**Вісник Тернопільського національного технічного університету**
**https://doi.org/10.33108/visnyk_tntu**
Scientific Journal of the Ternopil National Technical University
*2024, № 1 (117) https://doi.org/10.33108/visnyk_tntu2025.01*
*ISSN 2522-4433. Web: **visnyk.tntu.edu.ua***

**UDC 004.75:004.42**

# THE IMPACT OF MODERN CLOUD TECHNOLOGIES ON THE EFFICIENCY OF DEVOPS PROCESSES

**Mykhailo Luchkevych[1]; Iryna Shakleina[1]; Oleksii Duda[2]**

*[1]Lviv Polytechnic National University, Lviv, Ukraine*
*[2]Ternopil Ivan Puluj National Technical University, Ternopil, Ukraine*

***Abstract.*** *The article investigates the impact of modern cloud technologies on the efficiency of DevOps processes, which are key to the automation, flexibility and reliability of software development. With the rapid development of information technology, cloud platforms play an important role in accelerating software product development, testing and deployment. Implementing DevOps in the cloud environment reduces the time to market, increases the stability of software systems and optimizes the use of computing resources, which is critical for modern companies. This paper discusses DevOps teams' main challenges, including the need to scale infrastructure rapidly, ensure continuous integration and delivery (CI/CD), automate testing, monitor performance, and optimize costs. Particular attention is paid to security problems, configuration management and the human factor minimization when implementing program code changes. A comparative analysis of the three leading cloud platforms - Amazon Web Services, Google Cloud Platform, and Microsoft Azure – is carried out in the context of their impact on DevOps processes. The possibilities of automation, support for container technologies, infrastructure scalability, monitoring tools, and integration with CI/CD pipelines are evaluated. AWS was found to offer the broadest set of DevOps tools with a high level of automation, making it an attractive choice for organizations focused on complex enterprise solutions. Google Cloud Platform demonstrates the best support for Kubernetes and containerization, an essential factor for teams working with microservices architecture. Microsoft Azure provides the most profound integration with the Microsoft ecosystem and is the best choice for companies that use Windows and related products. An experimental study has shown that the choice of a cloud platform significantly impacts the speed of release cycles, the stability of the deployed infrastructure, the ability to respond quickly to changes in load, and the productivity of DevOps teams in general. Recommendations for choosing the optimal cloud platform are proposed depending on the specifics of the project, the scale of the organization, the level of system load, and automation requirements.*

***Key words:*** *DevOps, cloud technologies, AWS, GCP, Azure, CI/CD, automation, containerization, scalability, monitoring.*

*https://doi.org/10.33108/visnyk_tntu2025.01.112* *Received 20.02.2025*

## 1. INTRODUCTION

In today's digital environment, competition between companies requires constant innovation, data growth, and software complexity, which create new challenges for developers and software system operators. Traditional software development and maintenance methods are becoming insufficiently effective due to the need for rapid releases, system reliability, and continuous availability. DevOps, as an integrative methodology, combines development and operations processes with an emphasis on automation, team collaboration, and constant improvement. DevOps methodology allows organizations to adapt to market changes faster, reduce risks during releases, and improve the quality of the final product.

On the other hand, cloud computing has become the backbone of modern IT infrastructure. They offer dynamic scalability, high availability, and a wide range of automation and resource management tools. Leading cloud platforms such as Amazon Web Services (AWS), Google Cloud Platform (GCP), and Microsoft Azure have become integral to many DevOps processes. Their services cover everything from integration with version control systems to the full automated software delivery cycle. What's particularly noteworthy is that

 *Corresponding author: Oleksii Duda; e-mail: oleksij.duda@gmail.com*

each of these platforms offers unique solutions tailored to the needs of organizations of different sizes: AWS is distinguished by the most significant number of services and customization flexibility for complex projects; GCP focuses on innovation and efficient container management; Azure offers the best solutions for enterprises using the Microsoft ecosystem.

Despite the obvious advantages of cloud computing, its implementation is often accompanied by certain challenges. These include the complexity of the initial setup, the need for staff training, and cost optimization. Therefore, this article aims to provide an in-depth analysis of how cloud technologies affect the efficiency of DevOps processes, particularly in terms of automation, scalability, reliability, and continuous integration, which are critical factors for IT companies. The study also aims to evaluate the benefits and limitations of using AWS, GCP, and Azure in various scenarios, including their impact on release speed, team productivity, and risk mitigation in the operation of software systems.

## 2. PROBLEM STATEMENT

DevOps, as an integrative approach to developing and operating software systems, aims to ensure fast, high-quality and stable releases. Still, its implementation is accompanied by a number of technical and organizational challenges. One of the key challenges is infrastructure requirements, as traditional approaches to infrastructure management do not provide the necessary flexibility and speed. For DevOps, it is critical to automate the creation, configuration, and scaling of environments, which is difficult to achieve without specialized tools. Another important factor is the need for automation, which is the foundation of DevOps. Automating routine tasks, such as testing, integration, and delivery (CI/CD), helps reduce the likelihood of errors and significantly speeds up processes. At the same time, ensuring the continuity of integration and delivery is critical in today's competitive environment. Reliable tools and platforms are needed to implement these processes. In addition, the DevOps approach involves working in conditions of constant change, such as changing requirements, increasing workload, or introducing new technologies. This creates a need for an adaptive, flexible and scalable infrastructure that can effectively respond to the challenges of a dynamic environment.

Cloud technologies provide a wide range of solutions to overcome these challenges. AWS, GCP, and Azure offer tools that automate software deployment, integration, testing, and scaling. For example, AWS provides feature-rich services such as CodePipeline, CodeDeploy, and CloudFormation for CI/CD management; GCP offers container-oriented solutions such as Kubernetes (via Google Kubernetes Engine (GKE)) to simplify the management of scalable applications; Azure integrates DevOps tools such as Azure Pipelines to manage the application lifecycle from planning to release. However, the effectiveness of using these platforms depends on several factors, including the level of staff skills, project specifics, budget constraints, and the features of the chosen cloud environment.

Thus, the main problem comes down to the need to determine the impact of modern cloud platforms on the efficiency of DevOps processes. Questions remain as to how AWS, GCP, and Azure services affect key DevOps metrics, including infrastructure setup time, release speed, automation level, and scalability. It is also important to consider the existing limitations and risks associated with using cloud platforms in the context of DevOps to identify potential challenges and ways to overcome them. Answers to these questions will not only help to assess the role of cloud technologies in DevOps transformation but also to develop recommendations for their optimal use in different organizational contexts.

## 3. ANALYSIS OF LITERATURE SOURCES

An analysis of modern scientific sources and practical reports shows a significant role of cloud technologies in improving DevOps processes [1]. Studies show that cloud platforms

provide tools that simplify automation, scalability, and monitoring, but the effectiveness of these solutions largely depends on the specific conditions of use [2]. According to research [3], Amazon Web Services (AWS) offers a wide range of services to automate, integrate, and optimize DevOps processes, making the platform a popular choice for organizations seeking to implement DevOps at a high level. Among the key AWS services, a special place is occupied by CodePipeline, which provides automated integration and delivery, creating a continuous flow of building, testing, and deploying code [4]. The tool supports integration with third-party services such as GitHub and Bitbucket and with AWS's own storage. CodeBuild aims to reduce delays in testing and building code by using flexible computing resources [5]. This allows you to work effectively even with large projects, speeding up the release cycle. CloudFormation facilitates infrastructure management through the Infrastructure as Code (IaC) approach, which allows you to create templates for managing AWS resources [6]. This greatly simplifies infrastructure configuration, scaling, and recovery. Elastic Beanstalk provides rapid deployment of web applications without the need for in-depth resource management [7]. Developers upload the code, and the service automatically creates the infrastructure and provides load balancing, scaling, and monitoring.

Amazon EKS (Elastic Kubernetes Service) offers a managed Kubernetes environment for deploying, managing, and scaling containerized applications, which simplifies container orchestration in a DevOps environment [8]. AWS integrates with popular DevOps tools such as Jenkins, GitLab, and Docker, which extends the platform's capabilities [9].

Google Cloud Platform is one of the leading cloud platforms that demonstrate high efficiency in working with containerized environments, which is especially important for modern DevOps processes [10]. Research [11] confirms that the key advantage of GCP is Google Kubernetes Engine, a managed service for container orchestration that provides application scalability, management automation, and stability even in complex environments. This makes GKE the number-one choice for projects that require flexibility and high performance when working with containers. One of the most important GCP services is Cloud Build, which allows you to create efficient CI/CD pipelines [12]. The service supports integration with many code sources, including GitHub and GitLab, and provides high-speed tasks, including building, testing, and deployment [13]. Cloud Build provides the ability to execute processes in parallel, which significantly reduces the release cycle time. Another strong point of GCP is the Operations Suite (formerly known as Stackdriver), which offers advanced monitoring, logging, and diagnostics capabilities [14]. With this tool, developers can identify and resolve problems at an early stage, which increases application stability and minimizes the risk of failures.

GCP also integrates with popular DevOps tools such as Terraform [15], Jenkins, and GitLab CI/CD, allowing developers to create scalable solutions using well-known frameworks. Thanks to its powerful infrastructure, GCP is suitable for complex projects that require adaptability, stability, and rapid deployment of innovations.

Microsoft Azure is a powerful cloud platform that offers integrated solutions for implementing and optimizing DevOps processes within its ecosystem [16]. The main advantage of Azure is its tight integration with other Microsoft products, which provides convenience and efficiency for organizations that already use Windows Server, Active Directory, or Office 365. Azure DevOps is a comprehensive set of tools for managing the software lifecycle. One of the key components is Azure Pipelines [17], which supports multiplatform and allows you to run CI/CD processes for various types of applications, including traditional web applications, mobile applications, and containerized services. Thanks to integration with GitHub [18] and other repositories, developers can create fast and reliable pipelines to automate testing and release processes. Azure Repos provides an efficient version control system that supports both Git and TFVC (Team Foundation Version Control) [19]. It enables teams to centrally store,

version, and manage code with deep integration with tools such as Visual Studio and Visual Studio Code. Microsoft offers Azure Monitor to monitor application performance in real-time, which provides advanced capabilities for monitoring system health, analysing logs, and identifying potential problems [20]. Integration with Application Insights [21] allows you to get analytics on user interaction with the application and track key metrics, ensuring a quick response to possible failures. Azure also supports a wide range of DevOps tools, such as Terraform, Ansible, and Jenkins, which allows you to adapt the platform to the needs of different projects flexibly. In addition, using Azure Kubernetes Service simplifies the work with containerized applications by providing automated management of Kubernetes clusters [22].

Papers [23–25] compare and consider the use of cloud technologies to optimize scientific data processing tasks that have become an integral part of modern organizations. The authors emphasize the importance of automation, scalability, and security in the development and deployment of machine learning models. The growth of data volumes and model complexity puts organizations in a choice between available cloud platforms (AWS, GCP, Azure).

Thus, the research analysis in this area indicates the need to study the impact of cloud platforms on DevOps processes, as the rapid development of technology and growing business needs require a deep understanding of the capabilities of each platform to make informed decisions about their implementation.

## 4. MATERIALS AND METHODS

The study was based on analysing CI/CD tools offered by three leading cloud platforms – Amazon Web Services, Google Cloud Platform, and Microsoft Azure. Each of these platforms was tested to assess its impact on key aspects of DevOps processes. Implementing DevOps in a cloud environment requires a comprehensive approach that includes analysing project needs, selecting the right platform, setting up automation tools, deploying applications, and evaluating performance. Each of these steps plays a vital role in ensuring continuity of integration and delivery, optimizing infrastructure, and achieving high performance. Figure 1 shows the step-by-step structure of this process, considering the specifics of using AWS, GCP, and Azure services.

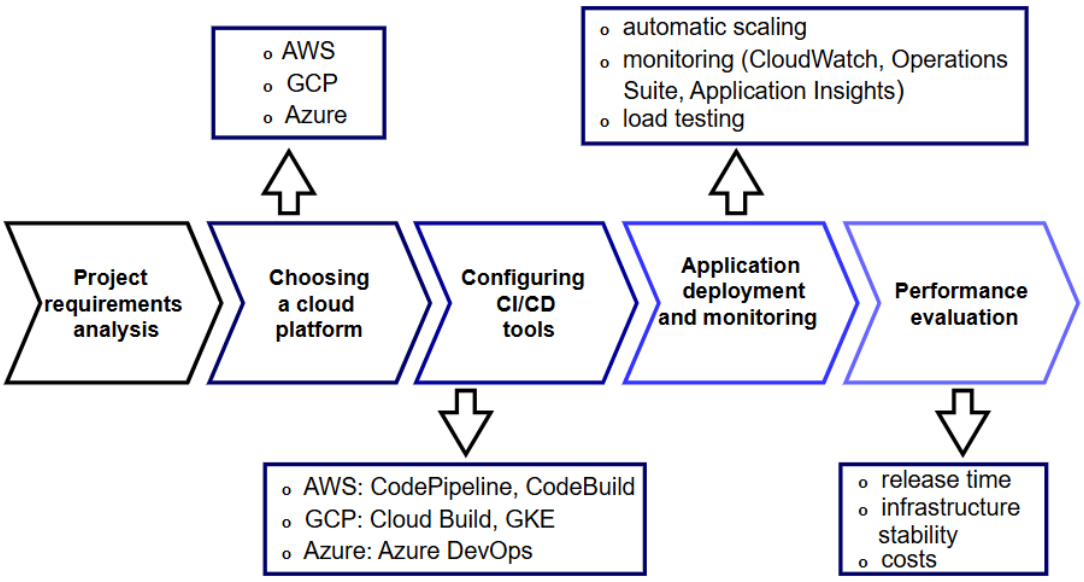


**Figure 1.** The process of implementing DevOps in a cloud environment

The objects of the study were cloud services and tools for automating and optimizing software development and operation from such platforms as AWS, GCP, and Azure. For the AWS platform, we used CodePipeline, CodeBuild, CloudFormation, and CloudWatch services that provide CI/CD process management, infrastructure automation, and monitoring. Google Cloud Platform services included Google Kubernetes Engine, Cloud Build, and Operations Suite, which provided container orchestration, automated application build, and monitoring. Microsoft Azure was used for Azure DevOps, Azure Pipelines, and Azure Monitor, which provided centralized tools for managing the development lifecycle, CI/CD, and performance monitoring.

Different types of software applications were piloted for each platform. These included web applications built on Docker containers, Python server applications using the Flask framework and the PostgreSQL database, and a REST API for processing requests. This made it possible to evaluate the effectiveness of the platforms in real-world use cases, including their performance, ease of setup, and scalability. Key performance indicators were identified to assess the impact of cloud technologies on DevOps processes, which allow for a comprehensive analysis of the capabilities of different platforms.

One of the important indicators was the time to set up CI/CD processes, which reflects the duration of the complete integration and delivery cycle with automation of testing and deployment. The second indicator is the duration of the release cycle, which characterizes the average time from the start of development to the deployment of a functional release in the production environment. The availability of monitoring tools was also studied, which determines the ability of the platform to effectively collect metrics, analyse logs, and identify application errors. Another key aspect was the scalability of the infrastructure, measured by the time and cost required to increase or decrease resources dynamically.

These indicators made it possible to objectively assess the effectiveness of AWS, GCP, and Azure services in various use cases. They provided a basis for comparing their capabilities in the context of DevOps. To achieve the research goal, multiple methods were used to objectively assess the effectiveness of cloud platforms in the DevOps environment. The experimental deployment of applications was carried out using each platform's built-in CI/CD tools. We analysed the speed of process setup, service stability, and process performance (total release cycle time and resource utilization during deployment). Monitoring tools assessed performance, including CloudWatch for AWS, Operations Suite for GCP, and Azure Monitor. Key indicators such as response time, resource utilization, and scaling efficiency were measured. The cost comparison included an analysis of the cost of deploying the same software applications on each platform, taking into account standard tariff plans. Both initial costs and long-term financial implications of using each provider were considered.

## 5. EXPERIMENTS

To evaluate the impact of AWS, GCP, and Azure cloud platforms on DevOps processes, a series of experiments were conducted to simulate real-world scenarios of using cloud services in developing and operating software systems. The experiments were organized in the form of three separate DevOps cycles, each of which was implemented on a different cloud platform, ensuring that the features of each of them were reflected. The AWS platform used CodePipeline services to automate integration and delivery processes, CodeBuild for compilation and testing, and CloudFormation for managing infrastructure as code. The resources were deployed in the us-east-1 region, which allowed us to analyse the platform's performance under standard load conditions. For GCP, Cloud Build was used, which provides quick creation of CI/CD pipelines, and Google Kubernetes Engine, which provides container orchestration. Experiments were conducted with containerized applications deployed in the us-central1 region to ensure stability and analyse scalability in a containerized environment. Azure DevOps was configured in the

Azure environment using Azure Pipelines to organize CI/CD processes. Azure Resource Manager was used to manage resources, which allowed us to implement IaC principles. The deployment was carried out in the East US region, where integration with other Microsoft services was also evaluated.

Each cycle was created and tested under similar conditions, which made it possible to compare the platforms' effectiveness and compliance with different DevOps scenarios. For each of the DevOps cycles, experimental tests were conducted to evaluate the effectiveness of the platforms in key scenarios that model real-world conditions.

The first scenario involved scaling servers under load. We simulated an increase in the number of requests to the application from 100 to 10,000 users within 10 minutes. The tests evaluated the platform's response time to the load increase, as well as the costs associated with the automatic scaling of resources. The second scenario concerned the response time when deploying updates. The experiments included deploying new versions of the application with minimal changes to the code. The time of the main CI/CD stages, such as build, test, and deployment, was measured, which allowed us to assess the speed and stability of the update delivery process. The third scenario involved monitoring and managing errors. The platforms were tested for the ability to integrate logging and monitoring tools to analyse application performance. Additionally, the system's response to errors, such as exceeding resource limits or errors in the source code, was checked, which made it possible to assess the reliability and flexibility of the platforms in handling faults.

These scenarios allowed us to obtain comparative data on cloud platforms' performance, scalability, and stability in real-world conditions. The experimental methodology was developed to ensure the comparability and accuracy of the data obtained. The infrastructure of each platform was based on the use of standard virtual machines with a configuration of 4 vCPUs and 16 GB of RAM, which ensured equal conditions for all tests.

Special Python scripts were created to automate the experiments that used Terraform tools for infrastructure management and Locust for load simulation. These tools ensured the accuracy and repeatability of each test. Performance was monitored using the built-in tools of each platform: CloudWatch for AWS, Operations Suite for GCP, and Azure Monitor for Microsoft Azure. Key metrics were collected, such as system response time, CPU/RAM resource utilization, and request execution delays.

To increase the reliability of the results, each scenario was run three times. This minimized the influence of random factors and provided a more reliable assessment of platform performance and efficiency in different conditions.

## 6. RESULTS

The experiments made it possible to evaluate the effectiveness of using AWS, GCP, and Azure cloud platforms in DevOps processes by key indicators. The advantages and disadvantages of each platform were identified depending on the usage scenario. AWS has demonstrated high functionality and efficiency in DevOps processes due to a wide range of services supporting automation and scalability. Among the platform's advantages is the automation provided by CodePipeline, CodeBuild, and CloudFormation services. This allowed us to significantly reduce the execution time of CI/CD processes after the initial setup. The scalability of the platform is realized through Auto Scaling Groups, which quickly respond to changes in load. CloudWatch also provided a wide range of real-time metrics, allowing us to identify problems and analyse application performance quickly. The platform's disadvantages include the complexity of infrastructure setup for novice users, making the process difficult for less experienced teams. In addition, AWS proved to be the most expensive platform in long-term use scenarios, which can be a significant factor for organizations with limited budgets.

GCP has demonstrated high efficiency in working with containerized applications, which makes the platform especially attractive for modern scenarios where containerized orchestration is a key aspect. Thanks to the Google Kubernetes Engine, the platform provided the best results in container management, showing stability even under conditions of sudden changes in load. GKE's scaling tools maintained performance under high loads, ensuring application reliability. Monitoring on the platform is organized using Google Operations Suite, which provides detailed metrics and analytics, greatly facilitating the control and optimization of applications. However, GCP has some drawbacks. Compared to AWS and Azure, the platform offers a less flexible set of tools for building CI/CD processes, which limits the ability to customize them. In addition, integration with third-party tools requires additional effort, which can pose some difficulties for teams with complex infrastructure requirements.

Azure has proven highly effective in providing an integrated environment for DevOps teams, especially those heavily using Microsoft products. The platform offers convenient tools for centralized management of all development lifecycle stages through Azure DevOps. Integration with services such as Visual Studio, GitHub, and Active Directory greatly simplifies work for organizations already operating in the Microsoft ecosystem. Azure Pipelines made setting up CI/CD for existing Microsoft projects easy, allowing them to focus on development without significant infrastructure changes. However, some aspects of the platform were less competitive. The scalability of resources compared to AWS and GCP required more manual intervention, which affected the speed of adaptation to changes in load. In addition, monitoring tools such as Azure Monitor required additional configuration to obtain detailed analytical data, which could complicate the initial implementation phase.

Table 1 summarizes the results of these experiments, providing a comparative analysis of the advantages and disadvantages of each platform in different use cases.

**Table 1**

Comparison of the results of experiments on the platforms.

| Indicator | AWS | GCP | Azure |
|---|---|---|---|
| CI/CD setup time | Long (due to the flexibility of the tools) | Moderate | Minimal (quick integration with the Microsoft ecosystem) |
| Release cycle time | Shortest (due to automation) | Medium (limitations in CI/CD tools) | Depends on integration with other services |
| Infrastructure scaling | Automatic and highly flexible | Effective for containers (GKE) | Requires additional customization |
| Monitoring and analytics | CloudWatch with extensive features | Stackdriver, but with limited customization | Application Insights with basic functionality |
| Performance under load | Excellent performance | Effective when using containers | Depends on the type of workload |

General trends in using cloud platforms for DevOps show differences in automation, containerization, scalability, and monitoring approaches. AWS has demonstrated the highest level of automation due to its developed ecosystem of CI/CD tools. GCP and Azure also support automation but have certain limitations regarding the flexibility of settings. GCP has proven to be the most effective platform for running containerized applications due to its deep integration with Kubernetes. Although AWS and Azure also offer similar capabilities, their implementation requires more effort to configure and manage. AWS provided the fastest response to load changes with Auto Scaling, while Azure faced the most difficulties in this category due to the need for additional customization for scaling. In terms of monitoring, GCP and AWS provided

the most detailed metrics and analytics, which made it easier to monitor application performance. Azure, on the other hand, required additional configuration to obtain the full range of analytical data, which complicated the implementation.

An experimental study of the impact of AWS, GCP, and Azure cloud platforms on DevOps processes allows us to evaluate their performance by several key indicators, such as CI/CD setup time, release cycle time, infrastructure scalability, monitoring efficiency, and performance under load. The experimental results show that the choice of a cloud platform depends on the needs of a particular project: AWS is suitable for scenarios with high automation requirements, GCP for containerized applications, and Azure for Microsoft ecosystems.

## 7. DISCUSSION

The study results suggest that AWS, GCP, and Azure cloud platforms have significant potential for optimizing DevOps processes. Still, each platform has its specific strengths and limitations, making their use dependent on the needs and characteristics of a particular project. Table 2 shows a comparative profile of the three leading platforms - AWS, GCP, and Azure - in terms of their impact on key aspects of DevOps, such as CI/CD automation, containerization support, scalability, and ease of integration.

**Table 2**

Comparative characteristics of AWS, GCP, and Azure in the context of DevOps.

| Criterion | AWS | GCP | Azure |
|---|---|---|---|
| CI/CD tools | CodePipeline, CodeBuild, CodeDeploy | Cloud Build | Azure DevOps |
| Working with containers | Support via Amazon ECS, EKS | High integration with GKE | Azure Kubernetes Service |
| Scalability | High, automatic scaling | Effective for containerized solutions | Additional settings for scaling |
| Integration | Flexible integration with many services | Focus on the Google ecosystem | Best integration with Microsoft products |
| Team training | High requirements to the level of knowledge | Medium level of complexity | Most convenient for teams using Microsoft products |
| Cost | High for long-term projects | Moderate, depends on the amount of data | Competitive for enterprise customers |

AWS demonstrates high efficiency in complex scenarios thanks to a developed ecosystem of tools for automation, scaling, and monitoring. Using services such as CodePipeline and CloudFormation, you have complete control over CI/CD processes and infrastructure management. This platform is the best choice for projects that require a fast release cycle, complex architecture, and rapid scaling as the load changes. Despite its advantages, AWS has certain limitations, including high cost of use, especially for large and long-term projects. In addition, DevOps engineers need a high level of skill to configure services effectively.

Google Cloud Platform has the best support for container technologies thanks to the Google Kubernetes Engine. The focus on Kubernetes makes this platform ideal for organizations implementing microservice architecture and working with containerized applications. GCP is an optimal solution for startups developing innovative container-based products and companies that actively use Big Data and machine learning. Thanks to its deep integration with TensorFlow and other ML tools, the platform facilitates the effective implementation of advanced artificial intelligence technologies. Despite its significant advantages, GCP has certain limitations, including

a smaller selection of CI/CD tools compared to AWS and difficulties with integration into existing IT systems due to a less developed ecosystem.

Azure features deep integration with the Microsoft ecosystem, making it an ideal choice for companies using Microsoft products such as Windows Server, Active Directory, and Microsoft 365. Azure DevOps tools provide centralized management of all stages of DevOps processes, allowing teams to effectively control application development, testing, deployment, and monitoring. This platform is especially effective for organizations operating in a Windows-based enterprise environment and for small and medium-sized teams that need to integrate with their existing infrastructure quickly. Despite the high level of integration, Azure has certain limitations. Scaling processes require more configuration than AWS and GCP, and monitoring tools require additional configuration to achieve the level of detail available in other cloud platforms.

Cloud platforms provide a wide range of opportunities for automation, scaling, and efficient infrastructure management, significantly reducing application deployment time and increasing DevOps teams' productivity. The availability of advanced technologies such as Kubernetes, AI/ML tools, and built-in CI/CD solutions makes them important for modern development. However, the implementation of cloud technologies is accompanied by several challenges. One of the key aspects is the need for training, as teams need time to master the specific services of each platform. Lack of proper training can lead to inefficient use of resources and configuration errors. Financial costs also play a significant role. Initial investments in cloud infrastructure, scaling, and monitoring can be substantial, especially for startups and small companies. Another critical factor is vendor dependency. Using specialized tools of a particular cloud platform creates the risk of vendor lock-in, which complicates project migration to another platform and can lead to additional costs in the future.

The choice of platform depends on many factors, such as the nature of the project, budget, team size, and experience. AWS is ideal for complex and large-scale projects. GCP is the best choice for microservice architectures and containerization. Azure offers an integrated environment for companies that already work with Microsoft products. Thus, each platform contributes to the transformation of DevOps processes, but maximum efficiency is achieved by choosing the right technology to meet business needs.

## 8. CONCLUSIONS

Cloud technologies play a key role in improving the efficiency of DevOps processes by introducing automation, scalability, and integration with modern infrastructure and application management tools. AWS, GCP, and Azure platforms, each with its own strengths, open up new opportunities for companies looking to optimize their development processes.

AWS provides the highest level of automation and offers a wide range of tools for managing complex CI/CD processes. Thanks to services such as CodePipeline and CloudFormation, the platform can significantly speed up infrastructure deployment and management. However, its implementation requires significant initial time and financial resources, which can be a barrier for small teams or startups. Google Cloud Platform specializes in working with containerized applications, providing efficient orchestration through the Google Kubernetes Engine. Integration with Big Data and machine learning tools makes the platform attractive for high-tech projects. However, the set of built-in tools for CI/CD is limited compared to competitors, which may require additional solutions. Azure is the best choice for organizations that actively use the Microsoft ecosystem. Deep integration with Visual Studio, Active Directory, and other services simplifies the implementation of DevOps processes. However, the platform has certain scalability limitations, and its monitoring tools require additional customization to collect analytical data fully. The platform choice depends on the project's specifics, including scaling needs, budget, technologies used, and the level of team competence. Cloud technologies continue to evolve,

offering new functionality that allows companies to effectively adapt to the growing demands of the modern digital environment. In this context, the study of their impact on DevOps processes remains a promising and important area for industry development.

**References**


1. El Aouni F. et al. (2024). A systematic literature review on Agile, Cloud, and DevOps integration: Challenges, benefits. Information and Software Technology, pp. 107569. https://doi.org/10.2139/ssrn.4827875
2. Battina D. S. Devops (2020) A New Approach To Cloud Development & Testing. nternational Journal of Emerging Technologies and Innovative Research (www. jetir. org), ISSN. pp. 2349–5162. Available at: http://www.jetir. org/papers/JETIR2008432.pdf.
3. Sravan S. S. et al. Significant Challenges to espouse DevOps Culture in Software Organisations By AWS: A methodical Review. 2023 9th International conference on advanced computing and communication systems (ICACCS). IEEE, 2023, vol. 1, pp. 395–401. https://doi.org/10.1109/ICACCS57279.2023.10113021
4. Raheja Y., Borgese G., Felsen N. (2018). Effective DevOps with AWS: Implement continuous delivery and integration in the AWS environment. Packt Publishing Ltd, 363 p.
5. Alalawi A., Mohsin A., Jassim A. A survey for AWS cloud development tools and services. IET Conference Proceedings CP777. Stevenage, UK: The Institution of Engineering and Technology, 2020, vol. 2020, no. 6, pp. 17–23. https://doi.org/10.1049/icp.2021.0898
6. Campbell B., Campbell B. (2020). CloudFormation In-Depth //The Definitive Guide to AWS Infrastructure Automation: Craft Infrastructure-as-Code Solutions, pp. 55–122. https://doi.org/10.1007/978-1-4842-5398-4_3
7. Mishra P. (2023). Advanced AWS Services. Cloud Computing with AWS: Everything You Need to Know to be an AWS Cloud Practitioner. Berkeley, CA: Apress, pp. 247–277. https://doi.org/10.1007/978-1-4842-9172-6_9
8. Lekkala C. (2023) Deploying and Managing Containerized Data Workloads on Amazon EKS. J Arti Inte & Cloud Comp, vol. 2, no. 2, pp. 1–5. https://doi.org/10.47363/JAICC/2023(2)324
9. Singh C. et al. (2019) Comparison of different CI/CD tools integrated with cloud platform. 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence). IEEE, pp. 7–12. https://doi.org/10.1109/CONFLUENCE.2019.8776985
10. Shah J., Dubaria D. (2019) Building modern clouds: using docker, kubernetes & Google cloud platform. 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, pp. 0184–0189. https://doi.org/10.1109/CCWC.2019.8666479
11. Bisong E., Bisong E. (2019). Containers and google kubernetes engine. Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners, pp. 655–670. https://doi.org/10.1007/978-1-4842-4470-8_45
12. Esfahani H. et al. (2016). CloudBuild: Microsoft's distributed and caching build service. Proceedings of the 38th International Conference on Software Engineering Companion, pp. 11–20. https://doi.org/10.1145/2889160.2889222
13. Sukhdeve D. S. R., Sukhdeve S. S. (2023). Introduction to GCP. Google Cloud Platform for Data Science: A Crash Course on Big Data, Machine Learning, and Data Analytics Services. Berkeley, CA: Apress, pp. 1–9. https://doi.org/10.1007/978-1-4842-9688-2_1
14. Riti P., Riti P. (2018). Monitoring in GCP //Pro DevOps with Google Cloud Platform: With Docker, Jenkins, and Kubernetes, pp. 165–190. https://doi.org/10.1007/978-1-4842-3897-4_7
15. Wang I. (2024). Provisioning Infrastructure on GCP //Terraform Made Easy: Provisioning, Managing and Automating Cloud Infrastructure with Terraform on Google Cloud. Berkeley, CA: Apress, pp. 95–170. https://doi.org/10.1007/979-8-8688-1010-7_4
16. Barrientos A., Duran Huanca J. G., Mayta Segovia H. A. (2023). Implementation of a Software Engineering Model with DevOps on Microsoft Azure. Proceedings of the 2023 8th International Conference on Information Systems Engineering, pp. 1–6. https://doi.org/10.1145/3641032.3641037
17. Narayanan P. K. (2024). Engineering Data Pipelines Using Microsoft Azure. Data Engineering for Machine Learning Pipelines: From Python Libraries to ML Pipelines and Cloud Platforms. Berkeley, CA: Apress, pp. 571–616. https://doi.org/10.1007/979-8-8688-0602-5_17
18. Satapathi A., Mishra A. (2022). Deploy an ASP. NET Web Application to an Azure Web App Using GitHub Actions. Developing Cloud-Native Solutions with Microsoft Azure and. NET: Build Highly Scalable Solutions for the Enterprise. Berkeley, CA: Apress, pp. 249–270. https://doi.org/10.1007/978-1-4842-9004-0_11
19. Chandrasekara C. et al. (2020). Getting Started with Azure Git Repos. Hands-on Azure Repos: Understanding Centralized and Distributed Version Control in Azure DevOps Services, pp. 139–170. https://doi.org/10.1007/978-1-4842-5425-7_7
20. Sahay R., Sahay R. (2020). Azure monitoring. Microsoft Azure Architect Technologies Study Companion: Hands-on Preparation and Practice for Exam AZ-300 and AZ-303, pp. 139–167. https://doi.org/10.1007/978-1-4842-6200-9_5
21. Satapathi A., Mishra A. (2021). Enabling Application Insights and Azure Monitor. Hands-on Azure Functions with C#. Apress, Berkeley, CA, pp. 233–261. https://doi.org/10.1007/978-1-4842-7122-3_10

22. Chawla H. et al. (2019). Azure kubernetes service. Building Microservices Applications on Microsoft Azure: Designing, Developing, Deploying, and Monitoring, pp. 151–177. https://doi.org/10.1007/978-1-4842-4828-7_5
23. Borra P. (2024) Comparison and Analysis of Leading Cloud Service Providers (AWS, Azure and GCP). International Journal of Advanced Research in Engineering and Technology (IJARET), vol. 15, no. 3, pp. 266–278. https://doi.org/10.2139/ssrn.4914145
24. Borra P. (2024) Comparative Review: Top Cloud Service Providers ETL Tools-AWS vs. Azure vs. GCP. International Journal of Computer Engineering and Technology (IJCET), vol. 15, pp. 203–208. https://doi.org/10.2139/ssrn.4914175
25. Kingsley M. S. (2023). Comparing AWS, Azure, and GCP. Cloud Technologies and Services: Theoretical Concepts and Practical Applications. Cham: Springer International Publishing, pp. 381–393. https://doi.org/10.1007/978-3-031-33669-0_12

**УДК 004.75:004.42**

# ВПЛИВ СУЧАСНИХ ХМАРНИХ ТЕХНОЛОГІЙ НА ЕФЕКТИВНІСТЬ DEVOPS-ПРОЦЕСІВ

**Михайло Лучкевич[1]; Ірина Шаклеіна[1]; Олексій Дуда[2]**

*[1] Національний університет «Львівська політехніка», Львів, Україна*
*[2]Тернопільський національний технічний університет імені Івана Пулюя, Тернопіль, Україна*

***Резюме.*** *Досліджено вплив сучасних хмарних технологій на ефективність DevOps-процесів, що є ключовими для автоматизації, гнучкості та надійності розроблення програмного забезпечення. В умовах стрімкого розвитку інформаційних технологій хмарні платформи відіграють важливу роль у прискоренні процесів розроблення, тестування та розгортання програмних продуктів. Упровадження DevOps у хмарному середовищі дозволяє зменшити час виходу продукту на ринок, підвищити стабільність роботи програмних систем й оптимізувати використання обчислювальних ресурсів, що є критично важливим для сучасних компаній. Розглянуто основні виклики, з якими стикаються DevOps-команди, зокрема необхідність швидкого масштабування інфраструктури, забезпечення безперервної інтеграції та доставки (CI/CD), автоматизації тестування, моніторингу продуктивності та оптимізації витрат. Особлива увага приділена проблемам безпеки, управління конфігурацією та мінімізації людського фактора при впровадженні змін у програмний код. Проведено порівняльний аналіз трьох провідних хмарних платформ – Amazon Web Services, Google Cloud Platform та Microsoft Azure – у контексті їхнього впливу на DevOps-процеси. Оцінено можливості автоматизації, підтримання контейнерних технологій, масштабованості інфраструктури, інструментів моніторингу та інтеграції з CI/CD-конвеєрами. Виявлено, що AWS пропонує найширший набір DevOps-інструментів із високим рівнем автоматизації, що робить її привабливим вибором для організацій, орієнтованих на складні корпоративні рішення. Google Cloud Platform демонструє найкращу підтримку Kubernetes і контейнеризації, що є важливим фактором для команд, які працюють з мікросервісною архітектурою. Microsoft Azure забезпечує найглибшу інтеграцію з екосистемою Microsoft і є оптимальним вибором для компаній, які використовують Windows-середовище та пов'язані продукти. Експериментальне дослідження показало, що вибір хмарної платформи має суттєвий вплив на швидкість релізних циклів, стабільність розгорненої інфраструктури, можливість швидкого реагування на зміну навантаження та продуктивність DevOps-команд у цілому. Запропоновано рекомендації щодо вибору оптимальної хмарної платформи залежно від специфіки проєкту, масштабу організації, рівня навантаження на систему та вимог до автоматизації.*

***Ключові слова:*** *DevOps, хмарні технології, AWS, GCP, Azure, CI/CD, автоматизація, контейнеризація, масштабованість, моніторинг.*

*https://doi.org/10.33108/visnyk_tntu2025.01.112* *Отримано 20.02.2025*