International Journal of Enhanced Research in Science, Technology & Engineering
ISSN: 2319-7463, Vol. 9 Issue 10, October-2020, Impact Factor: 6.754

Architecting Scalable and Secure Infrastructure for DevOps, Patterns and Practices for Cloud, Hybrid, and On-Premise Environments

Lakshmi Prasad Rongali[1], Ganga Ashok Kumar Budda[2]

[1]Meridian Cooperative Inc, USA
[2]Principal Product Development Project Manager

ABSTRACT

This research explores architectural strategies and best practices for creating DevOps-enabled infrastructure that is scalable, automated, and secure. Employing a qualitative, deductive research strategy with second-level data sources, this work compares the scalability of cloud, hybrid, and on-premises infrastructure deployments. Salient findings indicate the importance of Infrastructure as Code, container orchestration, and integrated toolchains. The complexities of integrating, as well as limitations of legacy systems, are discussed, including directions towards the future with AIOps and serverless technologies. The work delivers actionable guidelines for building DevOps-enabled infrastructure in the enterprise of today.

Keywords: DevOps, Infrastructure as Code, Cloud Computing, Hybrid Infrastructure, Automation, Scalability, Security, CI/CD, Containerization, Infrastructure Design

INTRODUCTION

DevOps has emerged as a new-age practice founded upon collaboration, flexibility, and continuous improvement in the software delivery dynamic. The success of DevOps approaches relies greatly on the underlying infrastructure. The cloud, hybrid, or on-prem configurations, organizations could have to design a scalable, automatic, and secure infrastructure from the outset to enable an effective deployment pipeline and stable operations. This research explores the underlying design patterns and best practices for designing infrastructure in line with the spirit of DevOps. Focused on scalability, security, and automation, the paper explains how new-age enterprises can optimize infrastructure to support rapid delivery cycles, reduce manual interventions, and maintain compliance. This research also presents comparative observations across deployment types and suggests approaches to implement in diverse operational conditions.

Aim

The aim is to explore and recommend architectural strategies for building scalable, automated, and secure infrastructure that supports effective DevOps practices.

Objective

- To explore architectural patterns that support scalable, automated, and secure DevOps infrastructure across cloud, hybrid, and on-premise environments.
- To evaluate the effectiveness of current infrastructure practices in enabling continuous integration, deployment, and operational efficiency in DevOps workflows.
- To identify key infrastructure-related challenges organizations face while implementing DevOps in varying deployment environments and operational contexts.
- To recommend best practices and implementation strategies for designing DevOps-ready infrastructure that ensures scalability, automation, and security from inception.

Research Question

1. What architectural patterns are most effective in building scalable, automated, and secure infrastructure for DevOps environments?

2. How effective are current infrastructure practices in supporting continuous integration, deployment, and operational efficiency within DevOps frameworks?
3. What is the primary infrastructure-related challenges faced by organizations when implementing DevOps across cloud, hybrid, and on-premise environments?
4. What best practices and strategies can be recommended for designing foundational infrastructure that aligns with DevOps principles of scalability, automation, and security?

Research Rationale

The institutions are adopting DevOps to increase software delivery and operational productivity, and the need for infrastructure supporting its underlying tenets, scalability, automation, and security has become the top priority. Legacy systems, skewed deployment environments, and the lack of adequate automation solutions, however, make numerous companies struggle to align their infrastructure with DevOps [1]. Convergence to cloud, mixed, and on-premise-based solutions contributes to the complexity of infrastructure design, necessitating agile and resilient architectures. Bridging the gap between DevOps goals and underlying infrastructure design is the need of the hour [2]. Through an inquiry into sound architectural patterns and an analysis of best practices, this research aims to provide decision-enabling insights to design infrastructure that is DevOps-ready from the outset. Challenges and recommendations in the form of tailormade strategies can help firms accelerate development pipelines, reduce deployment failures, and strengthen system security. The findings can empower IT decision-makers, architects, and DevOps practitioners to make well-informed infrastructure decisions with the potential to drive successful digital transformation.

LITERATURE REVIEW

Architectural Patterns for Scalable and Secure DevOps Infrastructure Design

The success of DevOps deployment is highly reliant on the infrastructure architectural foundation of an enterprise. Sources point out that there is a need for secure and scalable infrastructure to enable continuous integration, continuous delivery (CI/CD), and agile development cycles [3]. Microservices architecture and containerization are among the most popular adopted patterns that provide modularity, scalability, and rapid deployment. Microservices enable applications to be divided into independent pieces so that the development, deployment, and scaling of the features are done independently by different teams [4]. This is aligned with the principles of DevOps as it provides rapid iteration and decreases system-level failures. One of the significant design aspects is the implementation of the pattern of the use of Infrastructure as Code (IaC), which promotes repeatability, versioning, and the use of tools such as Terraform, AWS CloudFormation, and Ansible [5]. IaC significantly impacts infrastructure consistency and reduces the issue of configuration drift, which is significant in complex, multi-environment DevOps pipeline implementations.

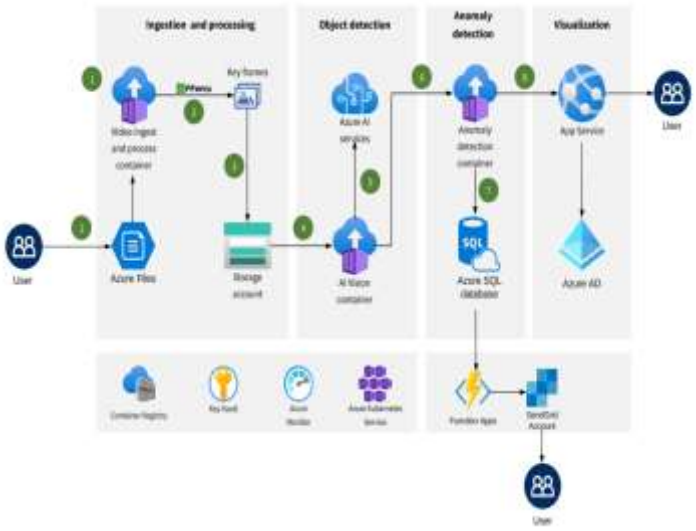

Fig 1: End-to-End AI-Powered Video Analysis Pipeline Using Azure Cloud Services

Immutable infrastructure and zero-trust design tackle security as part of the design of infrastructure. Immutable infrastructure does not allow components to be modified once they have been launched, reducing the attack surface and misconfigurations [6]. Zero-trust approaches, however, enforce strict access controls and do not trust implicitly, even

within the company network. There is also the role of architectures that are both cloud-native as well as hybrid. Cloud platforms, particularly in combination with Kubernetes and service meshes such as Istio, can offer scalable orchestration as well as secure connectivity between services, the research suggests [7]. In general, the literature is focused on well-managed architectural designs as being central to maintaining a resilient, scalable, and secure DevOps system.

Evaluating Infrastructure Effectiveness in Supporting DevOps Practices

The success of DevOps heavily relies on the capacity of the underlying infrastructure to support agile, continuous, and automated processes. Effectiveness is exemplified in good infrastructure, with the infrastructure being resilient, scalable, and integrally coupled with DevOps tools to enable seamless CI/CD, monitoring, and rollbacks [8]. The use of Infrastructure as Code (IaC) is an essential enablement that has allowed versioning, testing, and deployment of infrastructure configurations in a repeatable manner across varied environments. This consistency reduces deployment failures and decreases time to production [9]. Several research studies emphasized the necessity of cloud infrastructure, primarily public cloud services like AWS, Azure, and Google Cloud, towards the facilitation of elastic scaling and provisioning of resources, which is essential to process the load and rapid replication of the environment. The efficiency cannot be achieved because the infrastructure is not put in place to support auto-automation and observability from the outset. Technologies such as Jenkins, Prometheus, and Kubernetes are seen to enhance the use of the infrastructure significantly well well-integrated, allowing pipeline executions, monitoring, and container orchestration [10]. It is also empirically evident that legacy on-premises infrastructure is not flexible and automated in nature to support DevOps, which delays the speed of deployment and introduces operational overhead. Hybrid solutions can work well, coordinated to provide the scalability of the cloud with the preservation of legacy investments [11]. Ultimately, the best way to measure the effectiveness of infrastructure is in metrics such as deployment speed, lead time, mean time to restore (MTTR), and system reliability. They are quantifiable factors of how well infrastructure supports DevOps goals.

Challenges in Implementing DevOps Across Diverse Infrastructure Environments

Implementing DevOps in cloud, hybrid, and on-premise infrastructures has numerous technical, operational, as well as organizational issues. The central issue discussed in the literature is infrastructure consistency, where varying infrastructures create configuration drift as well as maintenance issues from a technical point of view [12]. In the hybrid configurations, deployment synchronization between the cloud and on-premise systems requires good orchestration tools and solid standards of interfaces, which are often lacking or in the process of development.Fragmentation in the toolchain is also a significant challenge. Organizations also face the challenge of standardizing tools for monitoring, configuration control, and CI/CD across different platforms, and silos and operational complexity are the consequences of this [13]. Moreover, legacy infrastructure is not scalable and cannot be automated, which is creating bottlenecks in the DevOps pipeline as well as limiting responsiveness to change.

Fig 2: Key Challenges in Implementing a DevOps Plan

Security and compliance contribute to the complexity of the implementation process. Studies reveal that using DevSecOps practices in heterogeneous systems is challenging with inconsistent control over access, disparate encryption standards, and

different regulatory expectations [14]. It is more so in industries with tight data governance requirements.In a cultural sense, DevOps demands cross-functional teamwork, which is difficult to apply in traditional IT departments. Change resistance, lack of proper training, and unclear responsibilities typically prevent the implementation of the practices of DevOps, especially in heterogeneous infrastructure management [15]. The successful DevOps adoption in varying infrastructure conditions requires technological alignment as well as organizational transformation, standard tools, and integrated security. Bridging each of these is critical to realizing the full potential of DevOps in complex enterprise systems.

Best Practices and Strategies for DevOps-Ready Infrastructure Implementation

Natively DevOps-supporting infrastructure requires the application of best practices that ensure to deliveryof infrastructure that is automatable, scalable, resilient, and secure. One of the most essential strategies in the literature is the application of Infrastructure as Code (IaC), which enables repeatable, versioned, and testable infrastructure deployments. Terraform, Ansible, and AWS CloudFormation are some of the tools the team applies to treat infrastructure as application code, reduce manual errors, and accelerate the deployments [16]. There is also another significant practice, which includes containerization using tools like Docker and orchestration using Kubernetes. This maintains portability, speed of scaling, and uniformity of the environment, which are the basics of successful CI/CD strategies [17]. Moreover, immutable infrastructure practice, where servers are swapped instead of being updated, minimizes drift and ensures more stable rollbacks.

Continuous monitoring and observability are also prioritized as recommended practices. Prometheus, Grafana, and the ELK stack make possible the identification of issues in their formative stages, the enhancement of performance, and the enabling of real-time decision-making. Security is best managed through DevSecOps, in which security testing is incorporated in the pipeline and policies are implemented through HashiCorp Vault and security code analysers [18]. In addition, the alignment of the infrastructure strategy with cloud-native architectures such as elasticity, self-healing, and auto-provisioning provides the greatest operational agility and resource utilisation. Organizations are also encouraged to conduct routine infrastructure audits and implement feedback loops to continue to improve [19]. It is generally advised to follow a holistic, automation-driven, and security-focused approach to design DevOps-enabled infrastructure scalable enough to support expanding enterprise demands.

Literature Gap

Though the literature is full of DevOps tools, practices, and general infrastructure concepts in general, there is a major void in the mapping of these aspects across different deployment modes, cloud, hybrid, and on-premise, within a common framework. Relatively few studies offer insights regarding the architectural designs of each of these environments, especially regarding real-time scalability, automating different aspects of complexity, and security tradeoffs [20]. There is also little specific research on the implementation problems of legacy systems and in the hybrid model, where the heterogeneity of the resources is difficult to integrate. It is indicative of the need for holistic approaches to map the underlying infrastructure to DevOps concepts, but also considering operational, organizational, as well as technological limitations.

METHODOLOGY

A ***qualitative approach*** is used in this research to investigate the best ***architectural approaches*** for setting up DevOps-compatible infrastructure. Researchers use a ***deductive approach*** where starting points are already defined objectives and theories from previous studies. ***Secondary data*** from peer-reviewed articles, whitepapers, technical reports, and case studies published by AWS, Microsoft Azure, and Google Cloud forms the basis of the analysis.

The goal of data collection is to discover important points about the scalability, automation, and security of DevOps infrastructure in any setting. The data is studied using ***thematic analysis,***which can spot and sort out similar issues and ideas. Researchers learn more about infrastructure practices and challenges that are important for the study's aims by studying communities. The coding and analysis of themes focused on their link to what the research aims to understand, making it easier to evaluate current DevOps practices.

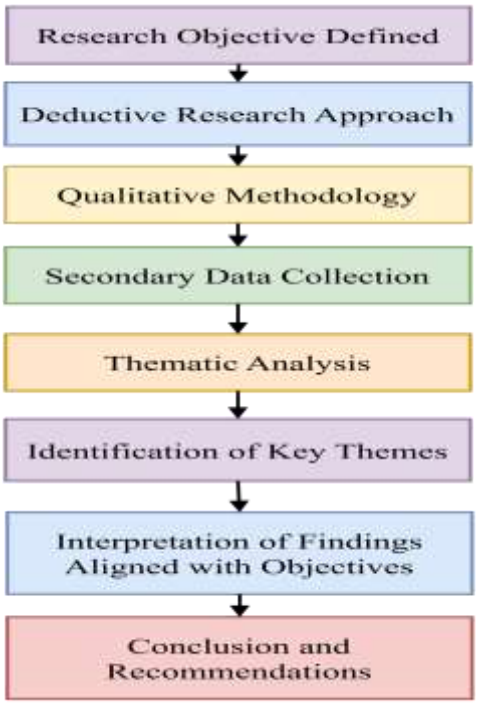

Fig 3: Research Flowchart

DATA ANALYSIS

Theme 1: Analysis of Architectural Components Supporting Scalable and Secure DevOps Infrastructure Across Deployment Models.

From secondary data, we can see that for DevOps to be effective, infrastructure needs to be easy to scale and fully secure. In all cloud, hybrid, and on-premise environments, a few architectural elements turn out to be important [21]. Both containerization and following a microservices architecture are most used in cloud and hybrid systems. Thanks to this, fault tolerance and resilience can be achieved, as you can deploy modules separately, expand horizontally, and isolate each part.

The deployment runs the same in all environments and lowers the risks of errors caused by different systems.

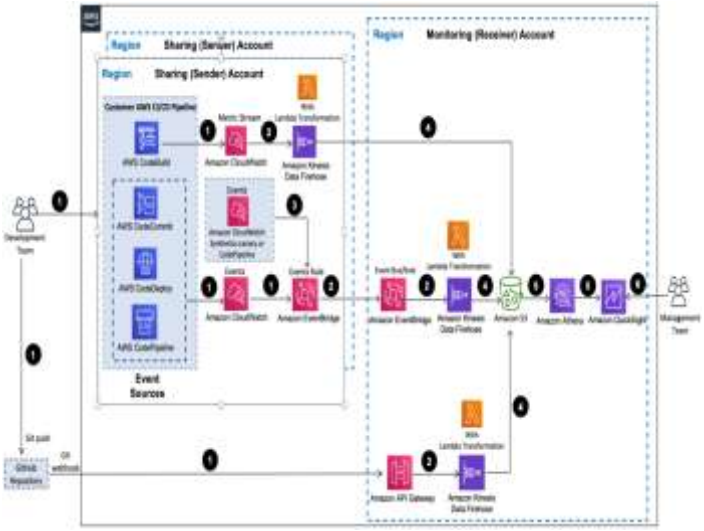

Fig 4: AWS-Based DevOps Monitoring and Data Sharing Architecture

Using serverless computing (AWS Lambda and Azure Functions), organizations are attracted by the ability to scale and cut costs. Managing how services interact in a distributed system and guarding security between services is largely thanks to API gateways, service meshes, and load balancers in hybrid models [12]. More companies are now using virtual machines and cloud systems like OpenStack to manage their DevOps needs on-site. In addition, Infrastructure as Code (IaC) allows for more effective automation and ensures configurations are the same, which is important when handling large and complex systems. In all types of deployment models, organizing access controls by using IAM and separating networks into segments has been seen as fundamental for protecting a system's integrity.

Theme 2: Assessment of Infrastructure Effectiveness Using DevOps Performance Metrics and Tool Integration Data.
Infrastructure effectiveness in DevOps is stressed by secondary data from industry reports and case studies as being measured through performance metrics. DevOps' support for infrastructure is typically measured according to deployment frequency, lead time for changes, mean time to recover from failures (MTTR), and the number of unsuccessful changes [23]. It has been noticed that highly efficient infrastructure supports companies to reduce the time needed for deployments and recoveries, showing that the right infrastructure design improves organizational responsiveness and reliability. It is key to measure a developer's ability to integrate new tools and software. GitLab CI and CircleCI, system automation is better with automated CI/CD tools like Jenkins [24]. Meanwhile, monitoring tools like Prometheus and ELK Stack help monitor the health of the system. Deployment and response to issues become much more dependable by smoothly connecting these tools at every layer of the infrastructure.

Cloud-native environments are often better than traditional systems because they can grow without problems and offer well-connected services. At the same time, setting up hybrid and on-premise solutions usually adds extra requirements, and this can result in performance problems because of latency or compatibility with different tools. The use of IaC and automated testing has a strong connection to better and more uniform results in every deployment. Data available demonstrates that aligning tools and infrastructure with DevOps metrics plays a key role in successful operations.

Theme 3: Identification of Key Infrastructure Challenges from Industry Reports and Case Study Comparisons.
Out of several documents analyzed, we saw common infrastructure obstacles that companies must overcome when rolling out DevOps in all types of environments. An important issue arises whenever hybrid platforms are used, since old and new systems must collaborate smoothly [25]. The reason is that configuration drift, unequal environments, and lots of upkeep are common.

Another major issue is that there are many separate toolchains, and many companies add different DevOps services without a well-organized strategy, which ends up creating siloed teams and reducing visibility. Studies reveal that groups using independent CI/CD, monitoring, and security software struggle to have complete and easy-to-see automation and traceability. Integrating security is still a major issue, mainly in infrastructure.DevSecOpswas not originally built in [26]. Because automated scanning and policy enforcement are not used universally, threats of breaches and non-compliance rise. At the same time, problems with required knowledge and people's resistance to change also hold infrastructure transformation back.

A large number of organizations have a hard time getting teams up to speed or adjusting old mindsets to DevOps, which keeps adoption from happening completely. According to the literature, cost barriers and vendor constraints limit how flexible organizations can be with their cloud and hybrid infrastructures. This proves that to reach DevOps maturity, teams should follow a strategy, enforce governance rules, and integrate different tools together.

Theme 4: Evaluation of Best Practices in DevOps Infrastructure Implementation from Technical Whitepapers and Enterprise Surveys.

Analysis of technical whitepapers and enterprise surveys shows that following specific best practices increases the chance of successful DevOps implementation. The main thing to use is Infrastructure as Code (IaC), including Terraform, AWS CloudFormation, and Ansible, for automated and repeatable provisioning and configuration management [27]. Because of this, developers don't have to worry about setting up the environments for each stage. A further best practice is implementing Docker and Kubernetes, both of which mean your applications can be managed using modules, while saving resources and allowing easy scaling. Organizations using these modern tools say they can deploy their software more often and recover from problems more swiftly, following key DevOps measurements.

In addition, it's important to implement monitoring and observability solutions such as Prometheus, Grafana, and ELK Stack. Because these tools give up-to-date information about the system, problems can be resolved, and the system can be improved ahead of issues impacting the organization.

Fig 5: DevOps Lifecycle with Integrated Tools and Platforms

One security best practice is to use DevSecOps practices from the beginning of the project all the way through. The achievement of the businesses is to automate security scans, apply access rules, and manage secrets with Vault and AWS IAM software [28]. The results show that teamwork across various roles, regular feedback, and smarter governance are important for infrastructure to stay agile. These efforts contribute to enterprises building a scalable, secure, and automatically managed infrastructure for success with DevOps.

Future Directions

The research is helpful for future studies to analyze the use ofautomated decision-making system in DevOps and serverless computing to increase how quickly infrastructure changes can be carried out and infrastructure supported. It is also required to design standard principles for DevOps infrastructure in both hybrid and multiple cloud environments, taking care of compatibility and compliance issues [29]. Gaining insight into edge computing and policy-as-code for enhancing prompt processing and organizational governance is also necessary. The research and use of better models to judge best practices that have a lasting effect on organizations that are not part of the Fortune 500, especially during digital transformation. Working together, different DevOps teams can explore and share insights into what makes the practical implementation of DevOps successful.

CONCLUSION

Researchers show that using solid infrastructure helps DevOps be effective wherever the work is done, on the cloud, at home, or at other companies. A review of secondary data allowed the study to recognize basic architectural features, measure how the infrastructure performed by examining DevOps metrics, and examine regular challenges in setting it up. It gave recommendations to ensure the system could grow, be automated, and stay safe. The outcomes highlight why it is important to use Infrastructure as Code, containerize services, and apply integrated toolchains. For organizations on a digital journey, having a carefully planned DevOps-ready infrastructure is necessary to ensure agility, dependability, and innovation as their IT environments become more and more complex and change frequently.

REFERENCES

[1]. Brunnert, A., van Hoorn, A., Willnecker, F., Danciu, A., Hasselbring, W., Heger, C., Herbst, N., Jamshidi, P., Jung, R., von Kistowski, J. and Koziolek, A., 2015. Performance-oriented DevOps: A research agenda. *arXiv preprint arXiv:1508.04752*.

[2]. Yarlagadda, R.T., 2018. Understanding DevOps & bridging the gap from continuous integration to continuous delivery. *International Journal of Emerging Technologies and Innovative Research (www. jetir. org), ISSN*, *2349*(5162), pp.1420-1424.

[3]. Shahin, M., Babar, M.A. and Zhu, L., 2017. Continuous integration, delivery, and deployment: a systematic review on approaches, tools, challenges, and practices. *IEEE access*, *5*, pp.3909-3943.
[4]. Zimmermann, O., 2017. Microservices tenets: Agile approach to service development and deployment. *Computer Science-Research and Development*, *32*, pp.301-310.
[5]. Chinamanagonda, S., 2019. Automating Infrastructure with Infrastructure as Code (IaC). *Available at SSRN 4986767*.
[6]. Putz, B. and Pernul, G., 2019. Trust Factors and Insider Threats in Permissioned Distributed Ledgers: An Analytical Study and Evaluation of Popular DLT Frameworks. *Transactions on Large-Scale Data-and Knowledge-Centered Systems XLII*, pp.25-50.
[7]. Rotsos, C., King, D., Farshad, A., Bird, J., Fawcett, L., Georgalas, N., Gunkel, M., Shiomoto, K., Wang, A., Mauthe, A. and Race, N., 2017. Network service orchestration standardization: A technology survey. *Computer Standards & Interfaces*, *54*, pp.203-215.
[8]. Johnston, J., Batchelli, A., Cormack, J., Fiedler, J. and Gajdos, M., 2015. *Docker in Production*. by: Bleeding Edge Press, Santa Rosa, CA 95404.
[9]. Shahin, M., Babar, M.A. and Zhu, L., 2017. Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. *IEEE access*, *5*, pp.3909-3943.
[10]. Sukhija, N. and Bautista, E., 2019, August. Towards a framework for monitoring and analyzing high performance computing environments using kubernetes and prometheus. In *2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)* (pp. 257-262). IEEE.
[11]. Ali, K.E., Mazen, S.A. and Hassanein, E.E., 2018. A proposed hybrid model for adopting cloud computing in e-government. *Future Computing and Informatics Journal*, *3*(2), pp.286-295.
[12]. Pawlish, M.J. and Varde, A.S., 2018. The DevOps paradigm with cloud data analytics for green business applications. *ACM SIGKDD Explorations Newsletter*, *20*(1), pp.51-59.
[13]. Enemosah, A., 2019. Implementing DevOps Pipelines to Accelerate Software Deployment in Oil and Gas Operational Technology Environments. *International Journal of Computer Applications Technology and Research*, *8*(12), pp.501-515.
[14]. Laracy, J.R. and Marlowe, T., 2018. Systems Theory and Information Security: Foundations for a New Educational Approach. *Information Security Education Journal*, *5*(2), pp.35-48.
[15]. Shahin, M., Zahedi, M., Babar, M.A. and Zhu, L., 2019. An empirical study of architecting for continuous delivery and deployment. *Empirical Software Engineering*, *24*, pp.1061-1108.
[16]. Lwakatare, L.E., Kuvaja, P. and Oivo, M., 2016. An exploratory study of devops extending the dimensions of devops with practices. *Icsea*, *104*, p.2016.
[17]. Al Jawarneh, I.M., Bellavista, P., Bosi, F., Foschini, L., Martuscelli, G., Montanari, R. and Palopoli, A., 2019, May. Container orchestration engines: A thorough functional and performance comparison. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)* (pp. 1-6). IEEE.
[18]. Enemosah, A., 2019. Implementing DevOps Pipelines to Accelerate Software Deployment in Oil and Gas Operational Technology Environments. *International Journal of Computer Applications Technology and Research*, *8*(12), pp.501-515.
[19]. Jensen, C.B. and Winthereik, B.R., 2017. Audit loops and audit implosion. *Redescribing relations. Strathernian conversations on ethnography, knowledge and politics*, pp.146-72.
[20]. Fahmideh, M. and Beydoun, G., 2018. Reusing empirical knowledge during cloud computing adoption. *Journal of Systems and Software*, *138*, pp.124-157.
[21]. Sabiri, K. and Benabbou, F., 2015. Methods migration from on-premise to cloud. *IOSR Journal of Computer Engineering*, *17*(2), pp.58-65.
[22]. Hussain, F., Li, W., Noye, B., Sharieh, S. and Ferworn, A., 2019, October. Intelligent service mesh framework for api security and management. In *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)* (pp. 0735-0742). IEEE.
[23]. Rajkumar, M., Pole, A.K., Adige, V.S. and Mahanta, P., 2016, April. DevOps culture and its impact on cloud delivery and software development. In *2016 International Conference on Advances in computing, communication, & automation (ICACCA)(Spring)* (pp. 1-6). IEEE.
[24]. Moroz, M., 2018. Acceleration of digital transformation as a result of launching programs financed from public funds: Assessment of the implementation of the operational program digital Poland. *Foundations of Management*, *10*(1), p.59. [25] Darwish, A., Hassanien, A.E., Elhoseny, M., Sangaiah, A.K. and Muhammad, K., 2019. The impact of the hybrid platform of internet of things and cloud computing on healthcare systems: opportunities, challenges, and open problems. *Journal of Ambient Intelligence and Humanized Computing*, *10*, pp.4151-4166.

[25]. Ahmed, Z. and Francis, S.C., 2019, November. Integrating security with devsecops: Techniques and challenges. In *2019 International Conference on Digitization (ICD)* (pp. 178-182). IEEE.

[26]. Lourenço, P., Dias, J.P., Aguiar, A. and Ferreira, H.S., 2019. Cloudcity: A live environment for the management of cloud infrastructures. In *Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering*.

[27]. Rath, A., Spasic, B., Boucart, N. and Thiran, P., 2019. Security pattern for cloud SaaS: From system and data security to privacy case study in AWS and Azure. *Computers*, *8*(2), p.34.

[28]. Carturan, S. and Goya, D., 2019, May. Major Challenges of Systems-of-Systems with Cloud and DevOps–a financial experience report. In *2019 IEEE/ACM 7th International Workshop on Software Engineering for Systems-of-Systems (SESoS) and 13th Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems (WDES)* (pp. 10-17). IEEE.