European Journal of
Technology
(EJT)

Artificial Intelligence for Continuous Quality Assurance in Cloud-Native Applications: A Review

Usha Mohani kavirayani, Krishna Bhardwaj Mylavarapu, Jenitha Pilli, Prathik Kumar Jannu, Javed Ali Mohammad, Sri Harsha Panchali

AJP

European Journal of Technology
ISSN 2520-0712 (online)
Vol.7, Issue 5, pp 35-49, 2023

www.ajpojournals.org

Artificial Intelligence for Continuous Quality Assurance in Cloud-Native Applications: A Review

Usha Mohani kavirayani[1*], Krishna Bhardwaj Mylavarapu[2], Jenitha Pilli[3], Prathik Kumar Jannu[4], Javed Ali Mohammad[5], Sri Harsha Panchali[6]

[1]Kent State University, MS in Computer Science
[2]MS in Computer Science, University of Illinois Springfield
[3]MS in Computer Science, University of Louisiana at Lafayette
[4]Computer Science Engineering, JNTU Hyderabad
[5]Masters in Telecommunications, Middlesex University
[6]Information Systems Engineer, CrowdStrike Inc

Article history

Submitted 27.09.2023 Revised Version Received 25.10.2023 Accepted 27.11.2023

Abstract

Purpose: This research aims to explore the role of Artificial Intelligence (AI) in enhancing quality assurance (QA) processes within cloud-native applications (CNAs). Specifically, it investigates how AI can address the challenges posed by the increasing complexity of cloud-native ecosystems and their associated quality control issues.

Methodology: The study adopts a qualitative research approach, involving case studies of organizations implementing AI-driven QA tools in cloud-native environments. Data is collected through interviews with industry experts, analysis of existing AI QA tools, and a review of relevant academic and industry literature. The research focuses on identifying the key AI technologies used, their integration into existing QA processes, and the impact on QA efficiency and accuracy.

Findings: The integration of AI in QA processes within CNAs has shown promising results in automating test generation, defect prediction, anomaly detection, and incident response. Machine learning, deep learning, and transformer-based models have enabled advanced functions such as log analysis, future failure prediction, and real-time test script maintenance. These technologies have significantly improved test reliability, coverage, and efficiency, particularly in highly automated and distributed cloud environments.

Implications to Theory, Practice, and Policy: From a theoretical perspective, this research contributes to the understanding of how AI can reshape traditional QA methodologies in the context of cloud-native applications. It highlights the need for a shift from manual, static testing approaches to intelligent, adaptive, and continuous QA processes. Practically, the study provides insights for organizations looking to integrate AI into their QA workflows, offering a roadmap for implementation and potential pitfalls. Policymakers can use these findings to guide the development of standards and best practices for AI-driven QA in cloud-native applications, ensuring that performance, resiliency, and efficiency are maintained across distributed cloud environments.

Keywords: Cloud-Native Applications (CNA), Continuous Quality Assurance (CQA), Artificial Intelligence (AI), Microservices, Containerization, Kubernetes, Smart Cloud Environments

INTRODUCTION

Cloud-native applications (CNAs) have transformed the pandemic of software engineering as well as the high demands and fast cycles of the contemporary IT world by taking full advantage of the modern technology stack of microservices, containerization, and orchestration to the utmost leading to high degrees of scalability and resilience. Intake of DevOps and continuous delivery pipelines is introducing organizations with a rising quantity, speed, and fluctuation of cloud-native deployments that result in the development of highly dynamic and distributed execution environments [1]. Meanwhile that these architectural advantages are accompanied with a new set of challenges that require a new mode of thinking and promises of quality and reliability and performance to be the prime ones among the challenges [2]. Conventional quality assurance (QA) methods have been created for monolithic, static, and predictable systems now they are no longer able to deal with the validation of interactions among microservices, behavior of elastic resources, various cloud infrastructures, and ever-changing software versions.

Continuous Quality Assurance (CQA) has become a crucial paradigm wherein the testing, monitoring, and validation activities are incorporated throughout the cloud-native systems' entire lifecycle [3]. CQA, as opposed to traditional QA, requires the use of automated and intelligent tools for observing runtime behavior, anticipating failures, spotting anomalies, and adjusting test cases at that very moment [4]. Implementing such functionalities in large systems is not possible through manual or rule-based techniques [5]. Consequently, AI is growingly embraced to support and partially take over the QA activities during the many stages of the software development, deployment, and production life cycle.

AI (artificial intelligence) techniques Defect prediction, automatic test case generation, continuous integration and delivery (CI/CD) pipeline optimization, and intelligent monitoring in complex cloud environments are just a few of the uses for techniques like machine learning (ML), deep learning (DL), natural language processing (NLP), and reinforcement learning [6]. Moreover, AI-powered techniques can spot performance degradations in advance, arrange dynamic resource allocations, and allow for the routine incident response through the analysis of huge amounts of logs, metrics, and traces. Notwithstanding these improvements, the literature is still not clear on where the systematic role of AI in continuous QA for cloud-native systems lies.

Structured of the Paper

The structure of this paper is as follows: Section II provides a foundation of Cloud-Native Applications and Quality Assurance. Section III explores AI Techniques in Continuous Quality Assurance. Section IV addresses AI-Enabled Quality Assurance Across the Cloud-Native Lifecycle. Section V reviews a relevant literature summary, and Section VI concludes with future research directions.

Foundations of Cloud-Native Applications and Quality Assurance

The necessity to investigate new and useful billing models has been sparked by Cloud-Native Applications (CNAs). Through a technique known as micro-billing, cloud service providers (CSPs) are now starting to provide more detailed pricing and resource-allocation control.

Microservices Architecture

The cloud-native architecture partitions the entire process into compact, self-governing units that communicate via APIs or an event-driven architecture. Not only does it provide flexibility, but it also poses certain challenges, such as testing across multiple services, managing several versions, and synchronizing complex inter-service dependencies. The transformation of SOA into microservices, characterized by low communication and decentralized governance, takes one step forward along the path of contemporary development patterns. The latter is certainly a diametric opposite to the previous monolithic systems, in which all the parts were too closely

interwoven, would be installed in one block, as shown in Figure 1. A monolithic architecture would therefore need to revise all its components, retest and re-deploy the whole system which obviously exposes risk and extends the release process.

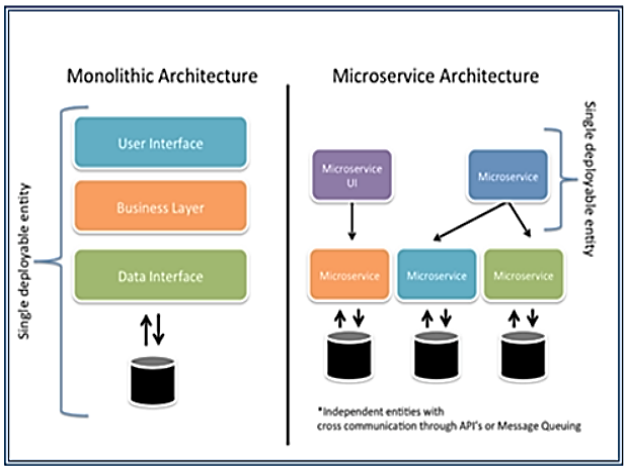

Figure 1: Microservice Architecture

Conversely, micro services are autonomous process units, which communicate through lightweight inter-process communication protocols, such as API that is based on HTTP. In contrast to the old models of services where several services might be involved in the same process and this would result in the entire system scaling when only one service could require additional capacity, microservices allow individual services to be scaled very precisely. The flexibility of this architecture should promote an iterative approach to development, this is what the modern development is about the core of the modern development is continuous, incremental changes. Teams can deploy new versions of individual microservices, as opposed to deploying larger, integrated systems, and this is much more likely to increase the agility, reliability, and maintainability of cloud-native systems.

Containers

Containerization technologies have modified the deployment of cloud technology, making it possible to deploy cloud-native by creating lightweight, isolated and consistent environments to run applications. The leader in the industry is Docker which is known to be portable, easy to use and has a strong ecosystem. Managed container services become unnecessary except that the very large cloud providers have one significant benefit, namely, the automatic infrastructure management and scaling. According to studies in the industry, containerized microservices are very effective in improving the performance of deployment so that they can initiate a quicker process and more frequent deployments compared to the traditional VM-based strategies. Containers are also resource-saving with regard to their less footprint and a similar operation system that is used. Containers provide organizations with better performance, reduced overhead, and scalability as they migrate to cloud-optimized container environments; thus, containers are a cornerstone of current cloud-native application development.

Kubernetes and Orchestration Platforms

Kubernetes is pretty much an orchestration system for containers that automatically manages Containerized applications are deployed and scaled by abstracting away the underlying infrastructure, allowing developers to concentrate on creating code instead of handling deployment concerns. Some of the key features include auto-scaling, self-healing, and service discovery which makes the management of containers across clusters of machines efficient [7]. Kubernetes works on the concept of pods, that are the smallest deployable units and may contain one or more containers sharing their resources like storage[8]. In Kubernetes, nodes are the

worker machines running these applications, each having a Kubelet agent to ensure appropriate containers are running. It presents services as a networking layer ensuring load balancing and communication of pods [9]. Deployments are the desired application state, and updates and scaling are easily achieved. The architecture of Kubernetes, which governs how the control plane interacts with the cluster, and how worker nodes execute application workloads, is demonstrated in Figure 2.

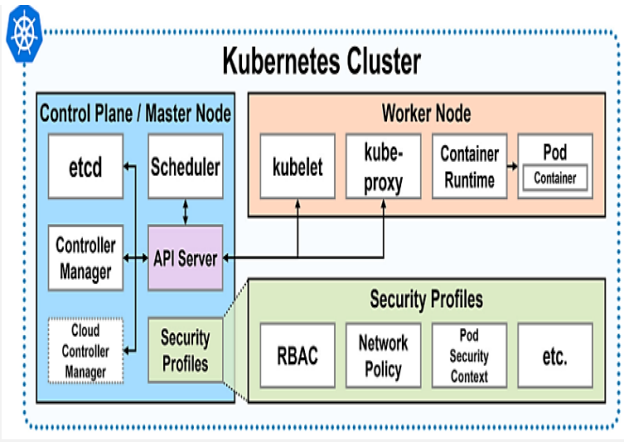

Figure 2: Kubernetes Architecture

Scheduling, configuration, and communication are done by the control plane, whereas containers are run by worker nodes with such components as kubelet, kube-proxy, and the container runtime. This design has features that allow a well-organized orchestration and also facilitate high-availability cluster design.

Challenges in QA for Smart Cloud Environments

The complexity of QA for smart cloud systems includes a wide range of novel issues posed by the dynamic, heterogeneous, and highly automated characteristics of modern cloud systems. Unlike conventional systems, the smart clouds have the following features: resource flexibility, resource allocation, microservice, multi-tenant, and real-time deliverable services. The other brilliant issue, which is a significant part of the situation, is resource elasticity: computing resources expand or contract in response to real-time demand [10]. Such volatility complicates the effort to make tests invariant and easy to run under similar, repeatable conditions, as would be needed in production [11]. Service Level Agreement (SLA) satisfaction is another serious subject. Smart cloud services are typically offered with high SLAs and sold in uptime, latency, throughput, and response-time packs[12]. The following are the Critical QA issues with regard to Smart Cloud environments:

- **Dynamic Resource Allocation:** The production environments cannot simulate the production performance and sizing degree of variance.
- **Multi-Tenant Risks:** There is difficulty in delivering isolation, security and performance on a series of tenants.
- **Data Consistency:** Data validation in distributed systems is made difficult due to eventually equal in distributed systems. A significant issue in the integrity of transactions in microservices is QA.
- **SLA Enforcement:** Should be checked continuously in terms of service reliability, latency, and availability.
- Microservices and Orchestration Complexity: Hard to validate parts of interdependent services and dynamic dependencies.

- **CI/CD and Rapid Deployment:** Fast-paced development processes require auto-test AI-enabled technologies.
- Heterogeneity and Interoperability: The diverse platforms, APIs, and configurations make it challenging to standardize QA strategies.
- **Autonomy and Adaptivity:** AI-driven or self-healing systems dynamically alter configurations, making it hard to validate behaviors against static test cases.

Limitations of Traditional QA Processes in Cloud-Native Systems

QA methods of the past that were designed for large, rigid, and easily forecasted systems, are having a hard time to cope with the requirements of cloud-native systems. The most important drawbacks are:

- Standard QA processes are ill-suited for inter-service and inter-environment communication. Such processes overlook the problems caused by the asynchronous nature of communications, the uncertainty of service availability, and the behavior of services being transient.
- In CI/CD settings that are fast-changing, tests that are either manually created or poorly scripted cannot cope with the high frequency of code changes and hence gaps in coverage occur.
- The conventional QA approach assumes stable environments for testing. In cloud-native applications, the infrastructure is continuously changing and automatically configured, making manual checking of environment consistency very difficult.
- The old monitoring tools are unable to handle the amount and speed of logs, metrics, and traces produced by cloud-native applications. Consequently, there are delays in detecting and diagnosing critical problems.
- Security testing done traditionally is infrequent and not continuous; this is not adequate because of the rapidly changing deployment environments, the presence of vulnerabilities in containers, and the attack surface at the API level.

AI Techniques in Continuous Quality Assurance

Artificial intelligence (AI) has become one of the main techniques that make it possible to have continuous quality assurance (CQA) for cloud-native applications. These techniques facilitate the entire process by providing high automation levels, increasing the accuracy of predictions, and allowing smart executives throughout the development, testing, and operation phases. This section of the article explains the basic AI categories—machine learning, deep learning, and natural language processing—and highlights how they contribute to scalable and adaptable quality assurance (across the cloud).

Machine Learning Approaches

The study of learning algorithms is known as machine learning (ML). When a computer program's performance on a particular task, as determined by a performance metric, increases with experience, this is referred to as learning. A model is the name given to the trained computer program [13] ML models, then, are statistical methods and computational algorithms that let computers to learn and make judgments or predictions without explicit programming. These models are founded on the ideas of pattern recognition and data-driven learning.

Supervised Learning

In supervised learning, labeled data is used to train the model to produce predictions or classifications. The matching output labels are linked with the input data (features):

- **Classification:** The algorithm learns to predict discrete class labels for input data in this job, such as categorizing a produced product as faulty or non-defective.
- **Regression:** In this case, the algorithm gains the ability to forecast continuous numerical values. For example, this involves forecasting a production line's energy usage.

Unsupervised Learning

Unsupervised learning is a machine learning method in which a model acquires patterns, structures or relationships in data with no direct instruction provided by labelled data [14]. It seeks to find intrinsic patterns or groups in the data, hence making it possible to perform tasks like clustering, dimensionality reduction and anomaly detection [15]. Contrary to supervised learning, unsupervised learning lacks known target labels, hence can be used to explore and comprehend unlabelled data:

- **Clustering:** In accordance with patterns or similarities in the input data, the algorithm groups together comparable data points. For example, this is the collection of devices with comparable performance attributes.
- **Dimensionality Reduction:** This algorithm eliminates unwanted features of the input in this task and preserves the valuable information. It helps visualize and digest high-dimensional data and may serve as a pre-processing phase of other machine learning problems.
- **Anomaly Detection (AD):** In this case, the algorithm gets to learn the regularities in the data and detects any data that does not conform remarkably to the regularities, which in most cases, represent anomalies or outliers. For instance, this technique may be applied to process machinery signals to detect wear by observing anomalous signal behavior.
- **Autoencoder (AE):** An AE is a kind of neural network that can learn to both encode and decode input; it is commonly employed for dimensionality reduction and unsupervised learning. The input data is compressed into a lower-dimensional representation (latent space) by an encoder network, and the original data is reconstructed from the latent space by a decoder network.

Semi-supervised Learning

This learning paradigm uses data from training that is both labelled and unlabelled. To enhance model performance, it makes use of both a greater quantity of unlabelled data and a smaller quantity of labelled data. It is helpful when data labelling is costly or time-consuming.

Reinforcement Learning

An agent interacts with its surroundings in this kind of learning. To optimize a reward signal, the agent learns to act in the environment. In order to accomplish long-term objectives, the agent learns to make the best choices in a variety of environmental conditions through trial and error. The primary applications of reinforcement learning include visual navigation, robotics, and video games, which all need multistep decision-making.

Deep Learning and Transformer-Based Models

In order to tackle the challenge, deep learning techniques like CNNs and RNNs have been employed to encode and analyze code as either trees or sequences. For instance, code sequential relations, which are typical of doors, have been modelled using long short-term memory (LSTM) networks. While these networks have some efficacy, they frequently fall short in scaling up and handling global code contexts.

Natural language processing has advanced significantly thanks to transformers, particularly BERT, which can capture the contextual connections between tokens [16]. Adaptations like Code BERT and Graph Code BERT, which modify the transformer architecture for code

representation and comprehension, were motivated by this ground-breaking work and directly target source code. These findings suggest that domain-specific previous knowledge and structural modifications might be applied to further enhance code analysis task solutions [17]. Because code is closely connected to natural language both structurally and semantically, their success has inspired them to pursue code analysis. The most recent techniques, CoRT and updated Code BERT, have demonstrated the effectiveness of transformers in modelling the code and identifying smells. In order to identify contextual dependencies included in the code, these models make use of the self-attention process.

AI-Enabled Quality Assurance Across the Cloud-Native Lifecycle

The AI that transforms QA in the cloud setting is the automation of the following major tasks, including test cases generation, defect prediction, and maintenance of scripting with the help of NLP, DL, and ML. Artificial intelligence-based tools invaluable in enhancing the percentage of the test coverage to higher proportions with a minimum amount of manual intervention. Besides automation, AI-based analytics are useful in identifying anomalies and determining their cause at the earliest stage possible; which makes troubleshooting faster and better. There is also smart monitoring usage which involves predictive capabilities to anticipate abnormalities or dynamic resources to minimize downtime [18]. The adaptive and highly efficient quality assurance processes of AI enable the overall performance, responsiveness and resilience of the cloud-based systems in complex smart cloud and real time smart cloud environments.

Quality Assurance in Cloud

A component of the ecosystem that ensures consumers receive high-quality products free from unintentional flaws is quality assurance, which is frequently used in tandem with quality control. Instead of using quality verification, manufacturers use testing to control the quality of their products [19]. Since its first introduction as a systematic strategy in the manufacturing industry, the idea of quality assurance has spread to various industries, including computer programming. High-quality product options are produced, gaining the trust and loyalty of customers. One of the main objectives of a QA program is to stop future product failures. The methodical verification of a product or service's compliance with predetermined standards is the aim of quality assurance. The goal of quality assurance is to establish and uphold standards for procedures that are created or manufactured.

AI-driven QA Tools and Techniques

QA technology and methodologies provided by artificial intelligence are changing the face of software testing by applying intelligence and flexibility and automating much of the job. They are anchored on machine learning, natural language programming, and neural networks in order to enable prediction of defects, generating test cases, and automatic script maintenance. As an example, with the help of AI, it is possible to locate the part of the code that is the most dangerous, prioritize tests, and even edit scripts when the application is up to the date, so that much manual effort is saved and the number of errors is decreased. NLP-based tools can help the requirements stated in natural language be turned into test cases, and deep learning models can be used as a means of discovering complicated anomalies in the UI or the code. Overall, AI helps to improve the effectiveness of the testing, its accuracy and speed, which is a demand in existing quality control procedure.

Automated Testing and Defect Prediction Using AI for Software QA

Traditional testing methods have failed to meet the new issues of complexity and increase of releases of software today. Artificial intelligence (AI) is, therefore, becoming a core part of automated testing frameworks as it allows one to predetermine the expected result of a program and expand testing opportunities by way of enlarging the test map. Machine learning algorithms and data-driven insight present by AI-powered testing allow forecasting defects in advance or

prioritizing best test based on the documented patterns within the historical testing data [20]. Understanding the metrics of the past development cycles enables AI to determine the components that are high-risk, so that quality assurance (QA) teams could assign their resources with a more targeted approach. Besides enhancing the accuracy of the tests, this predictive model reduces the dependence on manual tests. Also, AI may automate routine work on regression and performance testing and allow human testers to pay more attention to more important and experimental QA functions. Finally, AI testing reduces the time spent, facilitates more expedited releases, and sustains a top-level of software quality.

AI for Anomaly Detection and Root Cause Analysis for Software QA

Anomaly detection and root cause analysis AI-based are approaches which use ML algorithms, to automatically detect deviations in normal system behavior as well as derive their underlying reasons. These approaches are complementary to the established Quality Assurance (QA) approaches because they enable the preventive check-up, reduction of the detection time and enable easy correction of the problem. With the application of AI-powered systems in intelligent cloud environments, metrics, logs, and traces can be analyzed at scale to ensure accurate perception of performance bottlenecks, security breaches, and system failures. This maintains end-to-end service quality, service stability, and reliability of dynamic cloud infrastructures.

AI-Powered Monitoring and Incident Response in the Cloud

Proactive cloud AI monitoring and incident response have irreversibly changed reactive habits into smart, intelligent cloud management. By applying machine learning and big data analytics, AI actively monitors the real-time cloud metrics and knows in advance how the system behave, anomalies, and potential threats than the system based on threshold-based systems [21]. AI could also instigate automatic reaction, scale the resources, invoke a failover, or gather diagnostic data when an issue is reported, which lowers the response time significantly and eliminates the necessity of a manual response almost completely [22]. Using an example of AWS CloudWatch connected to Amazon Sage Maker enable real-time mining of the metrics, predictive modelling, managing dynamic resources, and intelligent notification through the AI-based algorithms. Moreover, AI helps with incident management by grouping alerts and assigning them to the competent teams, as well as recommending a possible solution using the past data. These features really boost the reliability, effectiveness and resiliency of the cloud environments and allow detecting problems early and resolve them soon and leverage the best resource utilization and depend less on human input.

LITERATURE REVIEW

Below literature summary, scholars have studied the subjects of automation, interoperability, and intelligent quality assurance protocols of cloud-native settings in a broad manner. The articles in general point to the progress in API migration, automated Kubernetes deployment, elastic performance testing, cognitive workload management, and DevOps-focused continuous QA. Table I provides an overview of the major focus areas, and methodologies of these works as well as their findings and future trends of research.

Jayakody, Perera and Perera (2019) outlines a method to reduce the time and effort required to recreate application programming interface artifacts, promote cross-collaboration, and enable effective migration between environments with different platforms without requiring major post-migration adjustments or extra work. Software development companies today maintain distinct environments for development, quality assurance, and production thanks to the Agile philosophy of developing and delivering products in a milestone-based method. These environments run on their own, with their own deployments and traffic control strategies [23].

Astyrakakis et al. (2019) presents an innovative, fully automated solution for setting up and keeping an eye on a Kubernetes cluster running on OpenStack. A solution that offers cloud-

native application validation automatically is what they suggest. When the suggested toolbox was evaluated, Kubernetes clusters were deployed with very short overall timeframes when compared to previous manual methods. For a containerized application with the Kubernetes Horizontal Pod AutoAlert (HPA) enabled, the validation procedure took around 11 minutes; for an application with the HPA deactivated, it took about 3 minutes. These total durations are comparatively shorter than a number of alternative non-automated methods. The previously stated total timings are comparable to the test-bed's underlying hardware and network materials [2].

Kratzke and Quint (2017) outlines research foci and trends pertaining to cloud-native application engineering techniques, as well as the results of a systematic mapping analysis that examined research publications addressing "cloud-native" subjects, research issues, and engineering methodology. Additionally, a definition of the phrase "cloud-native application" was given, taking into consideration all of the conclusions, insights, and already established and well-defined vocabulary from the examined publications [24].

Vankayala (2018) addresses the growing need for an adaptive, automated, and scalable performance testing framework capable of supporting dynamic workloads across microservices, containerized environments, and distributed infrastructures. The study presents a flexible performance testing architecture that is designed to dynamically process resources, load distribution, and perform predictive scaling in test performance using an orchestration of data. By embracing a mixed-method research strategy involving the incorporation of architectural modelling, quantitative benchmarking, and empirical validation in a variety of cloud service frameworks, it can be seen that the research proposal enhances the degree of consistency in throughput by 27% and test execution latency by 19% relative to traditional, un-developed, static-testing pipelines. Key innovations include a dynamic elasticity engine that aligns resource utilization with workload behavior and a modular design that integrates seamlessly with CI/CD toolchains, ensuring real-time performance insights during deployment cycles [25].

Mikkilineni, Morana and Keshan (2017) explains the continuation of the work begun during WETICE 2009's inaugural Cloud Computing session. In order to showcase globally interoperable public and private cloud networks operating cloud agnostic workloads, a novel computing paradigm utilizing distributed intelligent managed elements (DIME) and DIME network architecture was proposed in WETICE2010. The cognitive workloads have the ability to self-provision, self-heal, self-monitor, and self-control in order to modify their structure and preserve the required level of service quality. Workload deployment on an interoperable cloud network using the public Internet is demonstrated using a three-tier marketing application that consists of an author module, publisher module, and dispatcher module. This method offers workload management on any infrastructure, including bare-metal servers, virtual machines, and containers, defying the current trend of making workloads cloud-native [26]

Oliveira et al. (2016) assert that a new method of identifying issues, including those that could have arisen during the continuous deployment cycle, is required to realize the goal of quick software delivery without sacrificing the calibre of the services offered. Proposing such a continuous quality assurance approach, a native DevOps-centric approach to problem resolution focuses on a broader range of potential error sources (including code commits), uses DevOps metadata to clearly define the source of the problem, and results in a speedy problem resolution. It is demonstrated through initial experiments in both a private OpenStack® cloud environment and a public Container Cloud environment [27]

Table 1: Continuous Quality Assurance in Cloud-Native Applications using Artificial Intelligence

Reference	Focus Area	Key Findings	Approaches	Objectives	Future Work
Jayakody, Perera & Perera (2019)	API migration across heterogeneous environments	Minimizes time and effort in recreating API artifacts; supports seamless migration without post-migration changes	Automated API recreation, cross-environment migration framework	Enable cross-collaboration and efficient migration between dev, QA, and production environments	Extend automated migration to multi-cloud ecosystems and incorporate AI-driven consistency checks
Astyrakakis et al. (2019)	Automated Kubernetes deployment and validation	Automated Kubernetes cluster deployment yields significantly lower setup and validation times	Automated toolchain for Kubernetes-on-OpenStack deployment; validation tool for cloud-native apps	Provide a fully automated system for Kubernetes deployment and application validation	Optimize tool performance for large-scale clusters and integrate ML-based anomaly detection
Kratzke & Quint (2017)	Cloud-native application engineering methodologies	Systematic mapping reveals major research gaps and trends in CNA engineering	Literature-based systematic mapping and taxonomy development	Define and consolidate the terminology and methodologies for cloud-native applications	Expand mapping to emerging CNA technologies like serverless, service mesh, and edge computing
Vankayala (2018)	Elastic and scalable performance testing for cloud-native systems	Proposed elastic testing framework increases throughput by 27% and reduces latency by 19%	Data-driven orchestration, dynamic elasticity engine, empirical benchmarking	Develop an adaptive, automated, scalable performance testing framework for microservices	Incorporate predictive AI models for autonomous resource scaling and integrate with multi-cloud CI/CD
Mikkilineni, Morana & Keshan (2017)	Cognitive cloud workloads and interoperable cloud networks	DIME architecture enables self-managing, cloud-agnostic, interoperable workloads	Distributed Intelligent Managed Elements (DIME), cognitive workload model	Demonstrate self-provisioning, self-healing, cloud-agnostic workload execution	Extend interoperability to hybrid edge-cloud environments and enhance cognitive capabilities using AI
Oliveira et al. (2016)	DevOps-centric Continuous Quality Assurance	DevOps metadata improves root-cause identification and accelerates problem resolution	Native DevOps-centric CQA framework; experiments on public and private cloud containers	Detect and resolve issues across the continuous deployment cycle more efficiently	Integrate automated fault localization and ML-based prediction of deployment failures

Conclusion and Future Work

Continuous Quality Assurance (CQA), especially in cloud-native settings, has been made possible by Artificial Intelligence due to the limitations inherent in traditional frameworks used to implement quality assurance in a modern system that is difficult to understand, becomes elastic, and can be deployed within hours. The incorporation of machine learning, deep learning, and transformer-based models allows AI to improve all the stages of the cloud-native lifecycle, including automated test generation and defect prediction as well as intelligent monitoring, anomaly detection, and incident response. One capability of an AI model is linked to how it contributes to test coverage by mitigating the trivial bug-finding activity for the QA team. Intelligent AI-based QA also enhances the resiliency of the systems by actively determining performance bottlenecks, anticipating failures, and ensuring the optimization of adaptive resources in the dynamic and distributed systems. With the ongoing growth of cloud-native ecosystems, AI-based QA become indispensable to the creation of scalable, efficient, and self-healing systems to satisfy the needs of the current digital services.

Future studies ought to seek to consider integrating generative AI to aid in autonomous test creation, improve multimodal observability analytics, as well as creating self-learning QA agents that are learning to dynamically adapt to cloud-based environments. Also, AI combined with edge computing, zero-trust security, and policy-directed orchestration can also enhance continuous quality assurance in cloud-native ecosystems of the next generation.

REFERENCES

[1] D. Gannon, R. Barga, and N. Sundaresan, "Cloud-Native Applications," *IEEE Cloud Comput.*, vol. 4, pp. 16–21, 2017, doi: 10.1109/MCC.2017.4250939.

[2] N. Astyrakakis, Y. Nikoloudakis, I. Kefaloukos, C. Skianis, E. Pallis, and E. K. Markakis, "Cloud-Native Application Validation & Stress Testing through a Framework for Auto-Cluster Deployment," in *2019 IEEE 24th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2019, pp. 1–5. doi: 10.1109/CAMAD.2019.8858164.

[3] J.-M. Fernandez, I. Vidal, and F. Valera, "Enabling the Orchestration of IoT Slices through Edge and Cloud Microservice Platforms," *Sensors*, vol. 19, no. 13, p. 2980, Jul. 2019, doi: 10.3390/s19132980.

[4] M. Imran, H. Hlavacs, I. U. Haq, B. Jan, F. A. Khan, and A. Ahmad, "Provenance based data integrity checking and verification in cloud environments," *PLoS One*, vol. 12, no. 5, p. e0177576, May 2017, doi: 10.1371/journal.pone.0177576.

[5] M. Muniswamaiah, T. Agerwala, and C. Tappert, "Big Data in Cloud Computing Review and Opportunities," *Int. J. Comput. Sci. Inf. Technol.*, 2019, doi: 10.5121/ijcsit.2019.11404.

[6] M. Shahin, M. Ali Babar, and L. Zhu, "Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices," 2017. doi: 10.1109/ACCESS.2017.2685629.

[7] L. A. Vayghan, M. A. Saied, M. Toeroe, and F. Khendek, "Kubernetes as an Availability Manager for Microservice Applications," 2019.

[8] V. Medel, O. Rana, J. ángel Bañares, and U. Arronategui, "Modelling performance & resource management in kubernetes," in *Proceedings of the 9th International Conference on Utility and Cloud Computing*, 2016, pp. 257–262. doi: 10.1145/2996890.3007869.

[9] A. Kushwaha, P. Pathak, and S. Gupta, "Review of optimize load balancing algorithms in cloud," *Int. J. Distrib. Cloud Comput.*, vol. 4, no. 2, pp. 1–9, 2016.

[10] M. Faheem, U. Akram, I. Khan, S. Naqeeb, A. Shahzad, and A. Ullah, "Cloud Computing Environment and Security Challenges: A Review," *Int. J. Adv. Comput. Sci. Appl.*, 2017, doi: 10.14569/ijacsa.2017.081025.

[11] Q. K. Kadhim, R. Yusof, H. S. Mahdi, S. S. Ali Al-shami, and S. R. Selamat, "A Review Study on Cloud Computing Issues," *J. Phys. Conf. Ser.*, vol. 1018, p. 012006, May 2018, doi: 10.1088/1742-6596/1018/1/012006.

[12] S. Achouche, U. B. Yalamanchi, and N. Raveendran, "Method, apparatus, and computer-readable medium for performing a data exchange on a data exchange framework," 2019

[13] T. Hirota, K. Okuda, M. Masuda, and S. Yasukawa, "Cloud Native SDx Control Technology," *NTT Tech. Rev.*, vol. 16, no. 7, pp. 39–43, Jul. 2018, doi: 10.53829/ntr201807ra1.

[14] M. M. El Khatib, A. Al-Nakeeb, and G. Ahmed, "Integration of Cloud Computing with Artificial Intelligence and Its Impact on Telecom Sector—A Case Study," *iBusiness*, vol. 11, no. 01, pp. 1–10, 2019, doi: 10.4236/ib.2019.111001.

[15] S. Garg, "Predictive Analytics and Auto Remediation using Artificial Inteligence and Machine learning in Cloud Computing Operations," *Int. J. Innov. Res. Eng. Multidiscip. Phys. Sci.*, vol. 7, no. 2, 2019.

[16] P. Srivastava and R. Khan, "A Review Paper on Cloud Computing," *Int. J. Adv. Res. Comput. Sci. Softw. Eng.*, 2018, doi: 10.23956/ijarcsse.v8i6.711.

[17] B. H. Li, B. Hou, W. Yu, X. Lu, and C. Yang, "Applications of artificial intelligence in intelligent manufacturing: a review," *Front. Inf. Technol. Electron. Eng.*, vol. 18, no. 1, pp. 86–96, Jan. 2017, doi: 10.1631/FITEE.1601885.

[18] S. Gina, N. I. Belu, N. Rachieru, and V. Nicolae, "Improvement of the customer satisfaction through Quality Assurance Matrix and QC-Story methods: A case study from automotive industry," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 252, no. 1, p. 012045, Oct. 2017, doi: 10.1088/1757-899X/252/1/012045.

[19] R. von Solms and M. Willett, "Cloud computing assurance – a review of literature guidance," *Inf. Comput. Secur.*, vol. 25, no. 1, pp. 26–46, Mar. 2017, doi: 10.1108/ICS-09-2015-0037.

[20] N. Kumar, "Anomaly Detection in ERP Systems Using AI and Machine Learning," *Int. J. Sci. Res. Sci. Eng. Technol.*, pp. 522–530, 2019.

[21] R. Jathanna and D. Jagli, "Cloud Computing and Security Issues," *Int. J. Eng. Res. Appl.*, vol. 07, no. 06, pp. 31–38, Jun. 2017, doi: 10.9790/9622-0706053138.

[22] L. B. A. Rabai, M. Jouini, A. Ben Aissa, and A. Mili, "A cybersecurity model in cloud computing environments," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 25, no. 1, pp. 63–75, Jan. 2013, doi: 10.1016/j.jksuci.2012.06.002.

[23] J. A. D. C. A. Jayakody, A. K. A. Perera, and G. L. A. K. N. Perera, "Cloud Native Efficient Solution for API Migration Across Environments For Agile Integration Enterprises," in *2019 International Conference on Advancements in Computing (ICAC)*, IEEE, Dec. 2019, pp. 168–173. doi: 10.1109/ICAC49085.2019.9103393.

[24] N. Kratzke and P. C. Quint, "Understanding cloud-native applications after 10 years of cloud computing - A systematic mapping study," *J. Syst. Softw.*, 2017, doi: 10.1016/j.jss.2017.01.001.

[25] S. C. Vankayala, "Engineering Elastic Performance Testing Frameworks for Cloud- Native Applications: A Scalable Design Perspective," *J. Sci. Eng. Res.*, vol. 5, no. 8, pp. 301–315, 2018.

[26] R. Mikkilineni, G. Morana, and S. Keshan, "Globally interoperable network of clouds and cognitive workload quality of service assurance," in *Proceedings - 2017 IEEE 26th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, WETICE 2017*, 2017. doi: 10.1109/WETICE.2017.10.

[27] F. Oliveira *et al.*, "Delivering software with agility and quality in a cloud environment," *IBM J. Res. Dev.*, vol. 60, no. 2–3, pp. 10:1-10:11, Mar. 2016, doi: 10.1147/JRD.2016.2517498.

Routhu, K. K. (2022). From Case Management to Conversational HR: Redefining Help Desks with Oracle's AI and NLP Framework. *International Journal of Science, Engineering and Technology*, *10*(6).

Attipalli, A., BITKURI, V., KURMA, J., Enokkaren, S., Kendyala, R., & Mamidala, J. V. (2021). A Survey of Artificial Intelligence Methods in Liquidity Risk Management: Challenges and Future Directions. *Available at SSRN 5741342*.

Attipalli, A., BITKURI, V., Mamidala, J. V., Kendyala, R., & KURMA, J. (2022). Empowering Cloud Security with Artificial Intelligence: Detecting Threats Using Advanced Machine learning Technologies. *Available at SSRN 5741263*.

Attipalli, A., Enokkaren, S., BITKURI, V., Kendyala, R., KURMA, J., & Mamidala, J. V. (2021). Enhancing Cloud Infrastructure Security Through AI-Powered Big Data Anomaly Detection. *Available at SSRN 5741305*.

Bitkuri, V., Kendyala, R., Kurma, J., Mamidala, V., Enokkaren, S. J., & Attipalli, A. (2021). Systematic Review of Artificial Intelligence Techniques for Enhancing Financial Reporting and Regulatory Compliance. *International Journal of Emerging Trends in Computer Science and Information Technology*, *2*(4), 73-80.

Kothamaram, R. R., Rajendran, D., Namburi, V. D., Singh, A. A. S., Tamilmani, V., & Maniar, V. (2021). A Survey of Adoption Challenges and Barriers in Implementing Digital Payroll Management Systems in Across Organizations. *International Journal of Emerging Research in Engineering and Technology*, *2*(2), 64-72.

Padur, S. K. R. (2018). Autonomous cloud economics: AI driven right sizing and cost optimization in hybrid infrastructures. *International Journal of Scientific Research in Science and Technology*, *4*(5), 2090-2097.

Padur, S. K. R. (2019). Machine learning for predictive capacity planning: Evolution from analytical modeling to autonomous infrastructure. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, *5*(5), 285-293.

Padur, S. K. R. (2020). AI augmented disaster recovery simulations: From chaos engineering to autonomous resilience orchestration. *International Journal of Scientific Research in Science, Engineering and Technology*, *7*(6), 367-378.

Padur, S. K. R. (2020). From centralized control to democratized insights: Migrating enterprise reporting from IBM Cognos to Microsoft Power BI. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol*, *6*(1), 218-225.

Padur, S. K. R. (2021). Bridging Human, System, and Cloud Integration through RESTful Automation and Governance. *the International Journal of Science, Engineering and Technology*, *9*(6).

Padur, S. K. R. (2021). Deep learning and process mining for ERP anomaly detection: Toward predictive and self-monitoring enterprise platforms. *Available at SSRN 5605531*.

Padur, S. K. R. (2021). From Control to Code: Governance Models for Multi-Cloud ERP Modernization. *International Journal of Scientific Research & Engineering Trends*, *7*(3).

Padur, S. K. R. (2022). AI augmented platform engineering, transforming developer experience through intelligent automation and self optimizing internal platforms. *International Journal of Science, Engineering and Technology*, *10*(5), 10-5281.

Padur, S. K. R. (2022). Intelligent resource management: AI methods for predictive workload forecasting in cloud data centers. *J. Artif. Intell. Mach. Learn. & Data Sci*, *1*(1), 2936-2941.

Polam, R. M., Kamarthapu, B., Kakani, A. B., Nandiraju, S. K. K., Chundru, S. K., & Vangala, S. R. (2022). Data Security in Cloud Computing: Encryption, Zero Trust, and Homomorphic Encryption. *International Journal of Emerging Trends in Computer Science and Information Technology*, *3*(4), 31-41.

Polu, A. R., Buddula, D. V. K. R., Narra, B., Gupta, A., Vattikonda, N., & Patchipulusu, H. (2021). Evolution of AI in Software Development and Cybersecurity: Unifying Automation, Innovation, and Protection in the Digital Age. *Available at SSRN 5266517*.

Rajendran, D., Namburi, V. D., Singh, A. A. S., Tamilmani, V., Maniar, V., & Kothamaram, R. R. (2021). Anomaly Identification in IoT-Networks Using Artificial Intelligence-Based Data-Driven Techniques in Cloud Environmen. *International Journal of Emerging Trends in Computer Science and Information Technology*, *2*(2), 83-91.

Reddy Padur, S. K. (2021). From Scripts to Platforms-as-Code: The Role of Terraform and Ansible in Declarative Infrastructure Rollouts. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, 621-628.

Routhu, K. K. (2021). Harnessing AI Dashboards in Oracle Cloud HCM: Advancing Predictive Workforce Intelligence and Managerial Agility. *International Journal of Scientific Research & Engineering Trends*, *7*(6).

Routhu, K. K. (2022). From RFID to Geofencing: IoT-Enabled Smart Time Tracking in Oracle HCM Cloud. *International Journal of Science, Engineering and Technology*, *10*(4).

Singh, A. A. S., Tamilmani, V., Maniar, V., Kothamaram, R. R., Rajendran, D., & Namburi, V. D. (2021). Predictive Modeling for Classification of SMS Spam Using NLP and ML Techniques. *International Journal of Artificial Intelligence, Data Science, and Machine Learning*, *2*(4), 60-69.

Vattikonda, N., Gupta, A. K., Polu, A. R., Narra, B., Buddula, D. V. K. R., & Patchipulusu, H. H. S. (2022). Blockchain Technology in Supply Chain and Logistics: A Comprehensive Review of Applications, Challenges, and Innovations. *International Journal of Emerging Trends in Computer Science and Information Technology*, *3*(3), 72-80.

License