, reducing security vulnerabilities and risks (Dai *et al.*, 2021). Demonstrating commitment to security through regular assessments and improvements builds trust with customers, clients, and stakeholders, and it shows that the organization takes security seriously, values the protection of sensitive data, and is proactive in addressing security risks (Mohan *et al.*, 2018). Table 6 presents a set of questions grouped by category and response types. These questions aid in determining a company's degree of adoption of these practices.

Table 6. *Questions for assessing security practices*

Category	Question
Secure development practices	1. Are secure coding practices, such as input validation, output encoding, and proper error handling, followed during the software development process? (Yes/no question) 2. Are security requirements and considerations incorporated into the development lifecycle, including threat modeling and security testing? (Yes/no question) 3. Do you have a robust security testing strategy that includes static code analysis, dynamic application scanning, and penetration testing? (Yes/no question)
Security testing and vulnerability management	4. How frequently are security scans and vulnerability assessments performed on your applications and infrastructure? (Scale from 1 to 5) 5. Are there established processes for prioritizing and remediating security vulnerabilities? (Yes/no question) 6. How is access control managed for different systems and resources? (Open question) 7. Is this access control based on the principle of least privilege? (Yes/no question)
Access control and identity management	8. Are multi-factor authentication (MFA) and strong password policies enforced for privileged accounts? (Yes/no question) 9. Do you have centralized identity and access management (IAM) systems for managing user accounts and permissions? (Yes/no question) 10. Do you have a well-defined incident response plan outlining roles, responsibilities, and procedures in case of a security incident? (Yes/no question)
Incident response and security monitoring	11. Are security events and logs actively monitored for potential threats or malicious activities? (Yes/no question) 12. Do you have mechanisms for the timely response, investigation, and mitigation of security incidents? (Yes/no question) 13. Are security controls and practices aligned with industry standards and regulatory requirements applicable to your organization? (Yes/no question)
Compliance and auditing	14. Do you regularly perform security audits and assessments to ensure compliance? (Yes/no question) 15. Are there established processes for tracking and reporting security incidents, breaches, and compliance violations? (Yes/no question)

Culture and Organization Analysis

DevOps stresses the importance of collaboration and communication between different teams and departments, stating that an analysis of the culture and the organization helps to identify barriers or silos that hinder effective collaboration. By fostering a culture of open communication and breaking down organizational barriers, teams can work together seamlessly, improving efficiency and productivity (Maroukian y Gulliver, 2020). Therefore, culture and organization analysis are crucial for successfully implementing and adopting DevOps practices (Sousa *et al.*, 2019). Table 7 presents a set of questions, grouped by category and response type, are proposed to identify a company's current state in this regard.

Table 7. *Culture and organization analysis questions*

Category	Question
Collaboration and communication	1. How effectively do teams collaborate and communicate across different departments and functional areas? (Scale from 1 to 5) 2. Is there a culture of open and transparent communication, encouraging and valuing feedback and ideas? (Yes/no question) 3. Are there established channels and tools for seamless collaboration and knowledge sharing? (Yes/no question)
Cross-functional teamwork	4. To what extent do teams work together across the entire software development lifecycle? (Scale from 1 to 5) 5. Are there shared goals and responsibilities among different teams (*e.g.*, development, operations, QA)? (Yes/no question) 6. Is there a culture of shared ownership and collective accountability for the success of the software delivery process? (Yes/no question)
Continuous learning and improvement	7. Is there a culture of continuous learning and improvement within the organization? (Yes/no question) 8. Are there mechanisms for sharing knowledge, conducting post-mortems, and implementing lessons learned from incidents or failures? (Yes/no question) 9. Are individuals and teams encouraged to experiment, innovate, and learn from successes and failures? (Yes/no question)
Empowerment and autonomy	10. Do teams have the autonomy to make decisions and take ownership of their work? (Yes/no question) 11. Are individuals empowered to drive change, propose process improvements, and implement innovative solutions? (Yes/no question) 12. Is there a culture of trust, wherein individuals are trusted to deliver results and are free to explore new ideas? (Yes/no question)
Organizational alignment and support	13. Is there strong leadership support for DevOps principles and practices? (Yes/no question) 14. Are there defined goals and metrics that align with the principles of DevOps? (Yes/no question) 15. Is there organizational support, such as training programs, resources, and tools, to enable teams to adopt and implement DevOps practices effectively? (Yes/no question)

Assessing Performance and Metrics

By analyzing performance metrics, organizations can make informed decisions about infrastructure, optimization strategies, release management, and resource allocation (Forsgren & Kersten, 2018). This data-driven approach reduces reliance on subjective opinions and empowers teams to make evidence-based decisions. The yes/no questions in Table 8 have been formulated to understand a company's adoption of these practices.

Table 8. *Questions for examining performance and metrics*

Category	Question
Performance monitoring and analysis	1. Do you have a comprehensive performance monitoring strategy that captures relevant metrics for your applications and infrastructure? 2. Are performance metrics collected and analyzed continuously to identify bottlenecks, inefficiencies, and areas for optimization? 3. Do you have mechanisms to track and report performance trends over time and compare them against defined benchmarks or service-level objectives?
Automated performance testing	4. Are performance tests integrated into your CI/CD pipelines to ensure that performance is evaluated consistently throughout the software development lifecycle? 5. Do you have automated performance testing tools and frameworks to simulate realistic user loads and stress test your applications? 6. Are performance test results monitored and analyzed to identify performance regressions and guide optimization efforts?
Application and Infrastructure Scaling	7. Is there a strategy for scaling your applications and infrastructure to handle varying workloads and user demands? 8. Do you have automated mechanisms or policies to scale resources up or down based on predefined thresholds or performance metrics? 9. Are scaling decisions based on real-time performance data and predictive analytics to ensure optimal resource utilization?
Service level agreements (SLAs) and KPIs	10. Have you defined SLAs and KPIs that align with your business goals and customer expectations? 11. Do you have mechanisms to track and measure these SLAs and KPIs and provide stakeholders with regular reports and insights? 12. Are processes in place to analyze SLA/KPI data, identify areas for improvement, and take corrective actions to meet or exceed performance targets?
Incident response and performance optimization	13. Have you established incident response procedures for performance-related issues like slow response times or high error rates? 14. Are incidents thoroughly investigated to identify the root causes and implement performance optimizations? 15. Do you have a feedback loop to share lessons learned from performance incidents and drive continuous improvement in performance monitoring and optimization practices?

RESULTS AND DISCUSSION

Once the seven companies completed the form, they could learn more about adopting DevOps practices. Table 9 analyzes the results regarding the development process.

Table 9. *Survey responses – development process*

Category	Questions	Results
Version control	1, 2, 3	All the companies have implemented version control and use a branch-based structure. Additionally, 85.7% of them have established processes for resolving conflicts and merging code changes.
CI	4, 5, 6	All the companies use CI, but only one does not implement automatic builds for each code commit. Additionally, the frequency at which code changes are integrated is highly variable; in one company, it happens daily; in two, it is weekly; in another one, it is by demand; and the rest do not have a defined frequency.
Code quality and analysis	7, 8, 9	All the companies have established coding standards and guidelines to be followed by development teams, but only 86.5% have static code analysis tools integrated into the development process. Nevertheless, the survey identified that only 71.4% use these tools correctly and regularly monitor code quality.
Automated testing	10, 11, 12	85.7% of the companies include unit, integration, and end-to-end tests in their development workflow. However, only 57.1% give developers access to a suite of tests to validate their code changes. Furthermore, despite its implementation, only 42.9% of companies make significant use of automated tests within their development processes.
Feedback and iteration	13, 14, 15	In 71.4% of the companies, feedback mechanisms between development teams and stakeholders enable the continuous improvement of the process. However, only three companies collect feedback on features or enhancements during development.

Regarding operations, an analysis of the results is presented in Table 10.

Table 10. *Survey responses – operations*

Category	Question	Results
Infrastructure automation	1, 2, 3	Although 85.7% of the respondents have templates and standard configurations for the different components of the infrastructure, automation with tools like Ansible, Terraform, or CloudFormation is very scarce. Of this percentage, only five companies have version control of this infrastructure as code.
Deployment and release management	4, 5, 6	Only 57.1% of the companies have a well-defined, automated deployment process for applications and services. Additionally, 71.4% have no strategies to minimize downtime during deployments. Therefore, the implementations in production environments are variables not defined by a work plan.
Monitoring and incident management	7, 8, 9	In 57.1% of the companies, the monitoring system to track the health and performance of the infrastructure and applications is good, albeit with opportunities for improvement. In the remaining percentage, there is a significant opportunity for improvement, reflected in low ratings. Furthermore, 57.1% of the respondents indicate a lack of proactive monitoring to detect and resolve issues before end-users are affected. Nevertheless, 85.7% mentioned there are processes in place for incident management, including incident response and escalation, as well as post-incident analysis.
Configuration management and secret management	10, 11, 12	As for managing confidential credentials, the survey indicates that 85.7% of the companies use some type of profile manager, such as Active Directory, Vault, or Azure Key Value. Only one company suggests that it does not manage credentials. Additionally, 71.4% indicate a continuous improvement and learning culture within the operations teams.
Continuous improvement and feedback loop	13,14,15	Although 71.4% of the respondents indicate that there are mechanisms to collect feedback from stakeholders, only 57.1% have regular retrospectives or post-incident reviews to identify areas for improvement and implement changes.

The results for collaboration and communication are analyzed in Table 11.

Table 11. *Survey responses – collaboration and communication*

Category	Question	Results
Team communication and collaboration	1, 2, 3	85.7% of the respondents have established channels and tools for fluid communication and information exchange, in addition to holding regular meetings or stand-ups to discuss progress, challenges, and dependencies. However, only four companies state that communication is highly effective; the rest do not.
Cross-functional collaboration	4, 5, 6	All companies state that they have joint planning and decision-making processes involving representatives from different teams and that they share ownership and responsibility for the success of the software delivery process.
Documentation and knowledge sharing	7, 8, 9	All the surveyed companies state that their organization has a culture of documenting processes, procedures, best practices, and centralized repositories or knowledge bases to store and share documentation. However, only one company states that this has not been useful. In others, it has been highly effective for knowledge management.
Feedback loops and CI	10, 11, 12	Even though all companies state that they have an approach oriented towards CI, one company does not have mechanisms to provide feedback on processes and practices related to collaboration and communication. In 85.7% of the cases, these companies identified that the teams actively contribute and use resources for efficient collaboration and knowledge transfer. The most widely used practice to share information is meetings.
Automation of collaboration tools and workflows	13, 14, 15	All companies mention integrating the different tools used by various teams. 28.6% of the respondents use these tools effectively to guarantee visibility, transparency, and real-time project status updates. 42.9% use them well but could improve. For the rest, this issue requires strengthening.

Test automation is a critical factor for DevOps implementation in testing and quality assurance. Questions 1, 2, and 3 aim to analyze this aspect. In this regard, only 43% of the respondents acceptably implement test automation in their software development lifecycle. However, they identified that they still have some work to do. The other respondents are taking their first steps and, therefore, have yet to implement a great deal. The rest of the results are presented below in Table 12.

Table 12. *Survey responses – testing and quality assurance*

Category	Question	Results
Continuous testing	4, 5, 6	57.1% of the respondents do not have a well-defined strategy for continuous testing across their pipelines. Therefore, they do not run the tests automatically as part of the CI/CD process, but they are willing to do so.
Test environments and data management	7, 8, 9	Test environments are provisioned and managed inefficiently using infrastructure-as-code practices. However, 71.4% of the respondents have strategies to create and manage realistic test datasets. Additionally, it should be noted that there are few mechanisms to guarantee the integrity and security of test environments and data.
Test coverage and metrics	10, 11, 12	57.1% of the companies have no established metrics to assess the effectiveness of their testing efforts. Moreover, 71.4% do not have visibility into test progress and test results through dashboards or reporting tools. These companies are not measuring and monitoring test coverage well and at different levels.
Collaborative testing and feedback	13, 14, 15	85.7% of the respondents indicate that processes are in place to provide feedback on test results, defects, and test environment issues from all participants. In addition, 57.1% consider this an essential strategy.

In the case of monitoring and logging, the tools used are related to the cloud platform, which manages the infrastructure. Only in one case are the tools yet to be implemented. However, the survey shows that 71.4% of the companies have no automated processes to configure the monitoring of new components or services. These are the results of questions 1, 2, and 3. The rest of the results are shown in Table 13.

Table 13. *Survey responses – monitoring and logging*

Category	Question	Results
Alerting and incident response	4, 5, 6	57.1% of the respondents have well-defined alert mechanisms to notify relevant teams about incidents or anomalies. These mechanisms trigger alerts depending on specific thresholds. The other companies still need to implement something similar. Despite the above, only 28.6% have efficiently adopted this practice to respond to incidents using monitoring data.
Log management and analysis	7, 8, 9	In most cases, the companies leave the collection, storage, and management of the records generated by the applications and infrastructure to the cloud they use (AWS, Azure, or others). Therefore, in 71.4% of the cases, there are no centralized record management tools or services.
Performance monitoring and optimization	10, 11, 12	In 71.4% of cases, there are no established processes for performance optimization based on monitoring data. Thus, the effectiveness of monitoring and analyzing application performance metrics is very low.
Dashboards and reporting	13, 14, 15	71.4% of the companies indicate that they do not have personalized panels or ones adapted to the needs of different interested stakeholders, although they do have general panels to display this information. Additionally, 57.1% regularly review and analyze the data presented in said panels to drive improvements.

Regarding security practices, the results are analyzed in Table 14.

Table 14. *Survey responses – security practices*

Category	Question	Results
Secure development practices	1, 2, 3	71.4% of all respondents incorporate security requirements and considerations into the development lifecycle, including threat modeling and security testing. Additionally, 85.7% use secure coding practices, but only 57.1% have a solid security testing strategy, including static code analysis, dynamic application analysis, and penetration testing.
Security testing and vulnerability management	4, 5, 6	In 71.4% of the companies, there are established processes to prioritize and remedy security vulnerabilities. These processes mostly employ the principle of least privilege and Azure active directory permission managers or IAM from the Google Cloud Platform. However, the vulnerability scanning frequency is low.
Access control and identity management	7, 8, 9	85.7% of the respondents apply MFA policies, secure passwords for privileged accounts, and have accounts with centralized IAM systems. However, 42.9% need a well-defined incident response plan that describes roles, responsibilities, and procedures during a security incident.
Incident response and security monitoring	10, 11, 12	Although 71.4% of the companies state that their security controls and practices align with industry standards, only 51.7% have mechanisms for timely response and investigating and mitigating security incidents.
Compliance and auditing	13, 14	71.4% of the respondents state that they have processes in place to track and report security incidents, breaches, and compliance violations, in addition to regularly conducting security audits and assessments to ensure compliance.

As for culture and organization, according to the results of questions 1, 2, and 3 in relation to the collaboration and communication category, all companies have a culture of open and transparent communication, where comments and ideas are encouraged and valued based on the implemented tools. 42.9% of the respondents consider collaboration and communication between teams from different departments and functional areas to be highly effective, followed by 28.6% who think it is practical. The rest consider it moderately effective.

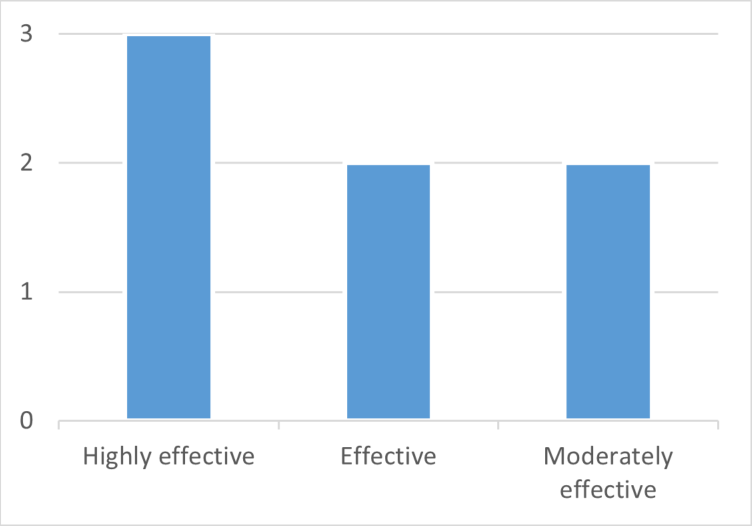


Figure 1. *Results regarding collaboration and communication*

The rest of the results are presented in Table 15.

Table 15. *Survey responses – culture and organization*

Category	Question	Results
Cross-functional teamwork	4, 5, 6	Although all respondents report shared goals and responsibilities among different teams, only 85.7% have a culture of shared ownership and collective responsibility for the success of the software delivery process. The teams rarely work together during the development cycle.
Continuous learning and improvement	7, 8, 9	85.7% of the companies have a culture of learning and CI. However, only 71.4% have mechanisms to share knowledge, root cause analyses, and implement lessons learned from incidents or failures. Notably, all companies' people and teams are encouraged to experiment, innovate, and learn from successes and failures.
Empowerment and autonomy	10, 11, 12	In all companies, the teams can make decisions, take ownership of their work, promote change, propose process improvements, and implement innovative solutions. However, one company indicates no culture of trust, *i.e.*, people are not trusted to get results and are not fully free to explore new ideas.
Organizational alignment and support	13, 14, 15	Although there is strong leadership support for DevOps in 85.7% of the cases, 28.6% of the companies still lack well-defined objectives and metrics that align with DevOps principles, and 42.9% lack real organizational support.

The results for tooling and automation are presented in Table 16.

Table 16. *Survey responses – tooling and automation*

Category	Question	Results
Tooling selection and integration	1, 2, 3	The companies have identified and implemented appropriate tools for the DevOps lifecycle stages, but there are opportunities for improvement. Integration is lacking, and a process for selecting new tools based on requirements and industry best practices needs to be established.
Infrastructure as code and configuration management	4, 5, 6	Infrastructure as code and its management through version control and automated provisioning have gained significant ground. In five of the respondents, it is a current practice. However, only 57.1% use configuration management tools to automate the deployment and configuration of infrastructure components.
CI/CD	7, 8, 9	85.7% of the respondents have established CI/CD pipelines to automate building, testing, and deployment with minimal human intervention.
Monitoring and alerting automation	10, 11, 12	57.1% of the companies do not have monitoring and alerting integrated into the CI/CD pipeline for automated health checks and the validation of deployed applications. However, automatic alerts and notifications are configured for critical events or performance anomalies.
Incident response and remediation automation	13, 14, 15	In 85.7% of the companies, incident response workflows are automated to facilitate rapid incident detection, response, and resolution. In addition, post-incident actions such as root cause analysis or configuration updates are automated to prevent similar incidents in the future. However, 28.6% of the respondents have no automation for reversing or repairing failed implementations or infrastructure issues.

Finally, the results regarding performance and metrics are provided in Table 17.

Table 17. *Survey responses – performance and metrics*

Category	Question	Results
Performance monitoring and analysis	1, 2, 3	71.4% of the respondents do not have mechanisms to track and report performance trends over time and compare them against defined benchmarks or service level objectives. Likewise, 57.1% need a comprehensive performance monitoring strategy that captures relevant metrics for their applications and infrastructure, where performance metrics are continuously collected and analyzed to identify bottlenecks, inefficiencies, and areas for optimization.
Automated performance testing	4, 5, 6	57.1% of the respondents do not have performance testing integrated into their CI/CD pipelines or have automated performance testing frameworks and tools to simulate realistic user loads and stress-test their applications.
Application and infrastructure scaling	7, 8, 9	71.4% of the companies do not have policies or automated mechanisms to increase or decrease resources based on predefined thresholds or performance metrics. However, 57.1% indicate that there is a strategy in place to scale their applications and infrastructure to handle different workloads and user demands.
SLAs and KPIs	10, 11, 12	57.1% of all the companies have defined SLAs and KPIs aligned with their business objectives and customer expectations, in addition to mechanisms for generating reports on their measurement. Additionally, processes are in place to analyze SLA/KPI data, identify areas for improvement, and take corrective action to meet or exceed performance targets.
Incident response and performance optimization	13, 14, 15	Although 85.7% of the respondents thoroughly investigate incidents to identify root causes and implement performance optimizations, only 57.1% have established incident response procedures for performance-related issues. Additionally, 71.4% have a feedback loop to share lessons learned from performance incidents and drive continuous improvement in performance optimization and monitoring practices.

Based on the above, the leading practices that companies adopt are versioning and CI. These two are the keys to chaining other practices such as unit testing, static code analysis, security measurement, and constant monitoring within an automated process. Likewise, most companies clearly state that code standardization helps increase maintainability. However, this also highlights that testing continues to represent a challenge for companies, especially unitary ones, due to the rising culture of trying to test as soon as possible to increase quality, as well as the implementation of early warnings that encourage continuous improvement.

Another challenge for DevOps teams in their attempt to bring the development and operations areas closer together is implementing practices related to configuration management and infrastructure as code. Automated deployment processes appear to be a desired aspect that the surveyed companies have not been able to implement successfully. Additionally, security seems to be a recurring opportunity for improvement, and it should be included in the development and operation of software from the early stages.

It is essential to stress the evident impact of DevOps on the creation of transparent, clear, and continuous communication channels, promoting collaboration between all members, the correct management of the documentation associated with projects, the creation of standard and known guidelines for all project members, and the generation of information that enables continuous improvement.

CONCLUSIONS

The research team obtained feedback from various companies in the industry, which allowed them to determine the effectiveness of the questions.

One aspect that stands out is the advantages of version control and its use in the industry. Although not all companies have adopted this practice to the fullest, most use a branch-based model that allows for organized work and adopting other DevOps suggestions. Likewise, CI plays a significant role in implementing automated preventive quality cycles, which companies have gradually discovered and implemented. However, said companies need to be aware of CI's full potential, as it is the gateway to integrating other practices often implemented separately, *e.g.*, CD, unit tests, and static code analysis. Moreover, although elements such as unit tests and automated functional tests have been integrated, they have not been widely incorporated into an automated cycle. Nevertheless, they are executed at specific times, as reflected in the survey's results.

On the other hand, infrastructure as code begins to spark companies' interest due to its advantages regarding the maintainability of various environments, the possibility of recovering from failures, ease of management, and the metrics that can be handled. Finally, security is an aspect that requires greater attention from the surveyed entities.

The research team hopes to generate a second version of the characterization and refine it with the feedback received. The team hopes that the second version will be the starting point for defining a DevOps implementation guide for software development practices for very small companies.

ACKNOWLEDGMENTS

The authors would like to express their gratitude to Universidad del Cauca, Popaýan, Colombia for participating in the research project *Implementación de prácticas armonizadas para el desarrollo de* *software basado en SCRUM y DevOps en empresas muy pequeñas* [Implementation of harmonized practices in software development based on SCRUM and DevOps in very small entities]. Likewise, the authors are grateful to Institución Universitaria Antonio Jose Camacho, Cali, Colombia, for supporting the researchers in executing this project.

REFERENCES

Akbar, M. A., Rafi, S., Alsanad, A. A., Qadri, S. F., Alsanad, A., Alothaim, A. (2022). Toward Successful DevOps: A Decision-Making Framework. *IEEE Access*, *10*, 51343-51362. https://doi.org/10.1109/ACCESS.2022.3174094

Artac, M., Borovssak, T., Di Nitto, E., Guerriero, M., Tamburri, D. A. (2017). DevOps: Introducing infrastructure-as-code. In: *IEEE/ACM 39th International Conference on Software Engineering Companion*, Buenos Aires, Argentina. https://doi.org/10.1109/ICSE-C.2017.162

Badshah, S., Khan, A. A., Khan, B. (2020). Towards Process Improvement in DevOps. In: *Proceedings of the Evaluation and Assessment in Software Engineering* (pp. 427-433). ACM. https://doi.org/10.1145/3383219.3383280

Chen, B. (2019). Improving the software logging practices in DevOps. In: *IEEE/ACM 41st International Conference on Software Engineering: Companion 2019*, Montreal, QC, Canada. https://doi.org/10.1109/ICSE-Companion.2019.00080

Dai, Q.-M., Mao, R.-F., Huang, H., Rong, G.-P., Shen, H.-F., Shao, D. (2021). DevSecOps: Exploring practices of realizing continuous security in DevOps. *Ruan Jian Xue Bao/Journal of Software*, *32*(10), 3014-3035. https://doi.org/10.13328/j.cnki.jos.006276

Forsgren, N., Kersten, M. (2018). DevOps metrics. *Communications of the ACM*, *61*(4), 44-48. https://doi.org/10.1145/3159169

Hemon, A., Lyonnet, B., Rowe, F., Fitzgerald, B. (2019). Conceptualizing the transition from agile to DevOps: A maturity model for a smarter is function. *IFIP Advances in Information and Communication Technology*, *533*, 209-223. https://doi.org/10.1007/978-3-030-04315-5_15

Hemon, A., Lyonnet, B., Rowe, F., Fitzgerald, B. (2020). From Agile to DevOps: Smart skills and collaborations. *Information Systems Frontiers*, *22*(4), 927-945. https://doi.org/10.1007/s10796-019-09905-1

ISO/IEC JTC 1/SC 7/WG 24 (2019). *ISO/IEC DTR 29110-5-6-3:2018 (ISO/IEC DTR 29110-5-6-3:2018)*. ISO. https://isotc.iso.org/livelink/livelink/open/jtc1sc7wg24

Jabbari, R., bin Ali, N., Petersen, K., Tanveer, B. (2016). What is DevOps? A systematic mapping study on definitions and practices. In: *Proceedings of the Scientific Workshop Proceedings of XP2016*. https://doi.org/10.1145/2962695.2962707

Lethbridge, T. C., Sim, S. E., Singer, J. (2005). Studying software engineers: Data collection techniques for software field studies. *Empirical Software Engineering*, *10*(3), 311-341. https://doi.org/10.1007/S10664-005-1290-X/METRICS

Lwakatare, L. E., Kilamo, T., Karvonen, T., Sauvola, T., Heikkilä, V., Itkonen, J., Kuvaja, P., Mikkonen, T., Oivo, M., Lassenius, C. (2019). DevOps in practice: A multiple case study of five companies. *Information and Software Technology*, *114*, 217-230. https://doi.org/10.1016/j.infsof.2019.06.010

Maroukian, K., Gulliver, S. R. (2020). *Leading DevOps practice and principle adoption*. arXiv. https://doi.org/10.5121/csit.2020.100504

Martinez-Fernandez, S., Vollmer, A. M., Jedlitschka, A., Franch, X., Lopez, L., Ram, P., Rodriguez, P., Aaramaa, S., Bagnato, A., Choras, M., Partanen, J. (2019). Continuously assessing and improving software quality with software analytics tools: A case study. *IEEE Access*, *7*, 68219-68239. https://doi.org/10.1109/ACCESS.2019.2917403

McCarthy, M. A., Herger, L. M., Khan, S. M., Belgodere, B. M. (2015). Composable DevOps: Automated ontology based DevOps maturity analysis. In: *Proceedings - 2015 IEEE International Conference on Services Computing*, New York, NY, USA. https://doi.org/10.1109/SCC.2015.87

Mohan, V., ben Othmane, L., Kres, A. (2018). BP: Security concerns and best practices for automation of software deployment processes: An industrial case study. In: *IEEE Cybersecurity Development (SecDev)*, Cambridge, MA, USA. https://doi.org/10.1109/SecDev.2018.00011

Mumbarkar, P., Prasad, S. (2022). Adopting DevOps: Capabilities, practices, and challenges faced by organizations. *AIP Conference Proceedings*, *2519,* e030029. https://doi.org/10.1063/5.0110594

Muñoz, M., Mejía, J., Lagunas, A. (2018). Implementation of the ISO/IEC 29110 standard in agile environments: A systematic literature review. In: *Iberian Conference on Information Systems and Technologies*, Cáceres, Spain. https://doi.org/10.23919/CISTI.2018.8399332

Pastrana Pardo, M. A., Ordóñez Erazo, H. A., Cobos Lozada, C. A. (2022). Approach to the best practices of software development based on DevOps and SCRUM used in very small entities. *Revista Facultad de Ingeniería*, *31*(61), e14828. https://doi.org/10.19053/01211129.V31.N61.2022.14828

Pastrana-Pardo, M.-A., Ordóñez-Erazo, H.-A., Cobos-Lozada, C.-A. (2022). Process model represented in BPMN for guiding the implementation of software development practices in very small companies harmonizing DEVOPS and SCRUM. *Revista Facultad de Ingeniería*, *31*(62), e15207. https://doi.org/10.19053/01211129.V31.N62.2022.15207

Pressman, R. S., Maxim, B. R. (2015). *Software engineering: A practitioner's approach (8th ed.)*. McGraw-Hill.

Sousa, L., Trigo, A., Varajão, J. (2019). *DevOps-fundamentos e perspetivas*. https://aisel.aisnet.org/capsi2019/8

Vadapalli, S. (2018). *DevOps: Continuous delivery, integration, and deployment with DevOps: Dive into the core DevOps strategies*. https://books.google.com.mx/books?id=N5RRDwAAQBAJ

Zhou, N., Mohindra, A. (2015). Causality-driven performance monitoring and scaling automation for managed solutions. In: *IEEE International Conference on Services Computing*, New York, NY, USA. https://doi.org/10.1109/SCC.2015.70