Journal of Sustainable Solutions
ISSN : 3048-6947 | Vol. 1 | Issue 4 | Oct - Dec 2024 | Peer Reviewed & Refereed

Implementing Infrastructure as Code (IaC) for Scalable DevOps Automation in Hybrid Cloud

Venkat Marella
Independent Researcher, USA.

Accepted: 08/12/2024 Published: 14/12/2024 * Corresponding author

How to Cite this Article:

Marella V. (2024). Advancements in Artificial Intelligence: Machine Learning Techniques and Their Real-World Applications, *Journal of Sustainable Solutions*, 1(4), 138-144.

DOI: https://doi.org/10.36676/j.sust.sol.v1.i4.46

Abstract

The devOps approach known as Infrastructure as Code (IaC) automates all of the infrastructure's requirements to improve deployment speed, security, scalability, and automatic backup and restoration. Writing code that explains the infrastructure—which allows resources to be generated, destroyed, scaled, replaced, and relocated with ease—is the focus of Infrastructure as a Code (IaC). Installing an operating system on it, setting up servers on instance, adjusting how the software in the instances communicates with one other, and much more are all part of the scripting environment process. In order to achieve an effective infrastructure across all sectors while maintaining security via the usage of public and private clouds, this paper examines a number of tools and technology sets. By automating infrastructure deployment and procedures, continually enhancing the integration and delivery process, and monitoring application performance indicators, DevOps dismantles communication silos and enhances teamwork and productivity. In DevOps, automation is essential, and "Infrastructure as code (IaC)" is a critical component of automation. The management of infrastructure in cloud and physical datacenter systems will also be covered in this article, along with the impact of agile, DevOps, and IaC on infrastructure management. e. In order to achieve an effective infrastructure across all sectors while maintaining security via the usage of public and private clouds, this paper examines a number of tools and technology sets. Our results indicate that adopting IaC has many advantages, but there may also be some difficulties in putting IaC into practice. Additionally, the research recognizes the contribution of DevOps, cloud systems, and agile to the deployment of Infrastructure as a code.

Keywords: - Infrastructure As Code (IaC), Backup and Restores, Cloud Systems, Agile, Automated, Workflows, DevOps.

I. INTRODUCTION

Companies' perspectives on development and operation have evolved as a result of the growing need to supply quick product upgrades and extend the lifespan of software releases in order to boost customer satisfaction. Lack of cross-departmental connectivity is causing problems for IT organizations and slowing down the process of delivering finished software to clients. Operating the infrastructure used to be the responsibility of a whole team [1]. Ordering quotations, assembling the rack, cooling, and power, as well as installing the operating systems, programs, dependencies, and configuring the application files, were just a few of the tasks this operation team (OPS) had to do in order to have additional infrastructure resources [2, 3]. Every time they needed to ascend and get more computers, they had to go through this procedure. Simultaneously, another team (Dev) created and constructed the software program, which was then sent to the OPS team for installation in the constructed infrastructure.

Many sysadmins are embracing the Infrastructure as Code method to deliver such resources using DevOps technologies like Ansible, Chef, Puppet, and Terraform, rather than racking computers and connecting in network connections [3, 4]. Developers and operations teams are thus concentrating on developing and running code to specify, implement, and maintain the cloud infrastructure. To address concerns about privacy, security, performance, fault tolerance, and latency, businesses are shifting to the cloud [4, 5]. In order to improve fault tolerance and enable a more seamless recovery from cloud provider disruptions, businesses are using numerous clouds. To describe many clouds in this context, the industry came up with the term "multi-cloud." However, since each provider has its own tools, procedures, and APIs, multi-cloud deployments introduce an additional degree of complexity [5, 6].
Code for Infrastructure (IaC) The process of building infrastructure is dynamic and often calls for constant adjustments and enhancements in areas like maintainability, fault tolerance, scalability, and performance. Delays and a reduction in organizational agility resulted from the manual and laborious process of building and deploying infrastructure components in conventional setups [6, 7]. Infrastructure components are today seen as little more than software constructs, a code that can be shared by other teams, thanks to the rise of IaC. Using a set of principles, techniques, and tools to optimize software delivery time, Infrastructure as Code (IaC) is an approach to infrastructure automation based on development and operations (DevOps) practices that encourage ongoing collaboration between developers and operations staff [5]. An IaC solution adheres to the following guidelines, to sum up this concept:

- The widely accepted idea of version control states that each release is associated with a source code build that is kept as a versioned artefact in the environment. A similar idea is used in IaC to use version-control commits in the source code repository to manage the infrastructure and modifications. This makes it possible to track modifications made to the infrastructure specification, including who made them, what changed, and [5, 6]. This is essential when reverting to an earlier version of the code to debug a problem.
- The ability of IaC to consistently provide the same environment and related properties (as specified in the version-controlled system) each time it is called upon is known as predictability [5].
- Consistency guarantees a comparable environment across many occurrences of the same baseline code. By doing this, while creating complicated infrastructure entities by hand, inconsistencies and deviations from configuration are avoided [4, 5].
- A repeatable solution is one that, depending on the input, consistently yields the same outcomes.
- The term "composability" describes a service that may be utilized to create complex application systems and is maintained in an abstract and modular manner [6]. Instead of worrying about the intricate technicalities and the logic involved in provisioning, this functionality enables customers to concentrate on the goal application development [6].

1.1 IaC Concepts

- All infrastructure resources and components, including directories, packages, user accounts, utilities, and settings, are specified as code [6, 7].
- Each IaC tool has a word that refers to its source code, such as manifests, templates, playbooks, recipes, and cookbooks [7, 8].
- IaC adds features including greater dependability, notable speed gains, and repeatability.
- IaC provides build consistency.

For instance, spinning up several environments (such as development, QA, staging, and production) from the same codebase guarantees that little configuration drift is introduced between the environments, preserving the domain's sanity.

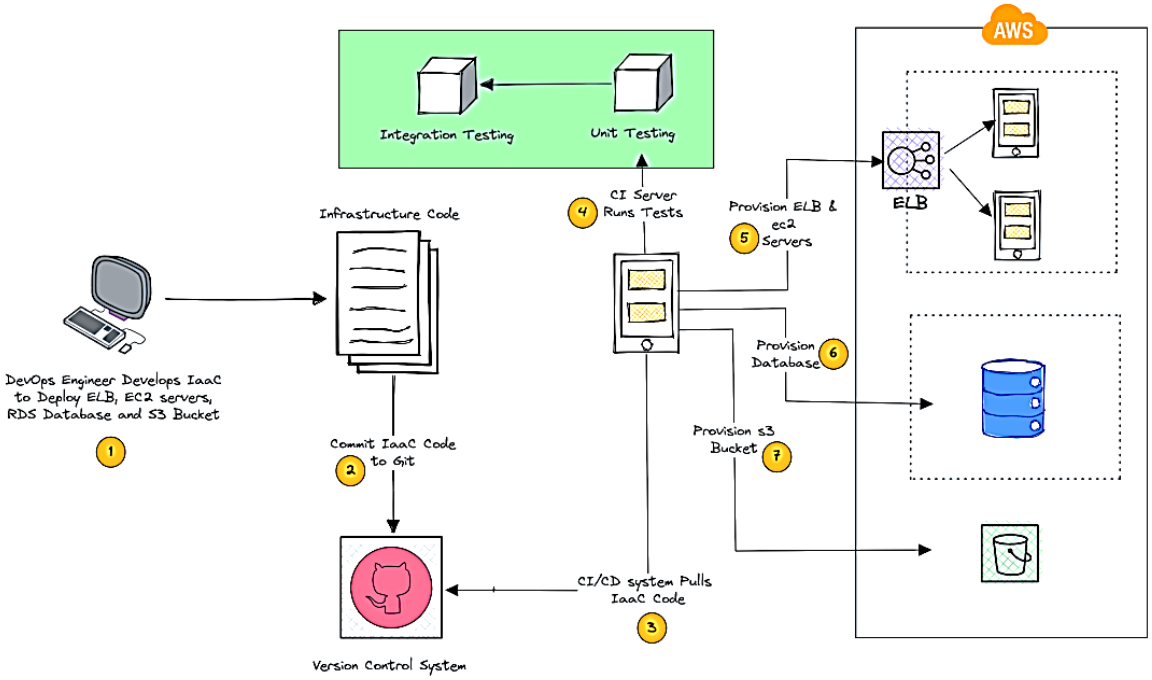

Fig. 1 Provisioning Infrastructure vs. Configuration Management. [8]

In the cutthroat and fast-paced commercial environment of today, cloud computing is increasingly commonplace. Businesses and developers stand to benefit greatly from faster runtimes, scalability on demand, code integrity, and improved software and application performance when paired with the DevOps culture [9].

Cloud computing is here to stay because it is expected to revolutionize the deployment and management of IT by lowering implementation, maintenance, and complexity costs while boosting innovation, speeding up time to market, and enabling the scalability of high-performance infrastructures and applications as needed [9, 10]. As cloud computing adaptation progressed, a number of service models emerged that fall into three main categories:

- Infrastructure as a Service, or IaaS, is a broad term that includes infrastructure-related features including networks, computation, raw storage space, virtual machines, and [10]. Sent to the client and used by them.
- PaaS Often referred to as Platform as a Service, it offers a toolbox and a variety of programming languages.
- Software as a Service, or SaaS, is a cloud-based platform that offers the end user interface of a whole program [11].

1.2 DevOps

A cultural movement that is bridging the gap between both development and operations teams is symbolized by DevOps, which is a mix of development and operations [12]. Software developers and IT teams may create, test, and deliver software more quickly and reliably by automating and integrating processes via a set of methods called DevOps. To accomplish automation, a variety of devOps tools are used.

1.3 IaC

The method of automatically configuring system requirements and provisioning both local and remote instances is known as infrastructure as code (IaC) [13]. Continuous deployment is automated with IaC scripts. Well-known IaC technologies that provide tools for autonomously configuring and provisioning software deployment infrastructure utilizing cloud instances include Terraform, Ansible, [12,14], and others. With an emphasis on cloud computing's IaaS service, which can be deployed on any deployment model with the use of appropriate tools retrieved from the devOps ecosystem, IaC and devOps are closely associated. It was useful to re-deploy the infrastructures [15].

- It became simple to share and use infrastructure and code.
- Infrastructure becomes scalable, allowing resources to be readily up- and down-scaled.
- We can lower our infrastructure expenses by implementing a pay-per-use strategy [15].

1.3.1 IaC in Public Cloud

AWS, Azure, Google Cloud, and other public cloud services are used in conjunction with devOps technologies to accomplish the goal of infrastructure automation [15, 16]. The tool types listed below comprise a typical approach for implementing IaC structure:

- Tools from Art Factory to save your information;
- Communication tools for collaboration; remote machine configuration via configuration management;
- Tools for continuous integration to include the code [16],
- Tools for managing source code control to oversee version control;

1.3.2 IaC in Private Cloud

The term "private cloud" refers to the on-premises deployment of resources via the use of virtualization and resource management technologies. Although on-premises deployment lacks many of the advantages of cloud computing, it is sometimes chosen because it offers dedicated resources. This deployment paradigm, which uses virtualization and application management technologies to attempt to improve resource usage, is often the same as older IT infrastructure [17]. There are two main types of private cloud [17].

- Private Platform as a Service (PaaS) and Private Infrastructure as a Service (PaaS) are often referred to as Enterprise PaaS.
- It is possible to deploy Private PaaS and Enterprise PaaS in both internal and external settings. It may be set up, for instance, on AWS and in a customer's local data center using private IaaS, virtualized, and bare metal. Private IaaS is not necessary for private PaaS or enterprise PaaS.

II. RELATED WORK

(Garg, A. 2024) In dispersed systems, manual infrastructure provisioning restricted stability, scalability, and flexibility. Implementing infrastructure changes now takes just a few minutes thanks to the adoption of Infrastructure as Code (IaC) concepts, which have significantly increased deployment agility [18]. Dynamic scaling was made possible by automation, which improved cost and resource efficiency [19]. IaC reduced the dangers of configuration drift and security flaws by standardized configurations as code, which increased consistency and dependability.

(Chijioke-Uche, J. 2022) Without automation and a versioning plan, implementing hybrid and multicloud infrastructure might have a detrimental effect on an organization's productivity [19, 20]. Because implementation solutions, such as automated and DevOps methods, provide resources for recurring infrastructure installation use cases, organization executives must make sure that infrastructures are deployed utilizing the infrastructure as code (IaC) approach.

(Ketonen, T. 2024) The area of cloud-to-cloud migration, which is made possible by the Infrastructure as Code (IaC) approach, is examined in this thesis. Organizations want to reduce operating costs, improve system flexibility, and adapt to shifting market needs as the cloud computing environment

continues to change quickly. The capacity to effectively switch between cloud providers becomes crucial in this situation [21].

(Santoro, F. 2024) Modern technological and communications infrastructures have become increasingly software-based systems in recent years due to the development of technologies like Infrastructure as Code (IaC) and Policy as Code (PaC) [22]. Faster deployment, scalability, and easier network administration have all been made possible by this progression. Furthermore, the environment has become more varied due to the increasing number of Infrastructure as Code (IaC)-based solutions, thus before provisioning and deploying the infrastructure, each business must choose the best option for its requirements while maintaining policy compliance.

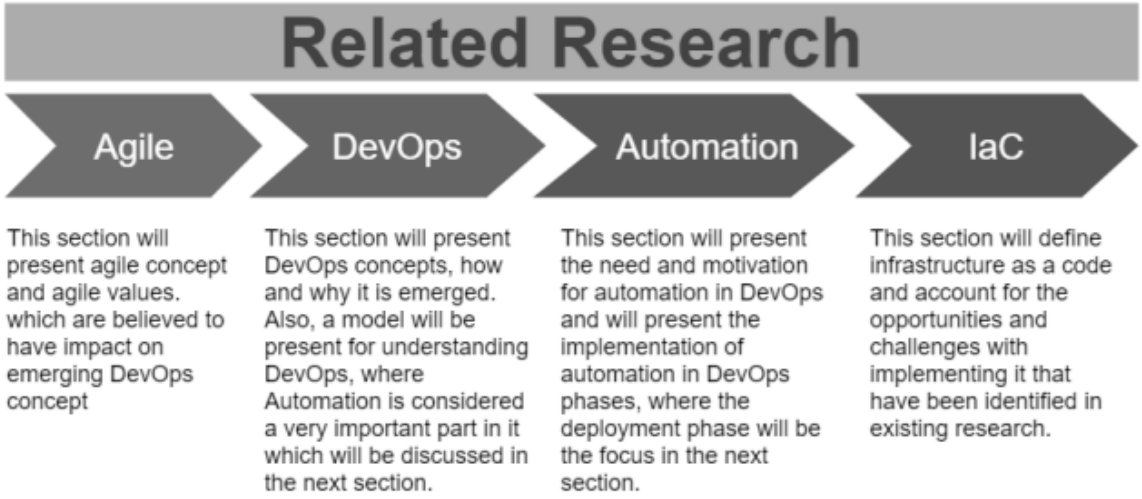

Fig. 1 A model to describe the flow of the related research [13].

III. RESEARCH METHODOLOGY

The procedures and methods by which the study will be carried out appropriately in a given field, where the researcher employs its instruments and guidelines to gather data, analyze it, and resolve an issue in that field, are known as the research methodology [13, 15]. The purpose of this paper's exploratory single case study is to identify and comprehend certain ideas, standards, or values that are influenced by the research participants' frame of reference. The case study was conducted at Rebtel Networks AB, [18], a technology firm that provides goods and services to migrants with the goal of empowering them. Only participants from the firm participated in the data gathering.

Table 1 Phases of data analysis.

Phase	Description of the process
1- The data should be known to the researcher	The participants' audio data was transcribed, read aloud many times in an active manner to look for trends and meanings, and notes were taken.
2- Generating initial codes	I utilized line-by-line coding to identify the most intriguing and significant aspects of the data while keeping my research topics in mind.
3- Searching for themes	I compiled and merged various codes into code groups by analyzing the resulting codes. I then classified these code groups into code categories after looking for higher level categories. The coding categories were then used to generate themes. I utilized tools like Atlas Cloud and mental maps to aid me in this process.
4- Reviewing themes	I looked to see whether the themes could be combined, divided into distinct themes, or removed. Two stages of refining were involved:

	a review of the themes using both the whole data set and coded data extracts. In this stage, creating theme maps was also helpful.
5- Defining and naming themes	In order to provide a precise description and name for every topic, the substance of each subject was ascertained, as well as what it captured in connection to the whole narrative. Four major topics were so selected to be presented in the results section.
6- Producing the report	In order to write the discussion and finish the final report, I conducted a final analysis of the chosen themes and connected their results to the research questions and associated activities.

IV. RESULT

The results of the study design will be shown in the parts that follow. The outcome, which comes from in-depth interviews, will be shown in four distinct themes that are created throughout the analytical process [18]. These topics include experiences in the Iron Age, the Cloud Age, Agile and DevOps, and the potential and difficulties of adopting IaC. DevOps, infrastructure management both beforehand and following the cloud, and infrastructure as a code are all familiar to the respondents in Table 1. Terraform (https://www.terraform.io/) is a technology for infrastructure as a code, and both Preto and Maharjan use the Microsoft Azure cloud for their infrastructure. For infrastructure as a code, Hammami uses the Cloud Deployment Manager and the Google Cloud Platform. Andrade uses Terraform and CloudFormation be (https://aws.amazon.com/cloudformation/) as tools for Infrastructure as a Code and Amazon Web Services (AWS) for Web infrastructure.

4.1 Agile and DevOps

This section demonstrates how Agile and DevOps are related. Understanding how these ideas impact IaC later on will be made easier with the aid of this section [16]. According to the answers, the business is reasonably agile. Preto said that since the business creates its own goods, implementing agile was simple. However, based on his experience, he believes that agile is still difficult to implement in outsourced projects since you have a fourteen-closed estimate to create something that satisfies certain specifications [18]. Consequently, additional work must be dedicated to precisely determining what has to be done, when, and how [16].

> *"We refine our goals, make incremental changes to the product, and release it out every few weeks, depending on our sprint time. It's quite effective [agile]."- Petro*

4.2 Experiences during the Iron Age

The results of the participant's experience dealing with physical infrastructure management are shown in this section [16, 18]. According to the findings, the respondents had several difficulties and issues while dealing with physical infrastructure management prior to the cloud.

> *"Due to a lack of tools, we were unable to remotely monitor and debug very successfully. Instead, we had to personally verify when anything went wrong. It was difficult to manage the infrastructure hardware alone as you had to consider and understand every piece of hardware, including the disk, CPU, memory, etc." - Hammami*

4.3 Experiences during the cloud age

The results from the participant's experience managing cloud infrastructure are shown in this section [19, 20]. As was already noted, there are several restrictions and difficulties with the physical infrastructure.

> *"It becomes easy, affordable, and appealing with the cloud's abstraction of the hardware layer. Before the cloud, it would have been almost impossible for us*

to have the same application distributed across many nations and continents." – Andrade

"You can deploy simply your code and further abstract the hardware to become serverless. That way, all you have to consider is your application's actual code." - Maharjan

4.4 The challenges and opportunities with implementing IaC

One intriguing finding from the study's findings is how respondents saw IaC's chances and problems from various angles. I will outline the potential and difficulties in this part [20, 21] based on the most frequently mentioned factors by the respondents.

- Adaptability and resource conservation the participants' most frequent answers were "flexibility" and "saving time and resources."
- The uniformity that IaC can attain was another topic that all of the responders brought up.
- The respondents found that establishing infrastructure resources manually using the GUI presented significant challenges due to human error and non-homogeneous naming [22].
- The process of versioning Versioning was seen by both Hammami and Preto as the third crucial component of IaC. According to a number of responders, versioning the infrastructure deployment makes it clear who performed what, where, and when [22].
- The capacity to repeat Every responder said that a major benefit they could have before was the ability to quickly and consistently build several environments (development, testing, and production) using the same code [23].
- IaC may be very important in infrastructure risk management, according to a number of responders [23]. IaC, according to Andrade and Maharjan, is a highly effective and affordable method of providing a disaster recovery solution for the infrastructure environment.
- Andrade and Maharjan said that having a continuous integration and continuous delivery pipeline is a crucial DevOps approach [24, 25].
- Records Preto contends that the code is the ideal location for documentation. The other respondents agreed with Preto, stating that the IaC code is really straightforward and simple to read and comprehend [21].
- Proficiency As previously said, manually generating infrastructure resources may be a highly annoying task, particularly when certain resources need to be recreated because they were misconfigured.

V. DISCUSSION

Gaining a deeper understanding of the infrastructure as a code (IaC) idea and identifying the potential and obstacles associated with its implementation are the goals of this research. I conducted a case study on an IaC deployment with five individuals in a single organization [22]. Understanding and gaining insight into the potential and problems related to infrastructure as a service (IaC) may be achieved by documenting the technical and managerial treatment of infrastructure in the Iron Age and the Cloud Age, from conventional management through agile to DevOps [21, 23]. Because manual chores are far more likely to be forgotten, neglected, or performed improperly, the research discovered that IaC can resolve human errors and non-homogeneous naming for infrastructure resources, which is a significant difficulty when done manually [25]. Evidence of this has also been found in earlier studies, such as the fact that IaC has a significant chance to replace manual setup and error-prone shell scripts with dependable and resilient code to provide a consistent representation of the established infrastructure.

VI. CONCLUSION

IaC refers to the method and technique of using code to represent the underlying resources that underpin our company. Teams of any size may utilize it, regardless of whether they use their own data center or

cloud providers like AWS, GCP, or Azure. The primary distinction between private and public clouds is that in the former, you are directly in charge of running the complete infrastructure, administering the machines, and invoicing for the services and machine payments. You don't have direct access to or control over the computers while using a public cloud. A private cloud prioritizes all types of IT upgrades, patches, and fixes in terms of security. IT can even manage the perimeters thanks to it. On the other hand, a public cloud safeguards data by controlling security at the software and hardware levels.

The goal of the research was to demonstrate how IT infrastructure is managed in both the Iron Age and the Cloud Age, as well as how ideas and technologies like agile, DevOps, and IaC impact IT infrastructure management procedures. However,

"Because IaC has not been widely adopted by IT organizations, empirical studies pertaining to their experiences and challenges have not been reported."

For businesses to have a successful continuous deployment, it is crucial to research and determine the consequences of the IaC implementation problems. Additionally, it's critical to discuss the advantages of using IaC, which may help businesses release software more quickly, create a stable and reliable environment, and increase customer satisfaction.

VII. REFERENCES

[1] H. Kharche, D. S. Chouhan, “Building Trust in Cloud Using Public Key Infrastructure”, International Journal of Advanced Computer Science and Applications, vol. 3, no. 3, (2012).

[2] K. Morris, Infrastructure as code: managing servers in the cloud,” O’Reilly Media, Inc.”, 2016.

[3] De Cesare, S., Lycett, M., Macredie, R.D., Patel, C., and Paul, R., 2010. Examining perceptions of agility in software development practice. Communications of the ACM, 53(6), pp.126-130. Ebert, C., Gallardo, G., Hernantes, J., and Serrano, N., 2016. DevOps. Ieee Software, 33(3), pp.94-100.

[4] Erich, F., Amrit, C. and Daneva, M., 2014, December. A mapping study on cooperation between information system development and operations. In International Conference on Product-Focused Software Process Improvement (pp. 277-280). Springer.

[5] Cham. Hermanns, J., and Steffens, A., 2015. The current state of ‘Infrastructure as Code ‘and how it changes the software development process. Full-scale Software Engineering, 19.

[6] Frey, C. B. (2019). The Technology Trap. Princeton University Press. Mantoux, P. (1983). The Industrial Revolution in the Eighteenth Century. The University of Chicago Press. Chicago.

[7] Miah, M. S., Pasupuleti, M. B., & Adusumalli, H. P. (2021). The Nexus between the Machine Learning Techniques and Software Project Estimation. Global Disclosure of Economics and Business, 10(1), 37-44.

[8] Pasupuleti, M. B. (2016). Data Scientist Careers: Applied Orientation for the Beginners. Global Disclosure of Economics and Business, 5(2), 125-132.

[9] Abbas, S. I., & Garg, A. (2024, June). Integrating Emerging Technologies with Infrastructure as Code in Distributed Environments. In 2024 3rd International Conference on Applied Artificial Intelligence and Computing (ICAAIC) (pp. 1138-1144). IEEE.

[10] Chijioke-Uche, J. (2022). Infrastructure as Code Strategies and Benefits in Cloud Computing (Doctoral dissertation, Walden University).

[11] Ketonen, T. (2024). Strategies and challenges in cloud-to-cloud migration using infrastructure as code.

[12] Santoro, F. (2024). Validation and Verification of Infrastructure as Code (Doctoral dissertation, Politecnico di Torino).

[13] B. Adams and J. Hall, "Monitoring and Logging in IaC Deployments: Techniques and Tools," IEEE Cloud Computing, vol. 7, no. 4, pp. 58-67, Oct.-Dec. 2020.

[14] R. Singh, "IaC in Multi-Cloud Environments: Strategies and Challenges," IEEE Transactions on Cloud Computing, vol. 9, no. 2, pp. 487-498, April-June 2021.

[15] A. Wilson, "The Role of IaC in Modern Platform Engineering," IEEE Software, vol. 38, no. 3, pp. 40-49, May-June 2021.

[16] J. Kim and R. Patel, "Future Trends in IaC: Innovations and Predictions," IEEE Cloud Computing, vol. 8, no. 1, pp. 20-29, Jan.-March 2021.

[17] V. Shvetcova, O. Borisenko, and M. Polischuk, "Domain-Specific Language for Infrastructure as Code," Proc. - 2019 Ivannikov Meml. Work. IVMEM 2019, pp. 39–45, 2019,

[18] K. Morris, Infrastructure as Code: Managing Servers in the Cloud, 1st ed. Newton, MA, USA: O´Reilly Media, 2016.

[19] G. Kumar and P. K. Bhatia, "Comparative analysis of software engineering models from traditional to modern methodologies," Int. Conf. Adv. Comput. Commun. Technol. ACCT, pp. 189–196, 2014.

[20] Okazaki, E.; Yao, R.; Sirven, J.; Crepeau, A.; Noe, K.; Drazkowski, J.; Hoerth, M.; Salinas, E.; Csernak, L.; Mehta, N. Usage of EpiFinder clinical decision support in the assessment of epilepsy. Epilepsy Behav. 2018, 82, 140–143.

[21] Plebani, M.; Aita, A.; Padoan, A.; Sciacovelli, L. Decision support and patient safety. Clin. Lab. Med. 2019, 39, 231–244.

[22] Hassol, A.; Deitz, D.; Goldberg, H.; Honicker, M.; Younkin, J.; Chaundy, K.; Walker, J.M.; Cummins, M.R. Health information exchange. CIN Comput. Inform. Nurs. 2016, 34, 145–150.

[23] Benson, T.; Grieve, G. Why interoperability is hard. In Principles of Health Interoperability; Health Information Technology Standards; Springer: Cham, Switzerland, 2021.

[24] Everson, J.; Patel, V.; Adler-Milstein, J. Information blocking remains prevalent at the start of 21st century cures act: Results from a survey of health information exchange organizations. J. Am. Med. Inform. Assoc. 2021, ocaa323.

[25] Vest, J.R.; Gamm, L.D. Health information exchange: Persistent challenges and new strategies. J. Am. Med Inform. Assoc. 2010, 17, 288–294.