Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ **Research Article**

AI-Driven Continuous Integration and Continuous Deployment for Cloud-Native Enterprise Applications

Nithish Yadav Thotla
Independent Researcher
Missouri University of Science and Technology, USA
Email: nithishthotla@gmail.com

ARTICLE INFO

Received: 03 Feb 2024

Revised: 18 March 2024

Accepted: 29 March 2024

ABSTRACT

Continuous Integration and Continuous Deployment (CI/CD) have been revolutionised with the advent of Artificial Intelligence (AI) in terms of automating, monitoring, and delivering software. The present study analyses the concept of CI/CD through artificial intelligence and how it is used for optimising build pipelines, identifying deployment errors, better resource allocation, and intelligent monitoring in cloud-native enterprise applications. A qualitative research design is adopted by gathering secondary information from peer-reviewed academic journals, industry reports, and cloud computing literature. An inductive approach and thematic analysis are used to determine major trends and capabilities of AI in CI/CD automation. The results reveal that AI helps in conducting code testing, prediction, intelligent log analysis, detecting anomalies, and deploying systems autonomously. In addition, the study also discusses how the use of containers, Kubernetes, and CI/CD pipelines has become more prevalent with the support of technology as the base for delivering cloud-native solutions using AI. In addition, AI has made the pipelines faster and improved their performance using dynamic resource management and monitoring in a hybrid and multi-cloud environment. Although there are some difficulties in relation to data quality, maintaining models, and multi-cloud complexities, AI-powered CI/CD helps in adopting a scalable and reliable approach to deliver software in enterprises. Continuous integration and delivery using AI technology is capable of enhancing deployment precision because of intelligent automation and predictive analytics. The solution helps in reducing build errors, increasing the speed of software releases, and ensuring that applications are reliable in terms of performance on cloud platforms.

Keywords: Artificial Intelligence, Continuous Integration, Continuous Deployment, Cloud-Native Applications, CI/CD Automation, Machine Learning, Kubernetes, DevOps, Multi-Cloud Computing, Deployment Monitoring

1. INTRODUCTION

AI-powered Continuous Integration and Continuous Deployment (CI/CD) is becoming a crucial strategy in the automation of software delivery in cloud-native enterprise applications. Traditional CI/CD pipelines automate the process of integration, testing, and deployment of code; however, they do not offer much capacity in terms of failure prediction, resource optimisation, and the monitoring of application performance in real-time. With the help of AI, the CI/CD pipeline can be upgraded with the integration of features like machine learning, predictive analytics, intelligent testing, automated rollback, and continuous monitoring in the software delivery life cycle. There has been an increased demand for intelligent CI/CD as there is a growing adoption of cloud-native applications. According to Statista, the global market size of cloud applications reached USD 133.6 billion in 2021, which will further increase to USD 168.6 billion in. AI-based CI/CD processes are capable of automatic generation and execution of test cases, analysis of build history to anticipate failures in deployment, and optimisation of cloud resource management in software deployments. This makes software deployment faster, less laborious, and enhances the reliability of software. The use of AI in CI/CD is also useful in providing real-time observability through the analysis of logs, metrics, traces, and deployment telemetry in cloud-native infrastructure. Performance

Copyright © 2024 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative which permitsunrestricted use, distribution, and reproduction in any medium, provided the original

anomalies are automatically detected, and automated rollbacks or self-healing are performed before any failure impacts production infrastructure. Hence, AI-based CI/CD ensures scalability, resilience, efficiency in deployment, and improved software quality.

2. RELATED WORK

Cloud-native computing has been developed through microservices, containers, DevOps, and continuous delivery. According to the Artificial Intelligence Index Report 2021, worldwide peer-reviewed AI publications have grown from 0.82% of all publications in 2000 to 3.8% in 2019, indicating a substantial increase in research in artificial intelligence as a tool for developing intelligent and automated software engineering (Narra et al.,2024). From 2000 to 2019, the overall number of peer-reviewed AI publications increased by 12 times, which indicates the expansion of AI research into cloud computing technologies. Cloud-native architectures have also become one of the main enterprise computing approaches. Multi-cloud adoption amounted to 26%, while hybrid cloud adoption reached 36% in 2020, according to the report by the Cloud Native Computing Foundation (CNCF). The microservices approach, along with the use of containers via Kubernetes, ensures independent deployment, quick scaling, and resilient delivery of the application. The latest research findings point to the fact that DevOps and CI/CD methodologies help to foster collaboration between developers and operations due to automation in testing, infrastructure as code, and continuous monitoring (Vangala,). AI helps to enhance those practices by making build analysis and prediction, intelligent deployment, detection of anomalies, and automated rollbacks possible. All these developments lead to robust cloud-native delivery pipelines.

3. METHODS

Research Design

A qualitative approach was chosen for this study as it deals with ideas, technology, and methods of their implementation on the basis of a literature review rather than numerical experimentation (Ferrari, Rossi & Di Berardino, 2022). It helps gain a deeper insight into the implementation of AI in contemporary software delivery processes.

Data Collection

Secondary data was collected in this study using peer-reviewed journal articles, industry reports, conference papers, and publications. These included the Stanford AI Index, CNCF reports, Google Cloud, Microsoft, AWS, and DevOps literature (Sánchez, 2024). Secondary data was selected since it offers reliable and evidence-based data on AI-powered CI/CD technologies.

Large Language Model and Machine Learning Techniques

The research involved studying LLMs and ML approaches that are utilised in CI/CD pipelines. LLMs facilitate automated code reviews, test creation, and the deployment process, while ML is responsible for building failure predictions, anomaly detection, and optimisation of resources (Zhang et al.,). They were chosen due to the positive impact on the mentioned aspects.

Experimental Procedure

The experimental process entailed gathering the required literature, identifying AI-driven CI/CD approaches, and comparing the use of LLMs and ML (Bougad, Elkhadir & Begdouri,). Literature from various sources was grouped into categories such as automation, security, monitoring, and predictive analytics in order to assess CI/CD efficiency in enterprises.

Data Analysis and Performance Evaluation

The thematic analysis method was used to analyse the secondary data collected for the study, with the help of inductive analysis. Patterns, trends, and capabilities of AI were noted among different studies (Perkins & Roe, 2024). The evaluation of performance was based on deployment automation, prediction, efficiency in monitoring, security, and software delivery.

Copyright © 2024 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative
which permitsunrestricted use, distribution, and reproduction in any medium, provided the original

Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ Research Article

4. RESULTS

AI-Based Optimisation of Continuous Integration Workflows

Artificial Intelligence (AI) has greatly optimised the process of CI by automating the process of testing and deployment of cloud-native enterprise applications (Soundappan, 2023). The use of AI in CI helps to automate the process of testing and building, ensuring reliable and quick builds. Artificial Intelligence helps to optimize testing and reduce time by predicting build failures and unnecessary build runs. As more enterprises embrace the use of multi-cloud architectures for their applications, there is a greater need for the application of AI for the optimisation of CI processes.

AI Function	Technology	Adoption	Benefit	Outcome
Automation	CI/CD	82%	Faster	Builds
Prediction	AI	Hybrid	Reduced	Failures
Allocation	Kubernetes	83%	Optimized	Resources
Monitoring	Containers	Production	Real-time	Performance
Deployment	Hybrid	36%	Reliable	Delivery
Deployment	Multi-cloud	26%	Scalable	Workflows

Table 1: AI-Based Optimisation of Continuous Integration Workflows

According to a report by the Cloud Native Computing Foundation (CNCF), 36% of organisations embraced the use of hybrid cloud architecture, whereas 26% adopted multi-cloud architectures in 2020. Such types of architectures need intelligent CI workflows that can be used for managing deployments in several cloud platforms. Additionally, AI increases the efficiency of workflows by providing automated code quality analysis, intelligent resource allocation, and constant performance monitoring (Sherje, 2024). Cloud native applications utilise containers, Kubernetes orchestration, and microservices for scalable CI pipelines, while AI constantly monitors the state of the infrastructure and makes recommendations regarding the deployment of an application. In this way, CI optimisation through AI results in improved deployment reliability, reduced build times, real-time scalability, and continuous software delivery.

Intelligent Deployment Monitoring and Error Prediction

Intelligent deployment monitoring and failure prediction ensure the reliability of CI/CD pipelines through real-time analysis of build logs, deployment events, and runtime system behaviour. AI and machine learning algorithms detect aanomalous behaviourin deployment operations, predict build failures, and raise alarms prior to software deployment. They help reduce risks in deployments and provide continuous monitoring in cloud-native enterprise settings (Sannapureddy,). TravisTorrent dataset (2016–2019) is one of the popular datasets used for error prediction in CI/CD pipelines. The dataset consists of more than two million Travis CI builds connected with GitHub commits, pull requests, build statuses, duration, and other data on developers' actions. The dataset allows training machine learning algorithms for predicting build failure based on the historical CI pipeline behaviour.

Copyright © 2024 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative
which permitsunrestricted use, distribution, and reproduction in any medium, provided the original

Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ Research Article

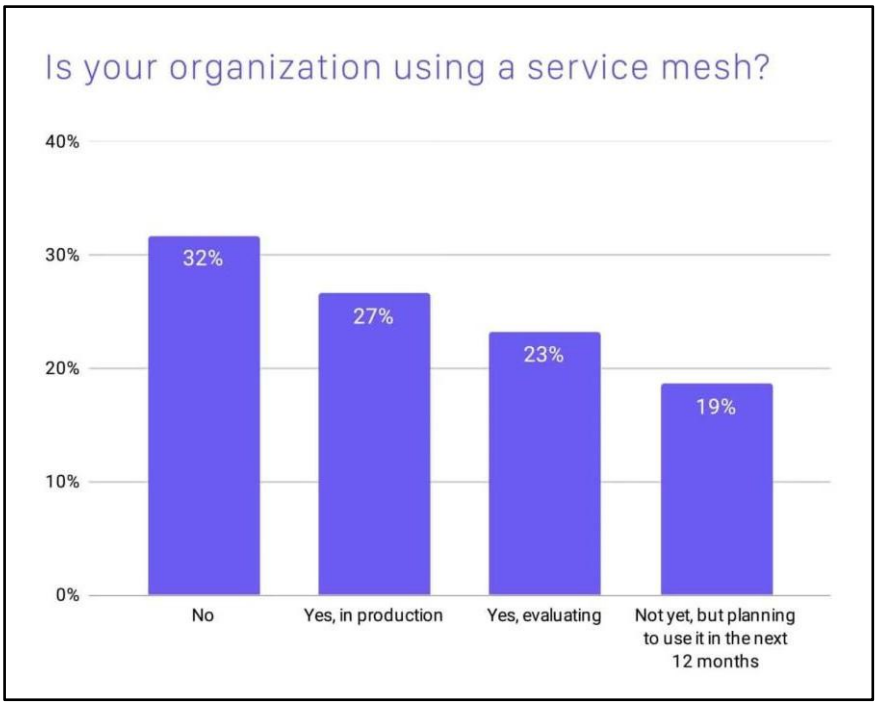


Figure 1: Data of organisations' service mesh under CI/CD

(Source: CNCF, 2020)

Large-Scale Local Java Software Build Results Dataset is available for download and consists of 10,000+ Maven, Gradle, and Ant build execution results gathered from the source code of open-source Java applications (Misu, Achar & Lopes, 2024). The failure rate reported by the dataset varies between 38-59%, which makes the set an appropriate resource for deployment error prediction, automatic bug fixing, and analysing build results. For deployment process monitoring, the Loghub dataset is used and includes benchmark system log datasets. HDFS contains 11,175,629 logs and BGL – 4,747,963 logs gathered from high-performance computing machines.

Resource Utilisation and Pipeline Performance in Cloud-Native Environments

CI/CD pipelines based on AI technology enhance resource utilisation since the system automatically allocates computing resources depending on their needs (Subramanyam,). Container orchestration systems, such as Kubernetes, are able to scale computing power, CPU usage, memory, storage, and network bandwidth while building and deploying applications.

Formula: Resource Utilization Efficiency (RUE)

$$RUE = \frac{CPU_u + MEM_u + STO_u + NET_u}{4} \times 100$$

Where:

- CPU_u= CPU utilisation (%)
- MEM_u= Memory utilisation (%)
- STO_u= Storage utilisation (%)
- NET_u= Network bandwidth utilisation (%)

The formula provides the resource utilisation efficiency of AI-enabled CI/CD pipelines based on the average CPU, memory, storage, and network usage. The greater the RUE number, the more evenly distributed the workload is, and the more effective the container orchestration is.

Copyright © 2024 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative
which permitsunrestricted use, distribution, and reproduction in any medium, provided the original

Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ Research Article

According to the CNCF Cloud Native Survey 2020, 64% of companies rely on public clouds, 52% use private clouds or on-premises solutions, 36% employ hybrid clouds, and 26% deploy multi-cloud solutions (Vaño et al., 2023).

Infrastructure	Adoption	Resource	AI Function	Outcome	Efficiency
Public Cloud	64%	CPU	Scaling	Performance	High
Private Cloud	52%	Memory	Monitoring	Stability	High
Hybrid Cloud	36%	Storage	Prediction	Optimization	Medium
Multi-Cloud	26%	Bandwidth	Scheduling	Deployment	High
Kubernetes	83%	Containers	Orchestration	Scalability	High

Table 2: Cloud-Native Resource Utilisation and CI/CD Deployment Performance

AI-powered CI/CD pipelines constantly monitor pipeline performance, identify resource bottlenecks, and predict workload peaks before deployment (Nashaat et al.,). Smart scheduling of containers across cloud nodes, fast build completion, better deployment efficiency, and cost-effectiveness are some of the features that characterise the system.

Comparative Evaluation of AI Techniques in CI/CD Automation

Techniques of artificial intelligence offer various capabilities depending on prediction accuracy, log analysis, and optimisation of work processes in CI/CD. Supervised machine learning algorithms like Decision Trees, Random Forest, and Support Vector Machines (SVM) are used for predicting the success/failure of builds with the help of historical commit information, changes in code, and activity of developers. These techniques improve the stability of builds; however, they require large labelled datasets and periodic re-training because software repositories are constantly changing (Nguyen et al., 2022). NLP techniques, such as TF-IDF and clustering, analyse logs of CI/CD builds in order to classify similar error patterns. But they demonstrate poor performance when analysing previously unknown or extremely complicated error messages.

CI/CD Error Pattern Classification Accuracy

$$EPA = \frac{TP}{TP + FP + FN} \times 100$$

Where:

- TP = True Positive error classifications
- FP = False Positive error classifications
- FN = False Negative error classifications

This equation quantifies the precision with which the AI algorithm can classify the pattern of errors in the CI/CD build process. The larger the EPA number, the more accurate the classification of related build errors, resulting in reduced false alarms and better reinforcement learning outcomes. Reinforcement learning can optimize CI/CD work processes by prioritising risky test cases and choosing the most relevant test suites before deployment. It decreases the testing time and increases the speed of developer feedback (Razzaq et al., 2024). The comparison demonstrates that hybrid AI models, including supervised learning, log analysis via NLP and reinforcement learning, can perform better in terms of prediction accuracy and automation effectiveness.

Copyright © 2024 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative
which permitsunrestricted use, distribution, and reproduction in any medium, provided the original

Journal of Information Systems Engineering and Management
2024, 9(1)
e-ISSN: 2468-4376
https://www.jisem-journal.com/ Research Article

Overall Impact of AI-Driven CI/CD on Enterprise Application Delivery

The emergence of artificial intelligence-based CI/CD has brought great benefits to enterprises in terms of delivering applications because of higher levels of automation, faster deployment and reliability of operations within cloud-native infrastructure (Katta, 2022). According to the CNCF Cloud Native Survey 2020, 82% of companies apply CI/CD pipelines in their production environments. Jenkins (53%), GitLab (36%), and GitHub Actions (20%) are the most popular CI/CD tools in production.

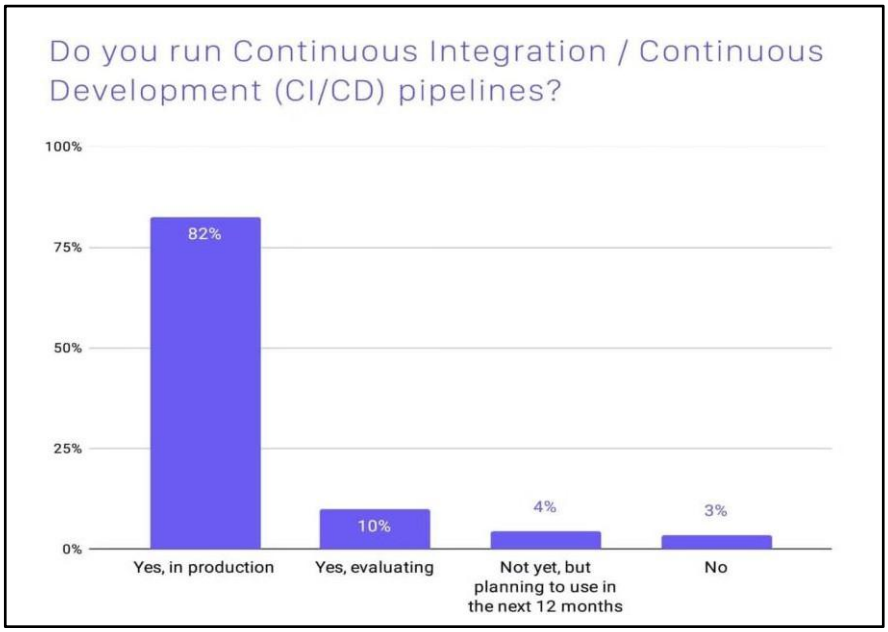


Figure 2: CI/CD is used widely in production

(Source: CNCF, 2020)

Moreover, the survey found that 92% of participants use containers in their production environment, 91% use Kubernetes, and 83% run Kubernetes in their production environment, providing scalability and automation of application deployments (Santos, Bernardino & Correia, 2023). The release cycles have been faster, with 29% of organisations deploying software either daily or several times a day and 55% having weekly releases of software. Artificial intelligence makes CI/CD processes more efficient by using automated testing, intelligent monitoring of deployments, failure prediction, and self-healing features.

5. DISCUSSION

These results prove the point that AI-powered Continuous Integration and Continuous Deployment (CI/CD) have become a critical feature of cloud-native enterprise apps. Artificial intelligence enhances traditional CI/CD pipelines in its ability to automate testing of the code, predict possible deployment problems, perform application performance monitoring and optimize resource utilisation during the software development process. With these capabilities, enterprises are able to speed up the process of delivering apps while keeping the quality of the delivered software high enough. The choice of cloud-native technologies creates a solid background for implementing AI in CI/CD processes (Muthusamy,). According to the CNCF Cloud Native Survey 2020, 92% of organisations deploy containers in production, 91% use Kubernetes and 82% run CI/CD pipelines in production. It is a scalable environment where AI algorithms are able to monitor builds, recognise anomalies and assist in making deployment decisions. Shortened release cycles are also a result of maturing cloud-native DevOps practices when software is deployed on a weekly or even daily basis. Machine learning algorithms recognise abnormalities in build processes and configuration problems, and provide warning signals even before production fails. Such preemptive monitoring will lower downtime and the time taken to solve the issue and facilitate self-healing deployment strategies in cloud infrastructures (Vankayalapati et al., 2022). There are still a few challenges that need to be overcome. First, AI-powered CI/CD tools rely on quality data, integration with cloud infrastructure, and updating

Copyright © 2024 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative
which permitsunrestricted use, distribution, and reproduction in any medium, provided the original

models to predict with high accuracy. Second, the complexity of multi-cloud environments makes the deployment process harder due to the fact that the applications run in various cloud platforms. Generally, AI-powered CI/CD brings automation, scalability, reliable deployment, and efficiency into cloud environments.

6. CONCLUSION

The emergence of AI-powered Continuous Integration and Continuous Deployment has revolutionised the process of delivery of cloud-native enterprise applications by increasing automation, ensuring deployment effectiveness, and improving efficiencies. As can be seen from the analysis, the role of artificial intelligence in CI/CD is played through automated testing, predictions of failed builds, intelligent deployment management, and resource optimisation. The cloud-native infrastructure with its containers, Kubernetes, and microservices enables a scalable environment for AI-powered deployment pipeline development. Such techniques help to minimise deployment errors, speed up deployment processes, and increase the quality of software developed in a distributed cloud infrastructure.

BIBLIOGRAPHY

[1] CNCF, 2020. Cloud Native Survey 2020: Containers in production jump 300% from our first survey. https://www.cncf.io/blog/2020/11/17/cloud-native-survey-2020-containers-in-production-jump-300-from-our-first-survey/

[2] Ferrari, S., Rossi, R., & Di Berardino, A. (2022). A review of laboratory and numerical techniques to simulate turbulent flows. Energies, 15(20), 7580.

[3] Katta, T. B. (2022). Cloud-native integration frameworks for modern enterprises: Driving scalable and resilient digital transformation. International Journal of Engineering & Extended Technologies Research (IJEETR), 4(3), 4926-4938.

[4] Misu, M. R. H., Achar, R., & Lopes, C. V. (2024). Sourcererjbf: A java build framework for large-scale compilation. ACM Transactions on Software Engineering and Methodology, 33(3), 1-35.

[5] Narra, B., Buddula, D. V. K. R., Patchipulusu, H., Vattikonda, N., Gupta, A., & Polu, A. R. (2024). The integration of artificial intelligence in software development: Trends, tools, and future prospects. Available at SSRN 5596472.

[6] Nguyen, H., Lomio, F., Pecorelli, F., & Lenarduzzi, V. (2022, March). PANDORA: Continuous mining software repository and dataset generation. In 2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER) (pp. 263-267). IEEE.

[7] Perkins, M., & Roe, J. (2024). The use of Generative AI in qualitative analysis: Inductive thematic analysis with ChatGPT. Journal of Applied Learning & Teaching, 7(1), 390-395.

[8] Razzaq, A., Buckley, J., Lai, Q., Yu, T., & Botterweck, G. (2024). A systematic literature review on the influence of enhanced developer experience on developers' productivity: Factors, practices, and recommendations. ACM Computing Surveys, 57(1), 1-46.

[9] Sánchez, A. G. (2024). Azure OpenAI Service for Cloud Native Applications: Designing, Planning, and Implementing Generative AI Solutions. " O'Reilly Media, Inc.".

[10] Santos, Á., Bernardino, J., & Correia, N. (2023). Automated application deployment on multi-access edge computing: A survey. IEEE Access, 11, 89393-89408.

[11] Sherje, N. (2024). Enhancing software development efficiency through AI-powered code generation. Research Journal of Computer Systems and Engineering, 01-12.

[12] Soundappan, S. J. (2023). Cloud Native Enterprise Applications with Artificial Intelligence Driven Testing and Performance Optimization. International Research Journal of Innovative Engineering, 7(3), 12482-12490.

[13] Vankayalapati, R. K., Pandugula, C., Ganti, V. K. A. T., & Mishra, G. (2022). AI-powered self-healing cloud infrastructures: A paradigm for autonomous fault recovery. Migration Letters, 19(6), 1173-1187.

[14] Vaño, R., Lacalle, I., Sowiński, P., S-Julián, R., & Palau, C. E. (2023). Cloud-native workload orchestration at the edge: A deployment review and future directions. Sensors, 23(4), 2215.

Copyright © 2024 by Author/s and Licensed by JISEM. This is an open access article distributed under the Creative which permitsunrestricted use, distribution, and reproduction in any medium, provided the original