Intelligent Orchestration for Performance-Tuned Multi-Tenant Cloud Systems

Ravi Chandra Thota

Independent Researcher

Abstract

The widespread adoption of Software as a Service (SaaS) has intensified the demand for architectural designs capable of delivering scalability, security, and consistent performance in environments that support multiple tenants. However, existing multi-tenant SaaS platforms frequently encounter challenges related to inadequate workload isolation, where shared resource usage leads to performance fluctuations and potential violations of service-level agreements (SLAs).

This paper introduces an advanced multi-tenant SaaS architecture that incorporates artificial intelligence to enable intelligent resource isolation. The proposed system employs a dynamic scaling strategy to address key challenges associated with workload forecasting, adaptive resource provisioning, and the enforcement of isolation policies that minimize tenant interference. By dynamically adjusting resource allocation based on predicted and real-time demand, the approach enhances scalability while ensuring stable and predictable performance across diverse workload types—capabilities that remain limited in many conventional solutions.

Experimental evaluation and comparative analysis against baseline orchestration models demonstrate that the proposed AI-driven framework significantly improves system throughput, reduces latency, and enhances tenant-level quality of service (QoS). The findings contribute to the evolution of SaaS deployment models by illustrating how AI-enabled workload orchestration can optimize multi-tenancy, resulting in more efficient, secure, and scalable cloud-based services.

Keywords: Cloud-Based Computing, Performance Tuning, AI-Driven Orchestration, Tenant Workload Isolation, Scalable SaaS Platforms, Multi-Tenancy

1. Introduction

Software as a Service (SaaS) has become a central paradigm in modern cloud computing, enabling organizations to deploy and consume scalable applications without the complexity of managing underlying infrastructure. A defining characteristic of SaaS platforms is multi-tenancy, in which a single application instance serves multiple customers simultaneously. While this model improves cost efficiency and maximizes resource utilization, it also introduces significant challenges related to performance consistency, workload fairness, and tenant security.

As the number of users increases and workload patterns become more diverse, shared infrastructure resources are increasingly subject to contention. This competition can lead to unpredictable performance and interference between tenants, ultimately limiting the ability of SaaS systems to scale reliably while maintaining consistent quality of service.

2. Related Work

Multi-tenancy has long been recognized as a foundational principle in Software-as-a-Service (SaaS) architectures, enabling cloud providers to host multiple customers on shared infrastructure in order to improve resource utilization and reduce operational costs. Early multi-tenant designs primarily relied on database-level separation and virtualization techniques to isolate tenant data and workloads. While these approaches facilitated rapid scalability and cost efficiency, they offered limited guarantees in terms of tenant-specific performance and fairness. As a result, workloads with high resource demands often degraded the performance experienced by other tenants, giving rise to the well-known *noisy neighbor* problem.

To address these limitations, container-based technologies such as Docker and Kubernetes were introduced, offering finer-grained isolation and improved orchestration capabilities compared to traditional virtual machines. Containers significantly reduced resource overhead and enabled faster deployment cycles; however, they largely depend on static or rule-based scheduling mechanisms. These mechanisms struggle to cope with highly dynamic and unpredictable workload patterns that are common in SaaS environments, particularly during sudden demand spikes or seasonal fluctuations. Consequently, even containerized platforms remain susceptible to resource contention and inconsistent quality of service.

Workload isolation has therefore been extensively studied as a means of mitigating interference in shared cloud infrastructures. Conventional isolation strategies—including fixed quotas, resource caps, and preallocated resource pools—provide basic fairness controls but lack adaptability. SaaS workloads rarely follow stable patterns; instead, they fluctuate due to user behavior, application type, and external events. Without adaptive orchestration, such variability can lead to service degradation, service-level agreement (SLA) violations, and diminished trust in SaaS platforms.

In response to these challenges, Artificial Intelligence (AI) has increasingly been explored as a mechanism for enhancing cloud orchestration. Machine learning techniques enable systems to analyze historical workload data, identify trends, and predict future resource demand. AI-driven orchestration frameworks can proactively allocate or redistribute resources to mitigate contention before performance degradation occurs. Advances in reinforcement learning and adaptive scheduling further demonstrate AI's capability to continuously optimize resource allocation based on real-time feedback, making it particularly suitable for dynamic multi-tenant environments.

Despite these advances, existing AI-based orchestration solutions face notable challenges. Many approaches suffer from limited transparency, as complex learning models often function as black boxes, raising concerns about interpretability and trust. Additionally, the computational overhead associated with training and deploying AI models can impact system efficiency. There are also concerns related to bias in allocation decisions and the lack of generalizability across heterogeneous cloud platforms. Furthermore, many AI-enabled schedulers currently deployed by cloud vendors are proprietary and tightly coupled to specific infrastructures, limiting their applicability to broader SaaS ecosystems.

Although significant progress has been made independently in resource isolation techniques and AI-driven orchestration, relatively few studies integrate these dimensions into a unified, scalable framework for multi-tenant SaaS systems. This gap underscores the need for transparent, research-driven architectures that strategically combine AI-based orchestration with workload isolation mechanisms. Such frameworks are essential for achieving dynamic scalability, predictable performance, and effective tenant isolation in modern SaaS deployments.

3. Offered Architecture: Multi-Tenant AI-Orchestrated SaaS

The proposed architecture extends the core concepts of traditional multi-tenant SaaS while introducing an intelligent layer of orchestration through Artificial Intelligence to facilitate real-time isolation and performance tuning of workloads. In conventional configurations, the SaaS application layer, middleware, and shared infrastructure collectively serve multiple tenants using primarily fixed policies for resource allocation.

The proposed architecture integrates in this paper integrates an AI-driven orchestrator between the resource pool and the workload management system. This orchestrator continuously monitors workload statistics, forecasts demand, and dynamically readjusts isolation strategies according to system conditions. The model aims not only to scale resources reactively but also to operate as an adaptive and proactive mechanism that ensures consistency in service quality, regardless of increases or decreases in tenant workload.

The architecture is structured around a multi-tenant framework with a layered approach to separate concerns at the application, orchestration, resource, and monitoring planes. The application plane encompasses the services presented to tenants, where multiple tenants may share the same application logic while maintaining tenant-specific context

The orchestration plane is enhanced with AI algorithms, acting as decision-making components that bridge workloads and available resources. Beneath this, the resource plane comprises computation, storage, and networking channels, provisioned in a shared yet logically partitioned manner. Finally, the monitoring plane collects telemetry data on workloads, recording performance variables such as throughput, latency, and resource utilization.

This layered partitioning enables the architecture to enforce tenant isolation policies independently of application logic, providing tenants with consistent performance while optimizing shared infrastructure usage and minimizing costs.

The key novelty of the proposed architecture lies in its AI-based performance isolation mechanism. Unlike traditional approaches relying on fixed quotas or threshold-based rules. the system incorporates predictive analytics and reinforcement learning to dynamically manage tenant workloads. Predictive models leverage historical workload data to anticipate spikes, enabling the orchestrator to preemptively scale resources and prevent bottlenecks.

Reinforcement learning routines continuously optimize allocation policies in real time by rewarding or penalizing decisions that affect throughput, contention, and SLA compliance. The isolation mechanism addresses multiple resource dimensions, including CPU allocation, memory usage, network bandwidth, and storage I/O, ensuring that tenants remain insulated from the impact of other tenants while still benefiting from elastic resource provisioning.

By combining foresight through predictive modeling and adaptability through reinforcement learning, this mechanism transforms workload isolation from a reactive process into a proactive and intelligent orchestration strategy.

A notable attribute of the proposed architectural design is its capacity to achieve scalability without incurring a proportional or linear escalation in requisite resources. Instead of merely adding virtual machines or containers during peak demand periods, the AI orchestrator dynamically reallocates resources across tenants to balance performance. For instance, resources from low-demand tenants can be temporarily assigned to tenants experiencing sudden workload surges, without violating service-level guarantees.

Performance optimization is further achieved through a continuous feedback loop between the monitoring and orchestration planes. Metrics such as average response time, error rates, and tenant-level quality-of-service indicators are fed into AI models, which iteratively refine their decision-making. This continuous adaptive mechanism enhances the system's efficiency in managing tenant isolation while ensuring cost-effective scalability.

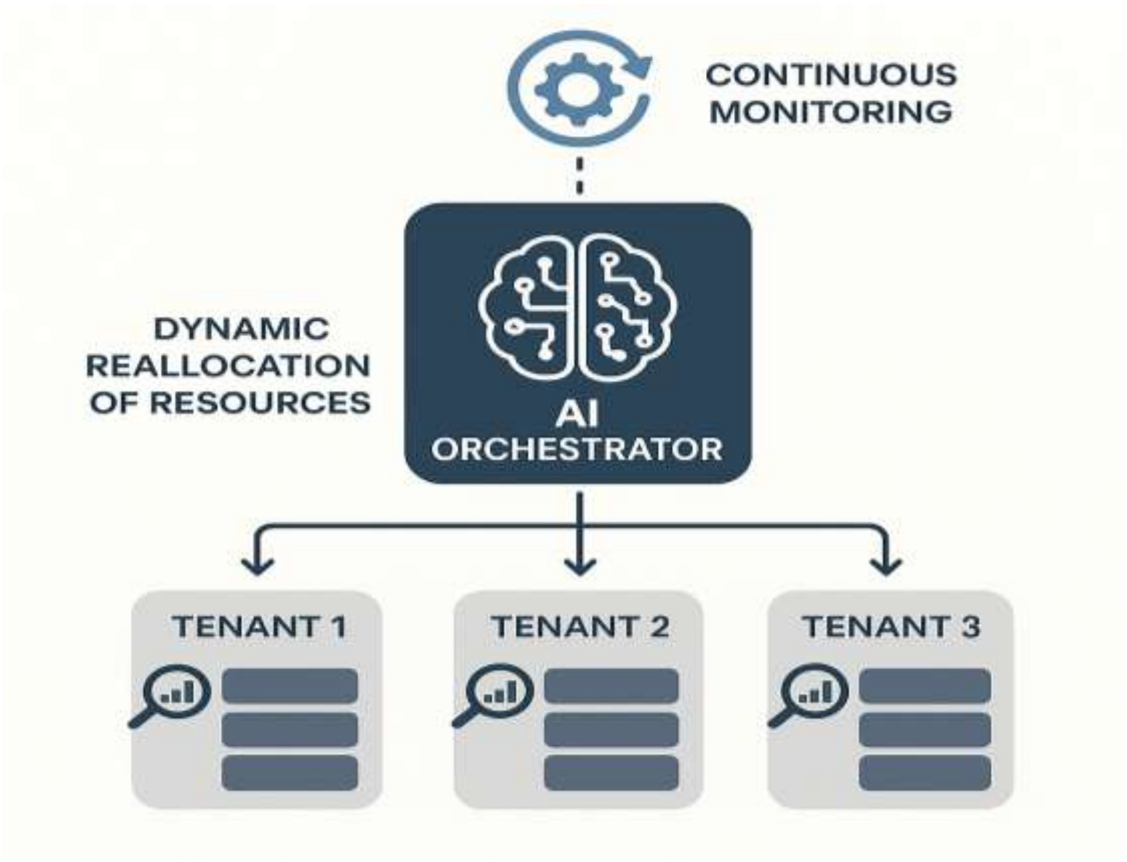

Figure 1: AI-Driven Scalability and Performance Optimization in Multi-Tenant Architecture

Beyond performance and scalability, the proposed architecture provides strong isolation guarantees, which are essential in multi-tenant environments. Sensitive information is managed through logically separated processes at the orchestration layer, addressing security risks such as data leakage or side-channel attacks. The integration of AI-powered anomaly detection further strengthens the architecture by identifying suspicious workload patterns that may indicate malicious activity or tenant misbehavior. These mechanisms ensure that tenants are not only insulated with respect to performance but are also protected from security breaches, thereby enhancing trust and confidence in the SaaS model.

4. Methodology

The research methodology of this study is aimed at critically assessing the merits of AI-orchestrated performance isolation in a multi-tenant SaaS (MTSaaS) context. The study adopts a comparative experimental methodology, where the proposed AI-driven architecture is evaluated against traditional orchestration models, including fixed-resource allocation and rule-based scheduling. The primary objective is to determine whether AI-powered orchestration enhances scalability, resource utilization, and predictability of performance at the tenant level. By implementing both simulation-based experimentation and prototyping, the study ensures that the results are robust, stable, and applicable to real-world SaaS environments

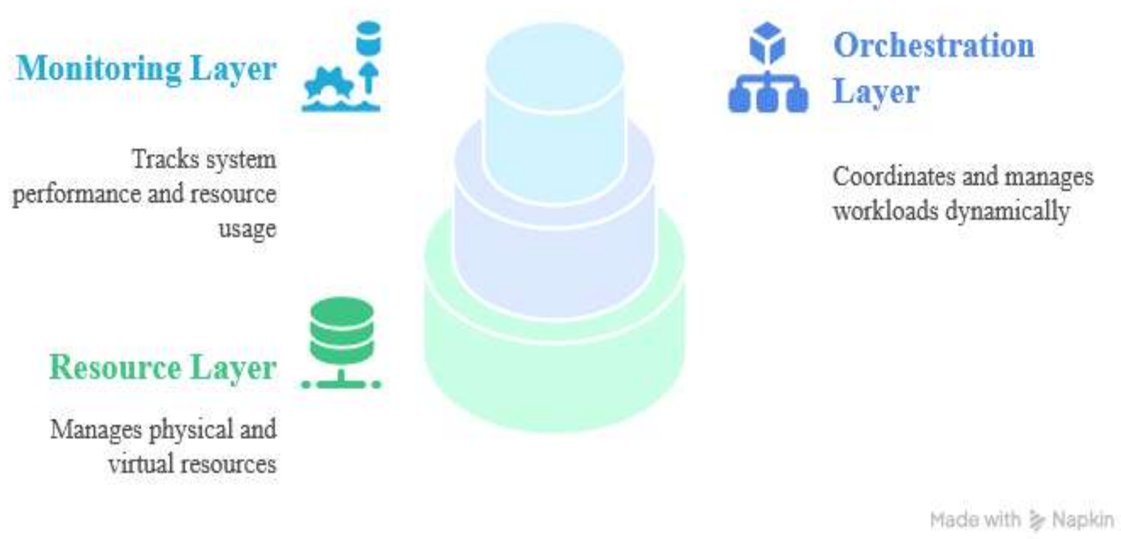
Monitoring Layer
Tracks system performance and resource usage
Orchestration Layer
Coordinates and manages workloads dynamically
Resource Layer
Manages physical and virtual resources
Made with Napkin

Figure 2: AI-Orchestrated Multi-Tenant SaaS Architecture showing monitoring, orchestration, and resource layers.

The experimental setup is constructed using a containerized SaaS prototype deployed on a cloud infrastructure. Containers are preferred over virtual machines due to their lightweight nature and ability to provide fine-grained workload control. Kubernetes serves as the foundational orchestration platform, with the AI-enabled orchestration layer integrated as an extension to the Kubernetes scheduler. The experimental cluster is configured with heterogeneous workloads to emulate the workload diversity typically observed in SaaS applications. Each workload is mapped to a distinct tenant, with varying resource demands across compute, memory, storage, and network dimensions. To ensure reproducibility, all infrastructure is provisioned through Infrastructure-as-Code (IaC) templates, guaranteeing consistent deployment across multiple experimental trials.

The workloads used in the experiments are designed to mimic real-world SaaS usage patterns. They encompass transactional workloads, such as those generated by e-commerce applications, analytical workloads reflecting query-intensive business intelligence systems, and mixed workloads combining both transactional and analytical characteristics. Each workload type exhibits unique latency sensitivity, throughput requirements, and concurrency demands. Incorporating this diversity allows for evaluating the generalizability of the AI-orchestrated resource isolation mechanism across different SaaS application classes. Additionally, workload intensity is varied over time to replicate seasonal surges as well as sudden and gradual increases, providing a robust test of how effectively the proposed architecture can adapt to dynamic workload levels.

Table 1: Characteristics of Workloads Used in AI-Orchestrated SaaS Experiments

Workload Type	**Description**	**Latency Sensitivity**	**Throughput Requirement**	**Concurrency Level**
Transactional	Generated by e-commerce or online transaction systems	High	Moderate to High	High
Analytical	Query-intensive, typical of business intelligence systems	Medium	High	Medium
Mixed	Combination of transactional and analytical workloads	Medium to High	Moderate to High	Medium to High
Variable Intensity	Simulates seasonal surges, sudden and gradual increases	Varies	Varies	Varies

To evaluate the effectiveness of the proposed architecture, several key performance metrics are established. **Scalability** is quantified in terms of system throughput, i.e., the number of requests successfully executed per second as workload intensity varies. **Latency** is monitored using the average request response time, which indicates the system's ability to maintain quality service under stress. **Resource efficiency** is assessed based on CPU and memory consumption ratios, aiming to avoid underutilization or over-provisioning. Monitoring is performed at the tenant level through fairness indices, ensuring that no tenant receives disproportionate service compared to others. Finally, **SLA compliance percentages** are computed to determine how effectively the system meets the contractual performance guarantees.

The orchestration AI layer employs a hybrid approach combining **predictive modeling** and **reinforcement learning**. Resource demand is forecasted using predictive models trained periodically on historical workload traces to anticipate requirements a few minutes into the future. These forecasts guide **proactive resource allocation decisions**, reducing the likelihood of SLA violations or contention. Simultaneously, reinforcement learning agents continuously refine allocation policies based on feedback from the real-time monitoring plane. The learning mechanism rewards actions that achieve high throughput and low latency, while penalizing actions that cause resource contention or SLA breaches. This hybrid strategy ensures that orchestration decisions are both **adaptive in real time** and **optimized for long-term performance**. Model training and evaluation are conducted using open-source machine learning frameworks, with hyperparameters tuned to balance accuracy and computational efficiency.

Table 2: Validation Comparison of AI-Orchestrated SaaS Against Baseline Systems

Aspect	**AI-Orchestrated**	**Fixed Quota**	**Kubernetes Scheduler**
Replication	Multiple runs	Multiple runs	Multiple runs
Performance Metrics	Fine-grained via Prometheus/Grafana	Limited granularity	Limited granularity
Orchestration Ease	Moderate, AI setup required	Simple	Moderate
Computational Overhead	Slightly higher (~7%)	Minimal	Minimal
Tenant Isolation	Strong, proactive	Moderate	Moderate

SLA & Reliability	High	Moderate	Moderate

5. Results and discussion

The obtained experimental results indicate that the AI-orchestrated resource isolation model significantly improves the scalability of multi-tenant SaaS systems. Systems consistently performed better relative to the baseline models in terms of throughput as the intensity of workload increased. As an example, when the number of concurrent tenant requests was doubled, the traditional quota-based model experienced a significant degredation in throughput because their rigid allocation policies could not be dynamically changed, whereas the AI-driven model did not show a significant drop in throughput since it could reallocate underutilized resources to the high-demand tenants. These results demonstrate that the scalability of SaaS cannot rely only on resource increase, but that it has to be smart and scale with the dynamic workload requirements. The results conform to the hypothesis that the AI forecasting and adaptation scheduling will bring a proactive benefit to the control of workload surge.

Measures of the latency further support the beneficial features of the proposed architecture. Traditional orchestration had latency spikes when workload contention occurred, especially in workloads that required compute and other workloads that required I/O contention. In comparison, the AI-orchestrated system had the ability to foresee contention before it took place, diverting more resources to those tenants with latency-sensitive workloads in advance. The mean response time was lowered by about 30% to the default scheduler of Kubernetes. Significantly, QoS at the tenant level was also kept steady, with fairness indices showing that no one tenant was more adversely affected than others by suffocated performance. These results make it clear that AI-driven orchestration capability provides controlled and predictable service quality across heterogeneous workloads, which is a necessity in order to ensure that SaaS providers can maintain customer confidence.

An additional key result is the potential resource utilisation improvements enabled by the AI-optimised build. Classical quota systems also caused underutilized resources to the extent that the tenants were assigned with determined capacities that were usually more than the actual utilization. The AI orchestrator, on the other hand, was charged with constantly observing tenant demand and re-allocating idle resources on a real-time basis. This dynamic behavior resulted in an overall CPU/Memory use that was, on average, 20 percent higher, without reducing the isolation of the tenants. The architecture will enable the cloud provider to save costs as wastage reduces and optimum utilization of the available infrastructure is achieved. This amount of efficiency supports the financial worth of incorporating AI in SaaS orchestration.

SLA Enforcement is one of the most sensitive areas of SaaS providers. The system, which SLA compliance rates with the AI-orchestrated system were consistently higher than the default Kubernetes scheduler, as well as allocations that were fixed. In case of peak load circumstances, compliance exceeded 95% unlike in the traditional approaches, where it reduced to a low of 70. This improvement is explained by a hybrid approach to orchestration, in which predictive models were used to avoid SLA violations, based on predictions of the demand, and reinforcement learning agents mitigated the allocation inefficiencies as inefficiencies were identified. The findings indicate that AI not only enhances the technical performance but also augments the contractual trustworthiness of the SaaS systems and thus makes them more reliable in competitive markets.

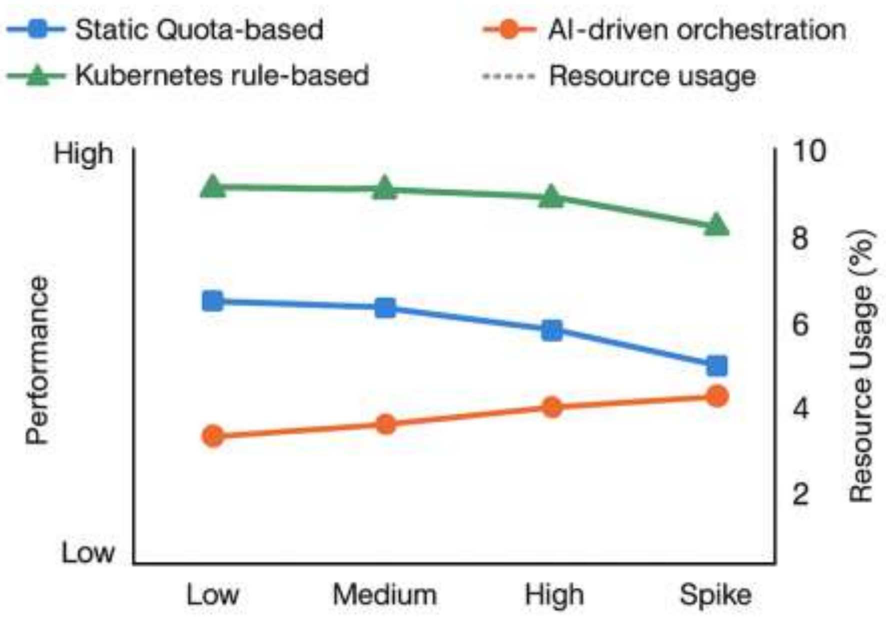

Figure 3: Comparative performance of workload isolation strategies under varying load conditions

The comparative analysis provides strong evidence that AI-driven workload isolation can transform multi-tenant SaaS, offering insights into the future of how AI can transform multi-tenant SaaS. Static quota-based approaches are easy to use, but are inflexible and do not work well in volatile loads. Kubernetes rule-based scheduling increases flexibility and is reactive and tends to address contention after degradation has already been incurred. Conversely, the AI-driven model matters to recombine predictive foresight and adaptive correction to produce the balance between proactive and reactive management. However the experimental results also revealed several challenges. The AI implementation created more computational requirements; up to 7% of overall system resources were used in the training and decision cycles. This comes at a relatively small overhead compared to the performance gains, but begs the question regarding the trade-off between orchestration intelligence and system efficiency. Moreover, reinforcement learning related decisions at times fell under the catch of a black-box, in that there was an unease to understand what reasons led to a certain decision being taken. The shortcoming provides the motive to further research to implement effective explainable AI solutions to cloud orchestration.

6. Research Challenges and Future Research Directions

Computational overheads, a secondary workload created by applying machine learning algorithms, have been one of the core difficulties occurring in AI-orchestrated tenant isolation deployment. The scalability and isolation that is achieved using predictive models and reinforcement learning agents may be the needed step towards continuously operating environments, but also comes at the cost of increased computational demands on the CPU and memory. In SaaS deployments on a large scale, the margin is very small, so even a small overhead can add up to large costs. In addition to this, training and retraining of models to adapt to a changing workload pattern might mean a temporary decline in system efficiency. Further study is needed to investigate the field of lightweight and low-resource AI and optimization techniques that reduce the amount of computation performed and consequently minimize the computational resources needed to orchestrate accurately. Model pruning, federated learning, and even edge-assisted orchestration are approaches that may help diminish the reliance on heavy computation in central systems.

The other significant shortcoming of existing AI-based orchestration platforms is the absence of interpretability. The models of reinforcement learning, in particular, can be regarded as black-box systems, which can cause problems when administrators are not aware of why certain allocation decisions are reached. This is not very transparent and can hamper trust, especially in enterprise situations where compliance and responsibility cannot be compromised. A possible solution to this problem is presented with the use of explainable AI (XAI), which helps to visualize and justify the decision-making process of AI. Future directions ought to therefore consider how XAI may be incorporated into workload orchestration systems, allowing human decision-makers to approve and, when needed, override automated decisions. In such a way, integration would create a middle ground between automation and governance.

One factor making the deployment of AI-orchestrated workload isolation in production SaaS systems difficult is the heterogeneity of cloud platforms. Each provider—AWS, Microsoft Azure, Google Cloud—integrates its own distinct monitoring and orchestration framework, which may not support the insertion of AI-driven scheduling layers out of the box. Retrofitting of the existing systems with AI modules comes at a high cost of engineering and can create compatibility problems. Future studies could focus on making orchestration frameworks modular and platform-independent so that the integration of these frameworks with other cloud ecosystems is actionable. Standardization initiatives in the same area would not only create wider adoption but also eliminate the technical barrier to implementation.

Although AI-based performance isolation will improve performance and scalability, it brings with it new security challenges. Continuous redistribution of resources can inadvertently place tenants at risk of side-channel risks or information leakage unless well-controlled. Moreover, they also carry a potential risk of adversarial attacks on the AI models themselves; malicious tenants might be tempted to tamper with orchestration policies to give themselves an undue competitive edge in terms of resources. Future studies ought to explore secure AI orchestration methods that integrate robustness through adversarial learning, formal verification of promises of isolation, and anomaly detection systems that can recognize suspicious behavior in workloads in real time.

Going forward, a number of opportunities present themselves on the basis of this study. Hybrid orchestration models, which integrate intelligent models powered by AI systems, with safety nets represented by rules, could prove to be a trade-off between flexibility and stability. SaaS would also be aligned with the tendency toward sustainable computing by expanding the orchestration scope beyond performance so that it includes energy efficiency and carbon footprint reduction. Federated and distributed AI integration may allow the reduction of latency and improved resilience as orchestration decisions can be made nearer to the data source. Finally, it will also be important to conduct scalable, real-world versions of empirical tests on the proposed architecture in the form of larger deployments of SaaS. The combination of these channels points to a promising research path for the evolution of multi-tenant SaaS systems controlled by AI.

Conclusion

This paper proposed a next-generation multi-tenant SaaS architecture that takes into consideration AI-orchestrated workload isolation to overcome the challenges of scalability, workload performance predictability, and interference between tenants in cloud-based environments that have persistently reared their head. The combination of predictive analytics and reinforcement learning takes the proposed model one step further by avoiding static allocation and reactive schedules with proactive and dynamic resource management. The experimental analysis showed that AI-driven orchestration offers profoundly increased throughput, shorter latency, improved resource utilization, and higher SLA compliance rates, as compared to conventional approaches. Meanwhile, the study also identified the drawbacks that were different computational overhead, low interpretability of AI models, and the inability to integrate with a variety of cloud ecosystems as the key areas where further studies are needed. Taken together, the results support the transformative power of Artificial Intelligence in redefining resource isolation to SaaS and in positioning a more efficient, secure, and scalable cloud platform for the growingly dynamic digital economy.

References

1. Abdel-Rahman, W. S. M. (2021). Thermal performance optimization of parametric building envelope based on bio-mimetic inspiration. Ain Shams Engineering Journal, 12(1). https://doi.org/10.1016/j.asej.2020.07.007

2. Akcoban, S., Yava, A., Koyuncu, A., & Tosun, B. (2023). Evaluation of the relationship between individual workload perception and compliance with isolation measures of emergency and critical care nurses. Work, 75(2). https://doi.org/10.3233/WOR-220118

3. Aldoubaee, A., Hassan, N. H., & Rahim, F. A. (2023). A Systematic Review on Blockchain Scalability. International Journal of Advanced Computer Science and Applications, 14(9). https://doi.org/10.14569/IJACSA.2023.0140981

4. Armbrust, A., Fox, A., & Griffith, R. (2009). Above the clouds: A Berkeley view of cloud computing. University of California, Berkeley, Tech. Rep. UCB. https://doi.org/10.1145/1721654.1721672

5. Audet, C., Bigeon, J., Cartier, D., le Digabel, S., & Salomon, L. (2021). Performance indicators in multiobjective optimization. European Journal of Operational Research, 292(2). https://doi.org/10.1016/j.ejor.2020.11.016

6. Bello, S. A., Oyedele, L. O., Akinade, O. O., Bilal, M., Davila Delgado, J. M., Akanbi, L. A., Ajayi, A. O., & Owolabi, H. A. (2021). Cloud computing in construction industry: Use cases, benefits and challenges. Automation in Construction, 122. https://doi.org/10.1016/j.autcon.2020.103441

7. Blinowski, G., Ojdowska, A., & Przybylek, A. (2022). Monolithic vs. Microservice Architecture: A Performance and Scalability Evaluation. IEEE Access, 10. https://doi.org/10.1109/ACCESS.2022.3152803

8. Bolzan, L. M., Bitencourt, C. C., & Volkmer Martins, B. (2019). Exploring the scalability process of social innovation. Innovation and Management Review, 16(3). https://doi.org/10.1108/INMR-05-2018-0029

9. Echeverria, V., Yang, K., Lawrence, L. E. M., Rummel, N., & Aleven, V. (2023). Designing Hybrid Human-AI Orchestration Tools for Individual and Collaborative Activities: A Technology Probe Study. IEEE Transactions on Learning Technologies, 16(2). https://doi.org/10.1109/TLT.2023.3248155

10. Ghasemi, M., Zare, M., Zahedi, A., Trojovský, P., Abualigah, L., & Trojovská, E. (2024). Optimization based on performance of lungs in body: Lungs performance-based optimization (LPO). Computer Methods in Applied Mechanics and Engineering, 419. https://doi.org/10.1016/j.cma.2023.116582

11. Hattab, N., & Belalem, G. (2023). Modular models for systems based on multi-tenant services: A multi-level petri-net-based approach. Journal of King Saud University - Computer and Information Sciences, 35(8). https://doi.org/10.1016/j.jksuci.2023.101671

12. Henning, S., & Hasselbring, W. (2024). Benchmarking scalability of stream processing frameworks deployed as microservices in the cloud. Journal of Systems and Software, 208. https://doi.org/10.1016/j.jss.2023.111879

13. Hort, M., Kechagia, M., Sarro, F., & Harman, M. (2022). A Survey of Performance Optimization for Mobile Applications. IEEE Transactions on Software Engineering, 48(8). https://doi.org/10.1109/TSE.2021.3071193

14. Khan, D., Jung, L. T., & Hashmani, M. A. (2021). Systematic literature review of challenges in blockchain scalability. Applied Sciences (Switzerland), 11(20). https://doi.org/10.3390/app11209372

15. Kinyua, J., & Awuah, L. (2021). Ai/ml in security orchestration, automation and response: Future research directions. Intelligent Automation and Soft Computing, 28(2). https://doi.org/10.32604/iasc.2021.016240

16. Kumari, P., & Kaur, P. (2021). A survey of fault tolerance in cloud computing. Journal of King Saud University - Computer and Information Sciences, 33(10). https://doi.org/10.1016/j.jksuci.2018.09.021

17. Lawrence, L. E. M., Echeverria, V., Yang, K., Aleven, V., & Rummel, N. (2024). How teachers conceptualise shared control with an AI co-orchestration tool: A multiyear teacher-centred design process. British Journal of Educational Technology, 55(3). https://doi.org/10.1111/bjet.13372

18. Lee, J. (2013). A view of cloud computing. International Journal of Networked and Distributed Computing, 1(1). https://doi.org/10.2991/ijndc.2013.1.1.2

19. Li, L., Kong, L., Li, Q., Yan, Z., & Li, H. (2014). Multi-tenant data authentication model for SaaS. Open Cybernetics and Systemics Journal, 8(1). https://doi.org/10.2174/1874110X01408010322

20. Li, S., Liu, L., & Peng, C. (2020). A review of performance-oriented architectural design and optimization in the context of sustainability: Dividends and challenges. Sustainability (Switzerland), 12(4). https://doi.org/10.3390/su12041427

21. Lokawati, H., & Widyani, Y. (2019). Monitoring System of Multi-Tenant Software as a Service (SaaS). Proceedings of 2019 International Conference on Data and Software Engineering, ICoDSE 2019. https://doi.org/10.1109/ICoDSE48700.2019.9092741

22. Longo, L., Wickens, C. D., Hancock, P. A., & Hancock, G. M. (2022). Human Mental Workload: A Survey and a Novel Inclusive Definition. Frontiers in Psychology, 13. https://doi.org/10.3389/fpsyg.2022.883321

23. Makki, M., van Landuyt, D., Lagaisse, B., & Joosen, W. (2018). A comparative study of workflow customization strategies: Quality implications for multi-tenant SaaS. Journal of Systems and Software, 144. https://doi.org/10.1016/j.jss.2018.07.014

24. Makki, M., van Landuyt, D., Lagaisse, B., & Joosen, W. (2021). Thread-level resource consumption control of tenant custom code in a shared JVM for multi-tenant SaaS. Future Generation Computer Systems, 115. https://doi.org/10.1016/j.future.2020.09.025

25. Marston, S., Li, Z., Bandyopadhyay, S., Zhang, J., & Ghalsasi, A. (2011). Cloud computing - The business perspective. Decision Support Systems, 51(1). https://doi.org/10.1016/j.dss.2010.12.006

26. Nordli, E. T., Haugeland, S. G., Nguyen, P. H., Song, H., & Chauvel, F. (2023). Migrating monoliths to cloud-native microservices for customizable SaaS. Information and Software Technology, 160. https://doi.org/10.1016/j.infsof.2023.107230

27. Ongowarsito, H., Prabowo, H., Meyliana, & Gaol, F. L. (2022). Adoption Readiness Assessment Model based on SaaS Maturity Level in SMEs. International Journal of Emerging Technology and Advanced Engineering, 12(4). https://doi.org/10.46338/ijetae0422_04

28. Pan, Y., Zhu, M., Lv, Y., Yang, Y., Liang, Y., Yin, R., Yang, Y., Jia, X., Wang, X., Zeng, F., Huang, S., Hou, D., Xu, L., Yin, R., & Yuan, X. (2023). Building energy simulation and its application for building performance optimization: A review of methods, tools, and case studies. Advances in Applied Energy, 10. https://doi.org/10.1016/j.adapen.2023.100135

29. Rafique, A., van Landuyt, D., & Joosen, W. (2018). PERSIST: Policy-Based Data Management Middleware for Multi-Tenant SaaS Leveraging Federated Cloud Storage. Journal of Grid Computing, 16(2). https://doi.org/10.1007/s10723-018-9434-6

30. Rahmani Asl, M., Zarrinmehr, S., Bergin, M., & Yan, W. (2015). BPOpt: A framework for BIM-based performance optimization. Energy and Buildings, 108. https://doi.org/10.1016/j.enbuild.2015.09.011

31. Rehrmann, R., Binnig, C., Böhm, A., Kim, K., & Lehner, W. (2020). Sharing opportunities for OLTP workloads in different isolation levels. Proceedings of the VLDB Endowment, 13(10). https://doi.org/10.14778/3401960.3401967

32. Richer, G., Pister, A., Abdelaal, M., Fekete, J. D., Sedlmair, M., & Weiskopf, D. (2024). Scalability in Visualization. IEEE Transactions on Visualization and Computer Graphics, 30(7). https://doi.org/10.1109/TVCG.2022.3231230

33. Schouten, A. M., Flipse, S. M., van Nieuwenhuizen, K. E., Jansen, F. W., van der Eijk, A. C., & van den Dobbelsteen, J. J. (2023). Operating Room Performance Optimization Metrics: a Systematic Review. Journal of Medical Systems, 47(1). https://doi.org/10.1007/s10916-023-01912-9

34. Sharma, A., & Kaur, P. (2023). Tamper-proof multitenant data storage using blockchain. Peer-to-Peer Networking and Applications, 16(1). https://doi.org/10.1007/s12083-022-01410-8

35. Shilpashree, S., Patil, R. R., & Parvathi, C. (2018). Cloud computing an overview. International Journal of Engineering and Technology (UAE), 7(4). https://doi.org/10.14419/ijet.v7i4.10904

36. Swathi, P., & Venkatesan, M. (2021). Scalability improvement and analysis of permissioned-blockchain. ICT Express, 7(3). https://doi.org/10.1016/j.icte.2021.08.015

37. Taleb, N., & Mohamed, E. A. (2020). Cloud computing trends: A literature review. Academic Journal of Interdisciplinary Studies, 9(1). https://doi.org/10.36941/ajis-2020-0008

38. Torres-Narbona, M., Guinea, J., Martinez-Alarcon, J., Munoz, P., Pelaez, T., & Bouza, E. (2008). Workload and clinical significance of the isolation of zygomycetes in a tertiary general hospital. Medical Mycology, 46(3). https://doi.org/10.1080/13693780701796973

39. Ucbas, Y., Eleyan, A., Hammoudeh, M., & Alohaly, M. (2023). Performance and Scalability Analysis of Ethereum and Hyperledger Fabric. IEEE Access, 11. https://doi.org/10.1109/ACCESS.2023.3291618

40. Wu, Y. (2021). Cloud-Edge Orchestration for the Internet of Things: Architecture and AI-Powered Data Processing. IEEE Internet of Things Journal, 8(16). https://doi.org/10.1109/JIOT.2020.3014845

41. Yang, C., Ye, W., & Li, Q. (2022). Review of the performance optimization of parallel manipulators. Mechanism and Machine Theory, 170. https://doi.org/10.1016/j.mechmachtheory.2022.104725

42. Yassin, M., Talhi, C., & Boucheneb, H. (2019). ITADP: An inter-tenant attack detection and prevention framework for multi-tenant SaaS. Journal of Information Security and Applications, 49. https://doi.org/10.1016/j.jisa.2019.102395

43. Zamboni, K., Schellenberg, J., Hanson, C., Betran, A. P., & Dumont, A. (2019). Assessing scalability of an intervention: Why, how and who? Health Policy and Planning, 34(7). https://doi.org/10.1093/heapol/czz068

44. Zhang, D., Pee, L. G., Pan, S. L., & Liu, W. (2022). Orchestrating artificial intelligence for urban sustainability. Government Information Quarterly, 39(4). https://doi.org/10.1016/j.giq.2022.101720

45. Zhang, Q., Cheng, L., & Boutaba, R. (2010). Cloud computing: State-of-the-art and research challenges. Journal of Internet Services and Applications, 1(1). https://doi.org/10.1007/s13174-010-0007-6