NEWPORT INTERNATIONAL JOURNAL OF SCIENTIFIC AND EXPERIMENTAL SCIENCES (NIJSES)
Volume 6 Issue 2 Page 37-43, 2025
©NIJSES PUBLICATIONS
Open Access
ONLINE ISSN:2992-5819
PRINT ISSN:2992-6149
https://doi.org/10.59298/NIJSES/2025/62.374300

Software Engineering Dimensions: Empirical Frameworks on Microservices Architecture in Cloud Computing Domains

[1]Godspower I. Akawuku, [2]Onyinyechi H. Ikediego and [3]Nwankwo Chekwube

[1,2]Department of Computer Science, Nnamdi Azikiwe University, Awka.
[3]Chukwuemeka Odumegwu Ojukwu University, Uli Campus Anambra State, Nigeria

ABSTRACT

Microservices Architecture (MSA) has become a popular approach for developing scalable and adaptable cloud applications, offering enhanced flexibility, maintainability, and resilience compared to traditional monolithic systems. However, the practical application of MSA introduces operational complexities, particularly in optimizing performance, ensuring fault tolerance, and managing inter-service communication. Empirical research is essential to address these challenges and validate MSA's performance. This involves employing methodologies such as case studies, performance testing, and profiling to provide actionable insights into key areas like scalability, latency, and system resilience. By evaluating system behavior under varying conditions, researchers can bridge the gap between the theoretical advantages of MSA and its real-world application in cloud environments. This research emphasizes the importance of empirical frameworks in guiding the implementation and evaluation of MSA. These frameworks offer structured methods for assessing performance, scalability, and reliability, thereby enabling organizations to make informed decisions about adopting and optimizing MSA. The findings contribute to a deeper understanding of the trade-offs and best practices associated with MSA, supporting its effective deployment in complex cloud-based systems.

Keywords: Software Engineering, Microservices, Frameworks, Cloud Computing Domains

INTRODUCTION

Microservices Architecture (MSA) has become a leading paradigm for designing scalable and adaptable cloud-based applications. Unlike traditional monolithic architectures that centralize functionalities within a tightly coupled unit, MSA breaks down applications into independently deployable services, each responsible for a specific business capability [1], [2] and [3]. This architectural shift offers enhanced flexibility, maintainability, and resilience, making it especially suitable for dynamic cloud environments [4], [3]. However, despite the well-documented theoretical and industry endorsements [4], [5], implementing MSA at scale presents operational challenges, particularly in performance optimization, fault tolerance, and inter-service communication. A notable example is Netflix, an early adopter of MSA on a global scale. The company initially faced challenges with service latency and fault recovery. To mitigate these issues, Netflix developed empirical tools such as Hystrix, a circuit breaker for service failure handling, and Chaos Monkey, which tests system resilience by simulating random failures [1], [2], [6], [3]. These efforts highlight the critical role of empirical validation in ensuring that microservices architectures remain reliable under real-world cloud conditions. Empirical research plays a vital role in substantiating the theoretical promises of MSA. By evaluating system performance under varying workloads, network conditions, and failure scenarios, researchers can generate actionable insights into scalability, latency, and system resilience [4]. Without empirical backing, organizations may overestimate the benefits of MSA, potentially leading to unexpected bottlenecks, degraded user experiences, and increased operational costs. To address this, researchers have developed empirical frameworks that provide structured methods for evaluating MSA performance. These frameworks are particularly relevant in cloud environments, where fluctuating workloads and virtualized resources can significantly impact system behavior. Tools such as Prometheus (for monitoring) and Jaeger (for distributed tracing) have become central to performance measurement and observability [4]. By clearly defining metrics like response time, throughput, and service availability, these frameworks bridge the gap between conceptual advantages and practical implementation. Without such validation, organizations risk adopting architectures that fall short of delivering consistent Quality of Service (QoS) [4],[7].

Empirical Research Methods in Microservices Architecture

A range of empirical research methodologies is employed to study Microservices Architecture (MSA) in cloud computing, each offering unique insights into system performance, scalability, and reliability. These methodologies facilitate a comprehensive understanding of MSA by offering both qualitative and quantitative assessments of real-world implementations and controlled experimental evaluations.

Case Studies

Case studies serve as a foundational empirical method for evaluating Microservices Architecture (MSA) in real-world settings. They offer deep, contextual insights into the motivations behind MSA adoption, the operational challenges encountered, and the performance outcomes achieved across various industries [4]. These studies often focus on how organizations navigate deployment complexities, inter-service communication, and the need for scalability when transitioning from monolithic to microservices-based systems. For example, Amazon transitioned to microservices to enhance scalability during high-traffic events such as Black Friday. By decoupling services, the company enabled independent scaling of components, which resulted in improved system performance and significant cost savings [8]. Similarly, Netflix adopted MSA to support a rapidly expanding global user base. It introduced resilience tools like Chaos Monkey for failure simulation and adaptive streaming capabilities, which allowed for thousands of daily infrastructure changes without disrupting services—enhancing both maintainability and innovation speed [8], [9],]10]. UPWARD, Inc. leveraged MSA to align with its growth strategy, implementing Infrastructure as Code (IaC), Platform as a Service (PaaS), and cloud-native practices. These efforts resulted in standardized deployments and faster release cycles, improving infrastructure consistency and operational efficiency [11]. In the healthcare sector, SPIDEP's eHealth platform underwent a performance evaluation comparing MSA with Service-Oriented Architecture (SOA) using a Kubernetes-based environment. The study found that MSA delivered better response times and resource efficiency, albeit with increased bandwidth consumption—a trade-off for improved responsiveness [12]. Alibaba's experience further illustrates the complexities of managing microservices at scale. The company focused on fault recovery, load balancing, and service dependency management using real production traces. Empirical results demonstrated performance improvements ranging from 8.59% to 12.34% in the Sock-Shop application and from 7.30% to 11.97% in Hotel-Reservation services when compared to baseline systems [13], [14]. These case studies highlight the adaptability and performance potential of MSA across diverse domains. They also reveal shared objectives among adopters—such as scalability, resilience, and innovation—supported by quantifiable performance metrics that guide implementation decisions.

Performance Testing

Performance testing plays a critical role in evaluating how Microservices Architecture (MSA) systems respond to both typical and extreme operational conditions. These tests focus on key metrics such as response time, throughput, latency, and system resilience, helping to ensure that microservices can sustain performance under stress, peak loads, and fault scenarios [4]. Amazon provides a noteworthy example of large-scale performance testing. As part of its transition to microservices, the company conducted rigorous scalability assessments to maintain service consistency during high-demand periods such as holiday sales events. Although specific test metrics were not publicly disclosed, the reported success in achieving cost efficiency and performance optimization suggests that empirical validation was central to its deployment strategy [8]. Netflix has pioneered fault tolerance testing using tools like Chaos Monkey, which intentionally disrupts production environments by simulating random service failures. This strategy has proven effective in verifying the robustness of microservices under partial outages, ensuring high availability and uninterrupted user experiences [8]. Empirical research on a .NET Core-based microservices system demonstrated that microservices can outperform monolithic systems under load. The study reported more than double the throughput and significantly reduced latency—attributed to fine-grained service components and more effective resource distribution [15]. Alibaba's evaluation of its StatuScale framework, deployed within Kubernetes environments, further illustrates the value of empirical testing. Using real workload traces, the company validated the efficiency of workload-aware auto-scaling. Response time improvements ranged from 7.30% to 12.34% across two production services, underscoring the effectiveness of adaptive resource management [14]. Additionally, the SPIDEP platform underwent spike testing to simulate abrupt traffic surges. Comparisons between MSA and Service-Oriented Architecture (SOA) implementations revealed that MSA achieved faster response times and more efficient infrastructure usage, despite a modest increase in bandwidth consumption [12]. Collectively, these cases demonstrate that performance testing—whether through load testing, stress simulation, or resilience evaluation—**produces essential empirical data**. This data not only validates architectural decisions but also supports the optimization and dependability of microservices systems in real-world cloud environments.

Profiling

Profiling is a vital empirical technique for analyzing the runtime behavior and resource consumption of microservices-based systems. It focuses on evaluating CPU usage, memory footprint, and communication latency between services—critical factors in identifying performance bottlenecks and optimizing system efficiency [4]. In one empirical study involving .NET Core-based microservices, profiling revealed significant improvements in CPU and memory usage compared to a monolithic implementation. These gains were

attributed to the ability to isolate and fine-tune resource-intensive components, thereby enhancing resource efficiency under real workload conditions [15]. Alibaba employed profiling extensively to support resource provisioning strategies in large-scale deployments. By analyzing real production traces, the company measured the correlation between Queries Per Second (QPS) and CPU utilization. This enabled the development of predictive QPS models, which guided more effective autoscaling decisions across its microservices infrastructure [14]. In addition to workload profiling, Alibaba performed dependency analysis to understand service interactions and failure propagation patterns. This involved mapping service call graphs and analyzing failure paths to identify critical dependencies and latent bottlenecks. Although not all performance metrics were disclosed, such analysis typically leverages profiling tools to uncover inefficiencies and streamline inter-service communication [13]. Common profiling techniques include sampling, instrumentation, and custom code injection—each providing varying levels of granularity and overhead. These tools are essential for in-depth introspection and performance tuning, helping organizations ensure that microservices remain responsive and resource-efficient under diverse operational conditions. Ultimately, profiling supports proactive system optimization, enabling continuous performance refinement in complex, distributed cloud environments.

Experimental Evaluation

Experimental evaluation is a critical research method used to assess the performance and scalability of Microservices Architecture (MSA) within controlled test environments. These evaluations are typically conducted in cloud-based testbeds, where researchers can simulate specific architectural configurations, scaling strategies, and optimization algorithms under reproducible conditions [4], [9]. One prominent example is Alibaba's empirical assessment of the **StatuScale** framework, which was deployed on Kubernetes and tested using real production workload traces. The experiments demonstrated significant performance enhancements, including response time reductions of up to 12.34% in the Sock-Shop application and 11.97% in Hotel-Reservation services. These results validated the framework's ability to optimize elastic scaling through resource-aware strategies [14]. Similarly, the SPIDEP platform underwent controlled spike testing to evaluate the comparative performance of MSA and Service-Oriented Architecture (SOA) implementations under sudden traffic surges. The experiments revealed that MSA outperformed SOA in response time and infrastructure efficiency, although at the cost of higher bandwidth consumption—a typical trade-off for enhanced responsiveness in distributed systems [12]. Another experimental study on a .NET Core-based microservices system evaluated system behavior under varying load conditions. The tests analyzed throughput, latency, and resource utilization across different scaling thresholds, offering actionable data for refining performance tuning and deployment strategies [15]. The advantage of experimental evaluation lies in its repeatability and control. Unlike live production systems, testbeds allow researchers to isolate variables, replicate scenarios, and gather quantifiable performance metrics. This level of control is essential for validating optimization techniques and ensuring that MSA implementations are robust, efficient, and adaptable before full-scale deployment.

Systematic Literature Reviews (SLRs)

Systematic Literature Reviews (SLRs) play a pivotal role in synthesizing empirical findings on Microservices Architecture (MSA), offering a comprehensive overview of the current research landscape. By categorizing existing studies, SLRs help identify key trends, unresolved challenges, and emerging research gaps, thereby guiding both academic inquiry and practical implementations [16], [4]. One such review focused on containerized microservices in cloud environments, revealing that most existing solutions addressed either load balancing or auto-scaling in isolation. Despite the interdependence of these mechanisms in ensuring Quality of Service (QoS) during peak traffic, integrated approaches remain underexplored—highlighting a gap in holistic system design [7]. A separate mapping study by Pezzè and Spina evaluated performance measurement techniques in cloud-based MSA deployments. It uncovered significant inconsistencies in benchmarking methodologies across studies, emphasizing the need for standardized performance evaluation models to enable valid cross-study comparisons [12]. Another review examined MSA deployments in multi-cloud environments. It identified a lack of formalized architectural patterns for managing system complexity, interoperability, and heterogeneity across cloud providers. Additionally, the study underscored rising security and compliance concerns, calling for the adoption of standardized frameworks to manage data visibility, authentication, and multi-cloud orchestration [16]. Collectively, these reviews offer valuable meta-analytical insights that extend beyond individual case studies. By consolidating empirical evidence, SLRs help shape research agendas, establish methodological rigor, and promote best practices in the development and evaluation of microservices systems.

Simulations

Simulations offer a cost-effective, risk-free approach for exploring microservices performance under hypothetical or extreme conditions that are difficult to replicate in production [17]. They are especially valuable in scenarios such as disaster recovery, traffic bursts, or resource failures. [17], proposed a runtime adaptation algorithm evaluated through a simulated email pipeline microservice. Using real traffic-inspired datasets and architectural modelling tools, the simulation demonstrated that global adaptation strategies outperformed local scaling in preventing cascading slowdowns and message losses under stress conditions. These findings underscore the utility of simulation-based experiments for testing advanced adaptation and scaling strategies in complex, distributed environments—without risking live system stability.

Key Performance Metrics for Evaluating MSA

To systematically assess the effectiveness of MSA in cloud computing environments, researchers employ several key performance metrics:

i. **Response Time:** The time taken by a service to process a request and return a response. Lower response times generally indicate better system performance [4].
ii. **Throughput:** The number of requests a system can handle within a specific time period. Higher throughput reflects greater efficiency and system capacity [4].
iii. **Latency:** The delay experienced in data transfer or communication between different microservices. Reducing latency is essential for maintaining optimal system responsiveness in distributed architectures [4].
iv. **Scalability:** The ability of the system to accommodate increasing workloads by scaling out (adding more instances) or scaling up (enhancing existing resources). Empirical research often evaluates how well MSA adapts to varying demands [18], [4], [5].
v. **Reliability:** The probability that a system will operate without failure for a specified period under given conditions. Reliability assessments focus on fault tolerance and resilience in MSA deployments [4].
vi. **Resource Utilization:** The efficiency with which computing resources (e.g., CPU, memory, network bandwidth) are allocated and consumed by microservices. Optimizing resource utilization is critical for cost efficiency in cloud environments [23], [7] and [3].
vii. **Failure Rate:** The frequency at which services or the overall system experience failures. Empirical evaluations assess system stability under different load conditions to determine robustness [20].

Empirical Frameworks for MSA in Cloud Computing

The evaluation of Microservices Architecture (MSA) in cloud computing environments relies on various empirical frameworks that integrate multiple research methodologies and performance metrics. These frameworks provide structured approaches to systematically assess MSA's effectiveness, particularly in terms of scalability, performance, reliability, and resource utilization. By leveraging empirical validation, researchers and practitioners can derive meaningful insights into the practical benefits and challenges of MSA implementation.

Performance Testing and Profiling Frameworks

Performance testing and profiling frameworks systematically analyze microservices-based applications to identify bottlenecks and assess scalability and resource consumption [4]. These frameworks define workload scenarios, utilize performance testing tools, and analyze collected metrics such as response time, throughput, and system latency [4]. Profiling techniques further investigate runtime behavior, focusing on CPU and memory utilization to optimize microservices performance and detect inefficiencies [4].

Scalability Assessment Frameworks

Scalability assessment frameworks measure how effectively an MSA deployment can handle increasing loads by gradually increasing concurrent users or transaction volumes and monitoring key performance indicators such as response time, throughput, and resource utilization [18], [4]. These frameworks also evaluate auto-scaling mechanisms to determine their efficiency in maintaining system performance under dynamic workload conditions [7], [13]. Empirical studies in this domain provide critical insights into the adaptability of microservices to cloud-native environments.

Fault Tolerance and Resilience Testing Frameworks

Fault tolerance and resilience testing frameworks focus on assessing the robustness of MSA systems in the presence of failures [4], [5]. These studies often involve injecting controlled faults into microservices or underlying cloud infrastructure to observe system recovery and service availability [4], [22]. Key performance metrics such as failure recovery time and overall system availability provide empirical evidence on the resilience of MSA deployments [20].

Key Insights from Studies on MSA

One of the most frequently observed advantages of MSA is its ability to enhance scalability and elasticity. Research indicates that microservices support independent scaling of components based on demand, resulting in more efficient resource utilization and improved responsiveness under fluctuating workloads [4], [5]. This adaptability is especially beneficial in cloud-native systems that require real-time performance adjustments [4]. Studies have also reported improved performance and responsiveness in MSA-based systems, particularly in complex or high-load scenarios. Fine-grained service modularity and optimized resource distribution contribute to reduced response times and increased throughput [4], [20]. However, the extent of these improvements varies based on architectural design, service granularity, and communication overhead [9]. Another major finding concerns deployment agility. The modularity of microservices often supported by containerization and CI/CD pipelines has been shown to facilitate frequent and independent releases. Organizations leveraging MSA report accelerated development cycles and improved responsiveness to business requirements [18], [21], [22]. Despite these benefits, empirical research has highlighted several persistent challenges. Monitoring and troubleshooting become increasingly complex in distributed microservices systems. Observability is hampered by the asynchronous nature of service interactions and the sheer number of moving parts [4], [22]. Effective logging, tracing, and diagnostics are essential to manage service dependencies and identify performance issues.

Communication overhead is another recurring concern. Unlike monolithic systems, where internal calls occur within a single process, MSA relies on network-based communication between services. This introduces latency and consumes additional resources [4], [10]. Research into API optimization, caching, and asynchronous messaging has helped address these inefficiencies, though implementation remains context-dependent [11]. A further operational complexity lies in resource management and provisioning. Coordinating multiple services requires intelligent load balancing and adaptive scaling strategies to prevent underutilization or contention. Studies have proposed AI-driven auto scaling models and Kubernetes-based orchestration frameworks to improve resource efficiency and ensure compliance with Quality of Service (QoS) standards [23], [7], [13].

Challenges and Limitations in Empirical Validation of MS

Although empirical research has significantly deepened understanding of Microservices Architecture (MSA) in cloud computing, several methodological and practical challenges continue to limit the generalizability and robustness of findings. One key challenge is the design of realistic experimental environments. Replicating the complexity of real-world cloud ecosystems requires emulating diverse workload patterns, intricate service dependencies, and dynamic scaling behaviors. These requirements make empirical testing resource-intensive and technically demanding [4], [9]. Another difficulty lies in isolating variables within distributed systems. Due to the highly interconnected nature of microservices, it is often challenging to attribute observed performance outcomes to specific architectural decisions or optimization strategies. Even when studies attempt to control for external factors, variations in infrastructure, platform configurations, and workload behavior frequently complicate interpretation [4], [9]. Generalizing empirical findings across different contexts also presents limitations. Differences in cloud service providers, deployment models, and industry-specific application types reduce the transferability of results. Consequently, insights derived from one study may not readily apply to alternative MSA deployments [4], [14]. A further methodological issue is the absence of standardized benchmarking frameworks. While traditional software performance testing benefits from established tools and reference workloads, MSA research often depends on custom experimental setups. These may include manually constructed Kubernetes clusters, container orchestration simulations, or internally developed test harnesses, which vary significantly across studies [4], [14]. As a result, comparing outcomes across empirical studies remains challenging. Though industry-wide efforts to standardize performance evaluation for cloud-native systems are ongoing, they are still in early development stages [4]. Finally, **cost and expertise constraints** frequently limit the scope and duration of empirical studies. Large-scale testing and profiling require substantial computational resources and cloud infrastructure, often incurring high operational expenses. Moreover, conducting rigorous long-term research demands specialized skills in areas such as distributed systems engineering, container orchestration, and cloud-native performance analysis. These requirements may limit research efforts in institutions with constrained resources [4].

CONCLUSION AND RECOMMENDATIONS

Empirical research has significantly advanced our understanding of Microservices Architecture (MSA) in cloud computing, revealing both its advantages and challenges. While studies consistently validate MSA's benefits in scalability, agility, and performance optimization, they also highlight complexities in areas such as monitoring, inter-service communication overhead, and resource management [4]. This growing body of evidence underscores the importance of continued research to address these challenges and refine best practices for MSA adoption in cloud environments. Despite the valuable insights gained, several critical areas warrant further empirical exploration. Key research gaps include the development of standardized benchmarking frameworks tailored to MSA performance evaluation. The lack of widely accepted benchmarks complicates the comparison of results across studies and hinders the establishment of performance baselines [4] [14]. Addressing this gap will enhance research consistency and enable more meaningful comparisons between different MSA implementations. Another area that requires attention is the empirical evaluation of MSA in multi-cloud and hybrid-cloud environments. As organizations increasingly adopt multi-cloud strategies to improve redundancy and flexibility, understanding the performance trade-offs and operational complexities of deploying MSA across multiple cloud providers becomes crucial [16]. Empirical studies should focus on factors like latency, data consistency, and inter-cloud networking challenges to guide best practices for hybrid-cloud MSA deployments [16]. Further investigation into security and compliance concerns is also essential. Although security is well-researched in cloud computing, empirical studies on MSA's security vulnerabilities and the effectiveness of security measures remain scarce [16], [24]. Future research should examine authentication mechanisms, API security protocols, and the impact of encryption and access controls on MSA performance. Cost optimization strategies for cloud-based MSA also require further empirical validation. While studies have explored various scaling policies, resource provisioning techniques, and serverless computing approaches, more research is needed to assess their long-term impact on cost efficiency and performance [23], [20] , [24], [5] [13]. Such studies will provide organizations with the insights needed to optimize cloud expenditure without sacrificing system performance. Emerging technologies, including AI/ML, edge computing, and serverless architectures, present new opportunities for MSA but also introduce unique challenges. Empirical research is needed to investigate how AI-driven automation can enhance microservices management, how edge computing can reduce latency for MSA workloads, and how serverless paradigms impact microservice performance [16], [25], [23],

[23]. Lastly, longitudinal studies tracking the evolution of MSA over time could provide valuable insights into the long-term effects of transitioning from monolithic architectures to microservices. These studies will help both researchers and practitioners understand the impact of architectural decisions on system maintainability, scalability, and performance over extended periods [20], [9].

REFERENCES

1. Dahal, S. B. (2023). Architecting microservice frameworks for high scalability: designing resilient, Performance-Driven, and Fault-Tolerant systems for modern enterprise applications. *Journal of Intelligent Connectivity and Emerging Technologies*, *8*(4), 58–70.
2. Oyeniran, N. O. C., Adewusi, N. a. O., Adeleke, N. a. G., Akwawa, N. L. A., & Azubuko, N. C. F. (2024). Microservices architecture in cloud-native applications: Design patterns and scalability. *Computer Science & IT Research Journal*, *5*(9), 2107–2124. https://doi.org/10.51594/csitrj.v5i9.1554
3. Sigala, N. S. (2025). Microservices Architecture in Cloud Computing: A Software engineering perspective on design, deployment, and management. *International Journal of Research and Innovation in Social Science*, *IX*(XV), 215–239. https://doi.org/10.47772/ijriss.2025.915ec0013
4. Desina, G. C. (2023). Evaluating the impact of Cloud-Based microservices architecture on application performance. *arXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2305.15438
5. Sudhir Samant, P. (2024). Microservices in the cloud: enabling scalability, flexibility, and rapid deployment. *Journal of Advanced Research Engineering and Technology (JARET)*, *3*(1), 1–10. https://iaeme.com/Home/issue/JARET?Volume=3&Issue=1
6. Seroukhov, S. (2024). *Microservices Design Patterns with Java: 70+ patterns for designing, building, and deploying microservices (English Edition)*. BPB Publications.
7. Rabiu, S., Yong, C. H., & Mohamad, S. M. S. (2022). A Cloud-Based Container Microservices: A review on Load-Balancing and Auto-Scaling issues. *International Journal on Data Science*, *3*(2), 80–92. https://doi.org/10.18517/ijods.3.2.80-92.2022
8. Thatikonda, V. K. (2023). Assessing the impact of microservices architecture on software maintainability and scalability. *European Journal of Theoretical and Applied Sciences*, *1*(4), 782–787. https://doi.org/10.59324/ejtas.2023.1(4).71
9. Pandiya, D. K. (2022). Performance Analysis of Microservices Architecture in Cloud Environments. *International Journal on Recent and Innovation Trends in Computing and Communication*, *10*(12), 264–274.
10. Pankaj, S. (2024). Mastering Microservices Architecture and Cloud Computing : In-Depth Strategies for Domain-Specific Optimization. *International Journal of Scientific Research in Computer Science Engineering and Information Technology*, *10*(5), 813–821. https://doi.org/10.32628/cseit241051071
11. Walia, C. (2023). *Mastering Cloud-Native microservices: Designing and implementing Cloud-Native Microservices for Next-Gen Apps (English Edition)*. BPB Publications.
12. Calderón-Gómez, H., Mendoza-Pittí, L., Vargas-Lombardo, M., Gómez-Pulido, J. M., Rodríguez-Puyol, D., Sención, G., & Polo-Luque, M. (2021). Evaluating Service-Oriented and microservice architecture patterns to deploy eHealth applications in cloud computing environment. *Applied Sciences*, *11*(10), 4350. https://doi.org/10.3390/app11104350
13. Xu, M., Yang, L., Wang, Y., Gao, C., Wen, L., Xu, G., Zhang, L., Ye, K., & Xu, C. (2023). Practice of Alibaba cloud on elastic resource provisioning for large-scale microservices cluster. *Software Practice and Experience*, *54*(1), 39–57. https://doi.org/10.1002/spe.3271
14. Wen, L., Xu, M., Gill, S. S., Hilman, M. H., Srirama, S. N., Ye, K., & Xu, C. (2025).
15. Rafi, H., Mohammed, A., & Kumar, R. (2025). Optimizing web application performance: evaluating microservices architectures in .net core for scalability and efficiency. *Policy Research Journal*, *3*(1), 77–82. https://policyresearchjournal.com/index.php/1/article/view/277
16. Alonso, J., Orue-Echevarria, L., Casola, V., Torre, A. I., Huarte, M., Osaba, E., & Lobo, J. L. (2023). Understanding the challenges and novel architectural models of multi-cloud native applications – a systematic literature review. *Journal of Cloud Computing Advances Systems and Applications*, *12*(1). https://doi.org/10.1186/s13677-022-00367-6
17. Bacchiani, L., Bravetti, M., Giallorenzo, S., Mauro, J., Talevi, I., & Zavattaro, G. (2021). Microservice Dynamic Architecture-Level Deployment Orchestration. In *Lecture notes in computer science* (pp. 257–275). https://doi.org/10.1007/978-3-030-78142-2_16
18. Artiom, K. (2025). Microservices Architecture: Accelerating feature development and scalability through monolith decomposition. *International Journal of Engineering and Computer Science*, *14*(01), 26744–26750. https://doi.org/10.18535/ijecs/v14i01.4958
19. StatuScale: Status-aware and elastic scaling strategy for microservice applications. *ACM Transactions on Autonomous and Adaptive Systems*. https://doi.org/10.1145/3686253

20. Mulahuwaish, A., Korbel, S., & Qolomany, B. (2022). Improving datacenter utilization through containerized service-based architecture. *Journal of Cloud Computing Advances Systems and Applications*, *11*(1). https://doi.org/10.1186/s13677-022-00319-0
21. Kambala, G. (2025). Integration Of Microservices And Cloud Computing: A Paradigm Shift In Enterprise Application Design. *International Journal of Creative Research Thoughts (IJCRT)*, *13*(2), a333–a350.
22. Walia A, Shew MA, Varghese J, Ioerger P, Lefler SM, Ortmann AJ, Herzog JA, Buchman CA. Improved cochlear implant performance estimation using tonotopic-based electrocochleography. JAMA Otolaryngology–Head & Neck Surgery. 2023 Dec 1;149(12):1120-9.
23. Mahesar, A. R., Li, X., & Sajnani, D. K. (2024). Efficient microservices offloading for cost optimization in diverse MEC cloud networks. *Journal of Big Data*, *11*(1). https://doi.org/10.1186/s40537-024-00975-w
24. Singh, S. S. G. (2024). Microservices security challenges and solutions in cloud environment. *International Journal of Science and Research (IJSR)*, *13*(3), 201–205. https://doi.org/10.21275/sr24303144734
25. Busi, S. P. (2025). Next-Generation cloud Computing: integration of microservices, edge computing, and emerging technologies. *International Journal of Computer Engineering and Technology (IJCET)*, *16*(1), 1848–1862. https://doi.org/10.34218/ijcet_16_01_134
26. Orue-Echevarria L, Casola V, Torre AI, Huarte M, Osaba E, Lobo JL. Understanding the challenges and novel architectural models of multi-cloud native applications–a systematic literature review. Journal of Cloud Computing. 2023 Jan 12;12(1):6.
27. Casola MA. *Single-Subject Writing Strategy Instruction: A Meta-Analysis* (Master's thesis, The University of Western Ontario (Canada)).

CITE AS: Godspower I. Akawuku, Onyinyechi H. Ikediego and Nwankwo Chekwube (2025). Software Engineering Dimensions: Empirical Frameworks on Microservices Architecture in Cloud Computing Domains. NEWPORT INTERNATIONAL JOURNAL OF SCIENTIFIC AND EXPERIMENTAL SCIENCES,6(2):37-43. https://doi.org/10.59298/NIJSES/2025/62.374300