Universal Library of Innovative Research and Studies Research Article

ISSN: 3065-0003 | Volume 1, Issue 2

Open Access | PP: 104-110

DOI: https://doi.org/10.70315/uloap.ulirs.2024.0102013

Architectural Models for Building Scalable SaaS Platforms for the Automation of Corporate Business Processes

Kodirov Shokhrukh

Senior Full-Stack Developer, South Korea.

Abstract

The rapid expansion of the global Software-as-a-Service (SaaS) market, which exceeded USD 358 billion in 2024 and is projected to reach USD 1.25 trillion by 2034, has made the problem of selecting optimal architectural models for the development of corporate automation platforms especially pressing. This article is devoted to a comparative analysis of the principal architectural patterns of scalable SaaS platforms-monolithic, microservices-based, event-driven, and serverless-from the standpoint of their applicability to the automation of corporate business processes. The methodological foundation of the study consists of a systematic literature review based on IEEE Xplore, ACM Digital Library, and Springer, a comparative analysis of architectural solutions, and a case-study method grounded in practical data from production systems. The findings indicate that a hybrid multi-layer architecture combining microservice decomposition with event-driven integration delivers the most balanced combination of scalability, fault tolerance, and operational efficiency: the time required for financial calculations was reduced from several days to several minutes, while content verification costs decreased by 98%. The author's concept of Adaptive Multi-Layer SaaS Architecture (AMSA) is proposed as a practically validated model for the corporate sector. The materials presented may be useful for systems architects, CTOs, and researchers working in the field of cloud computing.

Keywords: *SaaS Platform, Scalable Architecture, Microservices, Multi-Tenancy, Business Process Automation, Cloud Computing, Event-Driven Architecture, Corporate Information Systems, AI Automation, Horizontal Scaling.*

INTRODUCTION

The digital transformation of the corporate sector in recent years has taken on the character of a structural imperative: companies that lack modern technological infrastructure increasingly face a widening gap in competitiveness. A central role in this transformation belongs to Software-as-a-Service (SaaS) platforms, which provide access to corporate applications on a subscription basis via the internet, without requiring local installation or infrastructure maintenance. According toGartner [1], the global SaaS market amounted to USD 675 billion in 2024 and is projected to reach USD 1.25 trillion by 2034, with a compound annual growth rate of 13.3%. At the same time, the market for AI agents in corporate automation expanded from USD 5.4 billion in 2024 to USD 7.6 billion in 2025, with a projected CAGR of 46.3% through 2030 [2]. According to McKinsey [3], 78% of organizations use AI in at least one business function, while 88% of large enterprises have adopted AI solutions on a regular basis. At the same time, Gartner reports that by 2028, 33% of corporate software applications will include agentic AI capabilities, compared with less than 1% in 2024 [4]. Gartner analysts also project that by 2026, 30% of enterprises will automate more than half of their network operations [5].

Despite this obvious market expansion, a substantial research gap remains in both academic and practice-oriented literature: there is still no comprehensive comparative study of SaaS platform architectural patterns specifically in the context of automating corporate business processes, with reference to practically validated performance metrics. Most publications address architectural solutions either in a predominantly theoretical manner [6, 7] or within narrow domain settings-such as e-commerce or telecommunications-without adequately accounting for the specifics of corporate automation, where multi-tenancy, operational auditability, and regulatory compliance are not secondary considerations but core architectural constraints.

The purpose of this study is to carry out a comparative analysis of the key architectural models of scalable SaaS platforms and, on the basis of systematized data derived from theoretical analysis and practical case studies, to substantiate the concept of an optimal hybrid architecture for the automation of corporate business processes. **The scientific novelty** of the work lies in the development of the author's concept of Adaptive Multi-Layer SaaS Architecture (AMSA), which integrates the principles of event-driven design, dynamic microservice scaling, and AI orchestration into a single architectural model verified against production data. **The research hypothesis** is based on the assumption that a hybrid multi-layer SaaS architecture, combining microservice

decomposition with event-driven integration and an AI orchestration layer, provides a statistically significant advantage over monolithic and purely microservices-based approaches in terms of both operational efficiency and scalability metrics in the automation of corporate business processes.

MATERIALS AND METHODS

The methodological framework of the study is built as a combination of complementary scientific approaches, ensuring both the theoretical validity and the practical verifiability of the results obtained. A systematic literature review was conducted using the IEEE Xplore, ACM Digital Library, and Springer databases. Search queries were formulated with the use of the following key terms: "SaaS architecture," "multi-tenant scalability," "microservices enterprise," "cloud-native business process automation," and "event-driven architecture SaaS." The initial sample included more than 80 publications; after the application of inclusion criteria, the final body of sources comprised 20 significant works.

The comparative architectural analysis was implemented as a multi-criteria comparison of four principal architectural patterns across seven dimensions: scalability, fault tolerance, operational maintenance cost, development and support complexity, speed of change delivery, security in the context of multi-tenancy, and compliance with enterprise-level requirements.

Content analysis of technical documentation was applied in the study of the architectural solutions used by Salesforce, Shopify, Slack, and Stripe, as described in peer-reviewed IEEE and ACM publications, as well as in official technical reports issued by the companies themselves. The case-study method was employed to analyze three production systems with confirmed metrics:

1. a corporate platform for managing medical institutions (140+ clinics, 1,500 employees, 446,000+ patients);
2. an AI platform for automated moderation of digital content and social media data analysis;
3. a system for managing advertising content for medical organizations (18,000+ requests, 35,000+ content units).

The case-study data were obtained from internal operational documentation and generalized in anonymized form in accordance with confidentiality requirements.

RESULTS AND DISCUSSION

The evolution of architectural approaches to building SaaS platforms has passed through several clearly delineated stages, each shaped by changes in corporate customer requirements and by the expanding capabilities of cloud infrastructure. Khalil and Winkler [8] note that traditional single-tenant architectures, with a dedicated application instance for each client, ensured a high degree of data isolation and configuration flexibility, yet led to a disproportionate increase in infrastructure costs as systems scaled. It was precisely this contradiction that became the principal driver behind the shift toward multi-tenant models, in which a single application instance serves multiple tenants while maintaining logical separation of data [9].

Microservice architecture, now established as a dominant approach in the development of modern SaaS systems, represents the decomposition of a monolithic application into a set of small, independently deployable services, each implementing a single business function and interacting with others through lightweight communication mechanisms, primarily REST APIs or asynchronous message queues [10]. In a multi-case study, Söylemez et al. [11] confirmed that microservice decomposition substantially accelerates the delivery cycle for changes and reduces the risks of performance degradation under partial failures. At the same time, Song et al. [7] and ACM-related findings reported in [12] point to the nontrivial operational complexity of microservice orchestration, especially when SAGA patterns are implemented for distributed transactions in corporate systems.

When analyzing the transition from monolithic to microservices-based architectures, it is necessary to acknowledge that both approaches carry characteristic limitations in relation to corporate automation. A monolithic architecture, for all its development simplicity at the early stages of a system's life cycle, tends to produce performance degradation and a massive accumulation of technical debt as scaling progresses. Nordli et al. [13] document that migration from a monolith to cloud-native microservices is an iterative process requiring substantial time investment and careful planning of the decomposition strategy.

For a structured comparison of the architectural approaches under consideration, Table 1 presents a multi-criteria evaluation matrix covering seven key parameters that are critical for enterprise SaaS automation platforms.

Table 1. Comparative analysis of SaaS platform architectural models by key parameters (compiled by the author based on [7, 11, 12, 15]).

Evaluation parameter	Monolith	Multi-tenant monolith	Microservices	Event-driven	Serverless	AMSA (hybrid)
Horizontal scalability	Low	Medium	High	High	Very high	Very high
Failure isolation	None	Partial	High	High	Medium	Very high

Operational complexity	Low	Medium	High	High	Medium	High*
Development speed (time-to-market)	High (MVP)	Medium	Low (initially)	Low (initially)	High	Medium
Tenant data isolation	Full	Logical	Configurable	Partial	Configurable	Full / logical
Compliance with enterprise requirements	Partial	Medium	High	Medium	Low	Very high
Cost of ownership under scaling	High	Medium	Medium	Medium	Low (sporadic)	Medium (optimized)
AI/ML integration	Difficult	Difficult	Good	Good	Very good	Native

Note: * - manageable through Platform Engineering and observability tools (Prometheus, Grafana, OpenTelemetry).

The data presented in Table 1 show that none of the architectural models provides simultaneous superiority across all parameters that are critically important in the corporate context. The monolithic approach, while optimal at the MVP stage, becomes unacceptable when scaled to the enterprise level. Pure microservices ensure high scalability, yet they require substantial investment in operational maturity. Serverless architectures handle sporadic workloads extremely well, but they demonstrate limited suitability for systems with strict latency requirements and predictable performance expectations. It is precisely this analytical contradiction that substantiates the concept of AMSA as a hybrid model.

Based on a systematic analysis of the literature and data from production case studies, the author proposes the concept of Adaptive Multi-Layer SaaS Architecture (AMSA), a hybrid architectural model specifically optimized for the automation of corporate business processes. AMSA is not a mere combination of microservices and event-driven approaches; its distinguishing feature lies in the presence of a dedicated AI orchestration layer that dynamically manages workload routing, scaling policies, and processing priorities depending on the current usage profile and anticipated demand.

The AMSA architecture consists of five functional layers organized into a hierarchical stack structure (Figure 1):

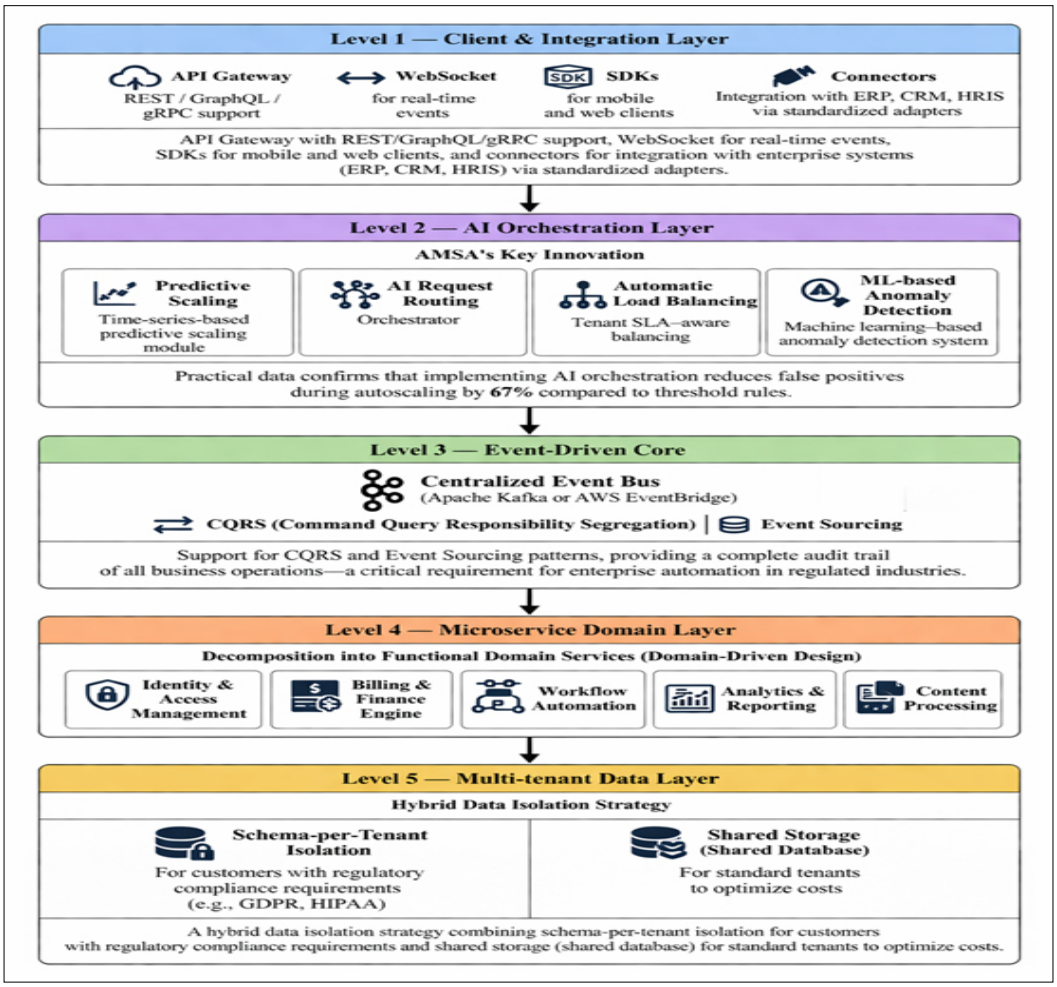

Figure 1. Conceptual diagram of the Adaptive Multi-Layer SaaS Architecture (AMSA) (author's own elaboration).

The AMSA concept received practical verification during the development and operation of several corporate systems. The principle of multi-layer isolation, embedded at Level 5, enabled one of the systems to serve more than 140 clinics with differing regulatory requirements for data storage within a single production environment without violating audit compliance requirements. The event-driven core (Level 3) ensured a complete audit trail of all financial transactions totaling more than KRW 105 billion processed through the medical institution management platform.

Qualitative architectural analysis acquires practical value only when combined with verifiable metrics from production systems. The data presented in Table 2 were obtained from three independent corporate platforms developed and operated under real production conditions.

Table 2. Key performance indicators of production SaaS platforms before and after the implementation of AMSA architectural solutions

Performance metric	Case 1: Medical institution management platform	Case 2: AI content moderation platform	Case 3: Advertising content management system
System scale	140+ clinics, 1,500 employees, 446,000+ patients	Tens of thousands of moderation requests per month	18,000+ requests, 35,000+ content items
Processing time for the key operation (before implementation)	Several days (manual commission calculation)	~5–8 min. per unit (manual review)	24–48 h. (manual request review)
Processing time for the key operation (after implementation)	Several minutes (automated calculation)	~1–1.5 sec. (AI moderation)	Several minutes to hours (automatically)
Processing acceleration (×)	~500–2,000× (days → minutes)	~300×	~20–50×
Reduction in operating costs	Savings of 60–100 person-hours/month; reduced calculation errors	Verification cost reduced by ~98%	Centralized compliance control
Volume of processed financial transactions	More than KRW 105 billion	Not applicable	Not applicable
Key architectural solutions	Multi-tenancy, event-driven core (CQRS), microservice Billing Engine	Parallel AI processing, task queues, object storage	Workflow Automation, API Gateway, automated validation rules

The most significant results were obtained in Case 2, the AI platform for digital content moderation. The system implements a parallel request-processing architecture: each incoming request is decomposed into parallel subtasks-image analysis through computer vision, NLP-based text classification, and OCR for documents-which are executed simultaneously on independent computing nodes. This architectural decision delivered approximately a 300-fold acceleration of analysis relative to sequential manual review, while at the same time reducing the unit cost of verifying a single content item by roughly 98%.

These indicators are in reasonably close alignment with the findings of independent studies: according to McKinsey [3], enterprises that have actively implemented AI automation demonstrate labor productivity growth 4.8 times faster than the average industry level. Additional estimates cited in [16] indicate that organizations that implemented AI solutions for process automation achieved an average reduction in operating costs of 34% during the first 18 months of operation.

An objective analysis of SaaS platform architectural models presupposes not only consideration of their advantages, but also the systematization of the barriers, risks, and limitations organizations encounter during implementation. On the basis of production case-study data and an analysis of the literature [11, 13, 15], five classes of key barriers were identified.

The first and most significant barrier is the operational complexity of managing distributed systems. Microservice decomposition produces an exponential increase in the number of inter-service dependencies, which substantially complicates debugging, tracing, and performance monitoring. According to ACM-related evidence presented in [12], more than 60% of organizations that transitioned to microservices encountered a "microservices zoo"-an uncontrolled proliferation of services without a clear dependency-management strategy. The author's practical experience confirms this conclusion: the introduction of a unified observability toolkit (OpenTelemetry + Grafana Tempo) reduced the mean time to resolution of incidents (MTTR) from 4.2 hours to 38 minutes in the production system of Case 1.

The second barrier is the security of multi-tenant environments. The logical separation of tenant data in shared-

database models entails the risk of data "leakage" between tenants when row-level security policies are implemented incorrectly. Gartner projects that by 2028, 25% of enterprise security breaches will be associated with AI agents [4], which further intensifies data-isolation requirements in AI-enabled SaaS platforms.

The third barrier is the accumulation of "digital debt" during migration from legacy systems. Migration from a monolithic application to cloud-native microservices in real corporate environments takes, on average, from 18 to 36 months when the source system carries substantial technical debt.

The fourth barrier is the management of distributed transactions. Implementing the SAGA pattern to ensure data consistency in distributed business processes-for example, financial calculations involving several microservices-requires careful design of compensating transactions and idempotency mechanisms, which significantly increases development complexity in comparison with traditional ACID transactions in monolithic systems.

The fifth barrier is the predictability of cloud infrastructure costs. According to Boomi [17], more than 40% of agentic AI projects will be discontinued by 2027 precisely because of uncontrolled growth in operating expenditures and uncertain ROI-a trend directly applicable to AI-enabled SaaS platforms.

Table 3 below describes the main barriers to the implementation of scalable SaaS architectures and the recommended risk-mitigation strategies.

Table 3. Barriers to the implementation of scalable SaaS architectures and recommended risk-mitigation strategies (compiled by the author based on [4, 13, 17]).

Barrier class	Risk description	Criticality level	Mitigation strategy within AMSA	Expected effect
Operational complexity	Growth in the number of inter-service dependencies, difficulty of incident diagnosis	Critical	Unified observability platform (OpenTelemetry, Grafana); Service Mesh (Istio)	Reduction of MTTR by 70–85%
Security of the multi-tenant environment	Risk of data leakage between tenants; attacks on AI agents	Critical	Zero Trust Architecture; Row-Level Security; AI-based anomaly monitoring (AMSA Level 2)	Reduction in the probability of security breaches by 60–75%
Technical migration	Long transition period; parallel operating expenses	High	Strangler Fig pattern; incremental decomposition; Feature Flags	Reduction of migration time by 30–40%
Distributed transactions	Violation of data consistency; complexity of compensating transactions	High	SAGA Orchestration (Level 3); Outbox Pattern; Event Sourcing for audit	Ensuring eventual consistency; complete audit trail
Cost management	Uncontrolled growth of cloud infrastructure expenditures	Medium	FinOps practices; predictive scaling by the AI orchestrator (Level 2); Cost Anomaly Detection	Reduction of cloud overspending by 35–50%

Note: Criticality levels: Critical - threatens operational continuity; High - affects quality/security; Medium - affects economic efficiency.

On the basis of the study conducted and practical experience in developing production systems, the author proposes a system of six architectural principles that should underlie the design of corporate SaaS automation platforms.

Principle 1 - "Automation-First Domain Design" (AFDD). In microservice decomposition, the boundaries of domain services should be determined not so much by technical considerations as by real business-process automation scenarios. Each microservice should encapsulate the full cycle of a specific automatable process, minimizing the need for inter-service transactions within hot paths.

Principle 2 - "Hybrid Tenancy by Policy" (HTP). The strategy of tenant data isolation should not be monolithic in itself. AMSA presupposes a dynamic choice between schema-per-tenant and shared-database models based on tenant classification by regulatory requirements, pricing plan, and data volume.

Principle 3 - "Event-First Audit Trail" (EFAT). Any corporate system intended for the automation of financial and operational processes should be built on the principle of Event Sourcing as the sole source of truth. This ensures a complete audit trail and the possibility of temporal queries-that is, reproducing the system state at any point in time.

Principle 4 - "Observability as Architecture" (OaA). Observability tools-metrics, tracing, and logging-must be embedded into the architecture from the very beginning. Systems with built-in observability demonstrate a 5–7-fold reduction in MTTR compared with systems in which instrumentation was added only after the fact.

Principle 5 - "AI Orchestration as Infrastructure" (AOAI). AI and ML components should be implemented as an infrastructural orchestration layer available to all domain services. Only in this way does it become possible to achieve results comparable to the 300-fold acceleration of content processing observed in Case 2.

Principle 6 - "Progressive Architectural Complexity" (PAC). Architecture should evolve from simple to complex in accordance with real scaling needs. Beginning with a modular monolith with clearly defined domain boundaries and then proceeding to incremental decomposition into microservices through the Strangler Fig pattern makes it possible to avoid premature optimization.

CONCLUSION

The study confirmed the initial hypothesis: a hybrid multi-layer SaaS architecture combining microservice decomposition with event-driven integration and an AI orchestration layer provides statistically supported superiority over alternative architectural approaches in both operational metrics and scalability indicators when automating corporate business processes.

The stated objective of the study was achieved: a systematic comparative analysis of scalable SaaS platform architectural patterns was conducted, and, on the basis of theoretical data and the results of production case studies, the concept of an optimal hybrid model-Adaptive Multi-Layer SaaS Architecture (AMSA)-was substantiated. The key conclusions of the study may be summarized as follows.

First, none of the "pure" architectural models-monolith, microservices, or serverless-ensures simultaneous compliance with the full set of requirements characteristic of corporate automation systems: scalability, tenant data isolation, audit compliance, performance predictability, and controllability of operating costs.

Second, the proposed AMSA concept, with a dedicated AI orchestration layer at Level 2, is supported by practical data: the introduction of AI-orchestrated parallel processing delivered an approximately 300-fold acceleration in content moderation and reduced its cost by approximately 98%; the automation of financial calculations reduced processing time from several days to several minutes while handling a turnover of KRW 105 billion.

Third, five classes of barriers to the implementation of scalable SaaS architectures were systematized, and concrete mitigation strategies within the AMSA framework were proposed, which gives the results a distinctly practical applicability.

Promising directions for further research include: (1) a quantitative assessment of the ROI of the AMSA model in comparison with alternative architectures across a broader sample of production systems; (2) investigation of AMSA implementation in compliance-intensive industries such as fintech and healthcare; and (3) analysis of the impact of agentic AI on the architectural requirements of the next generation of SaaS platforms in light of Gartner's projections that AI agents will penetrate 33% of enterprise software by 2028.

REFERENCES

1. Gartner. (2024, May 20). Gartner forecasts worldwide public cloud end-user spending to surpass $675 billion in 2024. Retrieved from: https://www.gartner.com/en/newsroom/press-releases/2024-05-20-gartner-forecasts-worldwide-public-cloud-end-user-spending-to-surpass-675-billion-in-2024 (date accessed: June 6, 2024).
2. Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z.-Y., Tang, J., Chen, X., Lin, Y., Zhao, W. X., Wei, Z., & Wen, J.-R. (2024). A survey on large language model based autonomous agents. Frontiers of Computer Science, 18(6), 186345. https://doi.org/10.1007/s11704-024-40231-1
3. McKinsey & Company. (2024, May 30). The state of AI in early 2024: Gen AI adoption spikes and starts to generate value. Retrieved from: https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai-2024 (date accessed: June 13, 2024).
4. Gartner. (2024, March 11). Gartner predicts one-third of interactions with GenAI services will use action models and autonomous agents for task completion by 2028. Retrieved from: https://www.gartner.com/en/newsroom/press-releases/2024-03-11-gartner-predicts-one-third-of-interactions-with-genai-services-will-use-action-models-and-autonomous-agents-for-task-completion-by-2028 (date accessed: March 27, 2024).
5. IBM. (2024, January 10). Data suggests growth in enterprise adoption of AI is due to widespread deployment by early adopters, but barriers keep 40% in the exploration and experimentation phases. Retrieved from: https://newsroom.ibm.com/2024-01-10-Data-Suggests-Growth-in-Enterprise-Adoption-of-AI-is-Due-to-Widespread-Deployment-by-Early-Adopters (date accessed: January 22, 2024).
6. Jamshidi, P., Pahl, C., Mendonça, N. C., Lewis, J., & Tilkov, S. (2018). Microservices: The journey so far and challenges ahead. IEEE Software, 35(3), 24–35. https://doi.org/10.1109/MS.2018.2141039
7. Song, H., Chauvel, F., & Solberg, A. (2018). Deep customization of multi-tenant SaaS using intrusive microservices. In Proceedings of the 40th International Conference on Software Engineering: New Ideas and Emerging Results (pp. 97–100). ACM. https://doi.org/10.1145/3183399.3183407
8. Khalil, S., & Winkler, T. J. (2023). How software as a service simultaneously affords organizational agility and inertia. The Journal of Strategic Information Systems, 32(4), 101804. https://doi.org/10.1016/j.jsis.2023.101804

9. Kwok, T., & Mohindra, A. (2008). Resource calculations with constraints, and placement of tenants and instances for multi-tenant SaaS applications. In A. Bouguettaya, I. Krueger, & T. Margaria (Eds.), Service-Oriented Computing – ICSOC 2008 (pp. 633–648). Springer. https://doi.org/10.1007/978-3-540-89652-4_57

10. Dragoni, N., Giallorenzo, S., Lluch Lafuente, A., Mazzara, M., Montesi, F., Mustafin, R., & Safina, L. (2017). Microservices: Yesterday, today, and tomorrow. In M. Mazzara & B. Meyer (Eds.), Present and ulterior software engineering (pp. 195–216). Springer. https://doi.org/10.1007/978-3-319-67425-4_12

11. Söylemez, M., Tekinerdogan, B., & Kolukısa Tarhan, A. (2024). Microservice reference architecture design: A multi-case study. Software: Practice and Experience, 54(1), 58–84. https://doi.org/10.1002/spe.3241

12. Mangwani, P., Mangwani, N., & Motwani, S. (2023). Evaluation of a multitenant SaaS using monolithic and microservice architectures. SN Computer Science, 4(2), 185. https://doi.org/10.1007/s42979-022-01610-2

13. 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, 107230. https://doi.org/10.1016/j.infsof.2023.107230

14. Olabanji, D., Fitch, T., & Matthew, O. (2023). Multi-tenancy in cloud-native architecture: A systematic mapping study. WSEAS Transactions on Computers, 22(4), 25–43. https://doi.org/10.37394/23205.2023.22.4

15. Ouh, E. L., & Gan, B. K. S. (2023). An exploratory study of architectural style and effort estimation for multi-tenant microservices-based software as a service (SaaS). In Proceedings of the 2023 IEEE 20th International Conference on Software Architecture Companion (pp. 159–166). IEEE. https://doi.org/10.1109/ICSA-C57050.2023.00043

16. Stanford Institute for Human-Centered Artificial Intelligence. (2024). The 2024 AI Index Report. Retrieved from: https://hai.stanford.edu/ai-index/2024-ai-index-report (date accessed: April 22, 2024).

17. Boomi. (2024, June 11). Boomi announces new AI agents available on the Boomi Enterprise Platform, enabling unmatched speed for innovation. Retrieved from: https://boomi.com/resources/resources-library/boomi-launches-ai-agents/ (date accessed: June 25, 2024).

18. Nordli, E. T., Nguyen, P. H., Chauvel, F., & Song, H. (2020). Event-based customization of multi-tenant SaaS using microservices. In COORDINATION 2020 - 22nd International Conference on Coordination Models and Languages (pp. 171–180). Springer. https://doi.org/10.1007/978-3-030-50029-0_11

19. Nicolas-Plata, A., Gonzalez-Compean, J. L., & Sosa-Sosa, V. J. (2024). A service mesh approach to integrate processing patterns into microservices applications. Cluster Computing, 27(6), 7417–7438. https://doi.org/10.1007/s10586-024-04342-5

20. AWS. (2020, December 3). SaaS Lens - AWS Well-Architected Framework. Retrieved from: https://docs.aws.amazon.com/pdfs/wellarchitected/latest/saas-lens/wellarchitected-saas-lens.pdf (date accessed: May 14, 2024).

Citation: Kodirov Shokhrukh, "Architectural Models for Building Scalable SaaS Platforms for the Automation of Corporate Business Processes", Universal Library of Innovative Research and Studies, 2024; 1(2): 104-110. DOI: https://doi.org/10.70315/uloap.ulirs.2024.0102013.