International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(23s) 2026 pp. 925-932
https://doi.org/10.70917/ijcisim-2026-5654
Article

Multi-Agent AI Architecture for Autonomous Retail Operations: Coordinating Intelligent Agents Across Cloud, DevOps, and Enterprise Platform

Venkateswara Rao Movva[1], Dr. Piyush Kumar Pareek[2]

[1] Software Engineer III. Email: movvavenkateswaraa@gmail.com ORCID: 0009-0007-6905-1999
[2] MTech, PhD, PostDoc, Patent Agent (Govt. of India), Prof. AIML, Nitte Meenakshi Institute of Technology, Nitte Deemed to be University, Bengaluru, Karnataka, India. Email: piyushkumarpareek88@gmail.com

Abstract: A multi-agent architecture supporting autonomous retail operations must accommodate dedicated functions for cloud, DevOps, and enterprise platforms. The deployment principle for these Cloud-native, DevOps-enabled, Enterprise Integrated Systems follows a federated approach to enable scalability, fault tolerance, and data retention while flexibly supporting latency-sensitive functions. Within each domain, intelligent agents, orchestrators, negotiators, and enterprise services interact to drive seamless operations. Coordinating agents at different levels and across domains address monitoring, lifecycle management, deployment, adaptation, incident response, and other cross-layer functions. Clear data governance policies covering provenance, quality, control, privacy, and retention capture the full data lifecycle, enabling compliance with industry- and scenario-specific regimes. Multi-agent systems support Cloud-native, DevOps-enabled, Enterprise Integrated Systems with dedicated functions for cloud, DevOps, and enterprise platforms. The federated deployment principle accommodates scalability, fault tolerance, and data retention while flexibly supporting latency-sensitive functions. Intelligent agents, orchestrators, negotiators, and enterprise services drive seamless operations within each domain. Coordination across layers and domains tackles monitoring, lifecycle management, deployment, adaptation, incident response, and other cross-layer functions. Data governance policies covering provenance, quality, control, privacy, and retention capture the full data lifecycle, enabling compliance with industry- and scenario-specific regimes.
Keywords: Autonomous retail operation, multi-agent systems, Cloud, Edge, DevOps, enterprise, supervision, coordination, interaction, communication, data governance, data policy, data privacy, artificial intelligence, intelligent agents, data provenance, security, reliability, deployment active learning,Multi-Agent AI, Autonomous Retail Operations, Intelligent Agent Coordination, Agentic AI Architecture, Cloud Computing, DevOps Automation, Enterprise Platforms, Multi-Agent Systems, Workflow Orchestration, Retail Process Automation

1. Introduction

The escalation of conflicts, pandemics, and economic crises has highlighted weaknesses in interdependence-based economic models of globalization [1]. Paradoxically, the return towards protectionist and isolationist policies increases risk due to reliance on mono-scaling of supplies and logistics [2]. These socio-economic trends are driving the transformation of the production-distribution-consumption-integrated processes. Investments in the invisible economy (algorithms, software, platforms) are increasingly concentrated in virtual infrastructures catering for the visible economy (offices, shops, warehouses, etc.) to delimit and control the costs of failure under increasingly unpredictable conditions [3]. The technology space is defined by Digital Twins, Digital Threads for Blockchain, Containers for Cloud natively deployment on Edge and Cloud, Digital Enablement platforms, etc [4]. These Invisible architecture is integrated at several interconnected platforms based on the principles of the Internet of Behaviours.

The orchestration of these Invisible architecture has been based de facto in centralised Orchestrator (Docker, Kubernetes, etc.) that need an evidently supervised AI in a Distributed Intelligence domain [5]. Proposed a Multi-Agent Architecture supporting the coordination of Intelligent software Agents over Data-Driven Supply Chains able to act according to the principles of Behaving Behaviours chain. With the definition of the Validation level of Data-Driven Supply Chains and the topology of the data communication infrastructure among the Agent ecosystem and the Orchestration of the Intelligent Agent residing in Invisible infrastructure providing Autonomous Retail operation support the role of Databases has become evident as a formalisation of Data ownership, Privacy Control and Consent Management [6].

Mathematical Formulas:

1.1 Agent–task allocation (Sections 4, 7)

Assignment of retail tasks to agents is a capacity-constrained utility maximization:

$$\max \ \Sigma_{i\in A} \Sigma_{j\in T} u_{ij} x_{ij} \qquad (1)$$

$$\text{s.t. } \Sigma_{i\in A} x_{ij} = 1 \ \forall j \ ; \ \ \Sigma_{j\in T} c_j x_{ij} \le C_i \ \forall i \ ; \ \ x_{ij} \in \{0,1\} \qquad (2)$$

1.2 End-to-end decision latency (Sections 3, 5)

The latency of one perceive–decide–act cycle decomposes across the four architectural layers, with the communication term summed over H inter-environment hops:

$$L_{e2e} = L_{perc} + L_{comm} + L_{dec} + L_{act} \qquad (3)$$

$$L_{comm} = \Sigma_{k=1}^{H} (d_k + m_k / b_k) \qquad (4)$$

1.3 Coordination overhead per paradigm (Section 3)

Messages exchanged per coordination round distinguish the three paradigms; the hybrid form partitions N agents into g clusters:

$$M_{cen}(N) = 2N \qquad (5)$$

$$M_{dis}(N) = N(N - 1) \qquad (6)$$

$$M_{hyb}(N, g) = N(N/g - 1) + g(g - 1) \qquad (7)$$

1.4 Availability of the federated deployment (Sections 3, 4)

Under the federated deployment principle, each environment is independently replicated and the system is available only if every environment is available:

$$A_{sys} = \Pi_{d\in D} [1 - (1 - a_d)^{rd}] \qquad (8)$$

1.5 Autonomy index (Sections 2, 8)

A single scalar summarizes how far an operation has moved toward autonomy, combining decision coverage, freedom from human intervention, and self-resolution of incidents:

$$\Omega = w_1\delta + w_2(1 - h) + w_3\rho \ , \quad w_1 + w_2 + w_3 = 1 \qquad (9)$$

1.6 Learning and adaptation (Section 6)

Agent-local adaptation follows a policy-gradient update on locally perceived experience; cloud-layer and cross-environment learning aggregate those updates in proportion to the data each environment contributes:

$$\theta^i_{t+1} = \theta^i_t + \eta \nabla_\theta J_i(\theta^i_t), \quad J_i = \mathbb{E}[\Sigma_k \gamma^k R_i] \qquad (10)$$

$$\theta^G_{t+1} = \Sigma_{d\in D} (n_d / n) \theta^d_{t+1} \qquad (11)$$

1.7 Governance gate (Section 5)

A refined pipeline or model is promoted only if the weighted governance score clears a threshold and no single dimension falls below its floor — the explicit quality-assurance process described in Section 6:

$$G = \Sigma_{k=1}^{5} \omega_k g_k \ , \quad \text{promote} \Leftrightarrow G \ge G_{min} \wedge \min_k g_k \ge g_{floor} \qquad (12)$$

1.8 Cross-environment scaling trigger and conflict resolution (Section 7)

The worked example of Section 7 — scaling a cloud service when DevOps deployment activity crosses a threshold — and the arbitration rule applied when agent instructions conflict:

$$n_{t+1} = \lceil \lambda_t \tau / (\kappa u^*) \rceil \ \text{ if } \ \lambda_t > \lambda_{thr} \qquad (13)$$

$$a^* = \arg\max_{a\in F(P)} \Sigma_i w_i u_i(a) \qquad (14)$$

2. Background and Motivation

Retail operations depend on a large number of stakeholders and service providers [7]. On one side, brands need to ensure stock availability, obtain new customers, retain existing customers, avoid stockouts, and reduce

markdown levels. On the other side, partners must take care of stock replenishment, support in-store executions, maintain compliance with regulations, and foster brand loyalty. Enterprises become busy trying to solve both management and execution problems simultaneously, thus losing focus [8]. In addition, most retail enterprises are not organized around a single service platform, and the setup of coordination among cloud solutions, DevOps platforms, and enterprise orchestration solutions is a challenge. Consequently, the proposed multi-agent-based architecture for autonomous retail operations is essential to support various environment levels and stacks.

Autonomous retail operations require cross-domain orchestration across three different parts: cloud platform, DevOps stack, and enterprise ecosystem. Cloud platforms offer high elasticity with a scale-to-zero capability, but intelligence and decision-making are limited. The DevOps stack provides smart resources orchestrated around constant decision making, but scaling up and down relies heavily on external service architecture. The enterprise ecosystem integrates all operations under an enterprise-oriented view, but resources are balanced across suppliers and brands without focusing on speed and intelligence. The lack of connection across those three domains limits the autonomy level of the entire operation. A multi-agent architecture focused on coordination across the three environments is needed. Prior work studying autonomous retail operation models included services that succeeded in making operations orbit those three domains, but focused only on DevOps or cloud architectures.

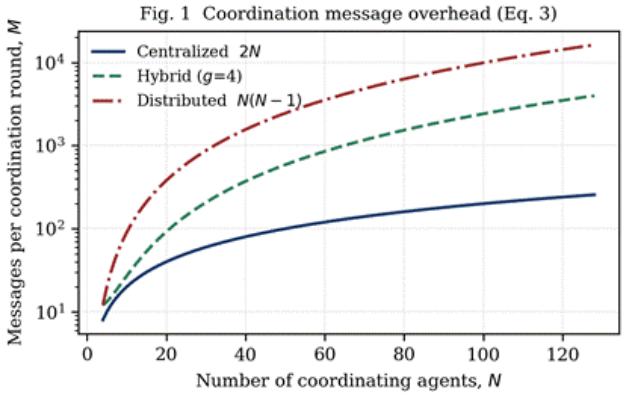

Fig. 1. Coordination message overhead per round against the number of agents, from Eqs. (5)–(7), with g = 4 clusters. Log scale on the ordinate.

3. Architectural Paradigms for Autonomous Retail

Three architectural paradigms—centralized, distributed, and hybrid—support multi-agent systems across multiple domains: cloud providers, DevOps platforms, and enterprise systems. Centralized architectures minimize latency but hinder scalability and fault tolerance. Distributed architectures scale with latency and fault tolerance costs, while hybrid architectures attempt to strike a compromise. The choice of architecture for an autonomous retail implementation hinges on the regulatory impact of latency and the operational impact of scalability and fault tolerance.

As an example, consider a standard enterprise system that conducts transactions with third parties. Such a system can be represented as a contracting process where the duration of the contracts is significantly greater than the time taken for an update to the enterprise system. The cloud side of a data update is therefore naturally centralized; a single agent makes the update and notifies other agents so they can amend own data copies and long-duration contracts [9]. At the same time, the enterprise system must be operating under regulatory scrutiny and the agents therefore require additional oversight to ensure proper functioning.

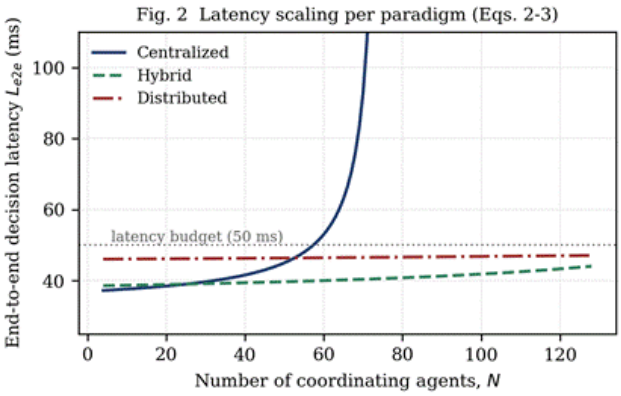

Fig. 2. End-to-end decision latency against agent count, from Eqs. (3)–(4) with L_{perc} = 8 ms, L_{dec} = 12 ms, L_{act} = 10 ms, and a queueing term at the coordination tier saturating at N = 75 (centralized), 220 (hybrid), 600 (distributed).

4. Reference Architecture

A modular reference architecture for multi-agent collaboration across cloud, DevOps, and enterprise environments supports orchestration of autonomous retail operations [10]. It comprises perception, decision, actuation, and governance layers. The communication interfaces to the offering portfolios of these environments are decoupled from the internal agent functions and their orchestration [11]. This opens these interaction channels for any third-party solutions possessing the respective interfaces, data schemas, and models mapped to the agent interactions[12]. The main data streams for coordinating across environments are also highlighted. Core components include the agent framework, orchestration engine, data fabric, policy engine, and monitoring [13]. Essential aspects of successful deployment—communication protocols, data schemas, and non-functional requirements such as latency, reliability, and security—are specified.

Distributed intelligent agents are key enablers for orchestrating operations across environments. By extending the interaction portfolio of such agents to incorporate the full data-and-solve cloud offering module, the deployment of intelligence agents to support cloud-based operations, conflict resolution, and combined DevOps-and-business policy orchestration becomes possible [14]. A defined agent architecture serves as a foundation for addressing operational engagements toward autonomously realizing business operations and reducing operational risks over time while monitoring sensor data to alert business entities and partners of emerging abnormal situations that require attention [15].

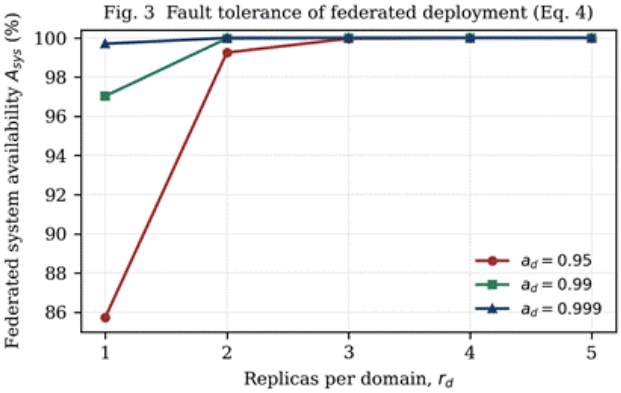

Fig. 3. Federated system availability against replicas per environment, from Eq. (8) with |D| = 3 and equal per-instance availability a_d.

5. Communication Protocols and Data Governance

The proposed architecture comprises multiple environments (cloud, DevOps, and enterprise platforms) that play different roles in relation to a retail operation. Communication across environments is therefore crucial [16]. Environmental borders are crossed by agents taking actions in one environment, which requires the execution of services in another. Furthermore, agents in disparate environments interact with one another to accomplish their goals [17]. Effective cross-environment communication requires appropriate data models for message content and

governance policies concerning access control, privacy preservation, and these mechanisms are applicable across environments.

A retail operation resides in heterogeneous environments. As such, the required communication protocols must address data transport between different environments [18]. When it comes to the content of those messages, unambiguous message semantics are needed to ensure that both sending and receiving agents are capable of interpreting message content correctly [19]. Latency requirements and data consistency guarantees—if any—must also be addressed. Such requirements are closely related to data governance: who has access to a given dataset? For what purpose can the data be used? Are there any rules (e.g. regulatory policies) that must be associated with the use of the data? Where do the data come from? Who owns a dataset? How can the data be temporally tracked to evaluate their quality? These operations are even more complex in a multi-environment setup, where data can change between different environments and multiple agents can apply transformations on the same dataset in a partly overlapping time window.

TABLE I. SYMBOLS USED IN THE FORMAL MODEL

Symbol	Meaning	Domain / units
A, T	Set of intelligent agents; set of retail tasks	finite sets
D	Set of environments {cloud, DevOps, enterprise}	$\|D\| = 3$
$\mathbf{u_{ij}}$, $\mathbf{c_j}$, $\mathbf{C_i}$	Utility of agent i on task j; task cost; agent capacity	$\mathbb{R} \geq 0$
N, g	Number of coordinating agents; number of coordination clusters	$\mathbb{N}$
$\mathbf{a_d}$, $\mathbf{r_d}$	Per-instance availability and replica count in environment d	[0,1]; $\mathbb{N}$
δ, h, ρ	Decision coverage; human-intervention rate; self-resolution ratio	[0,1]
θ, η, γ	Policy parameters; learning rate; discount factor	$\mathbb{R}$
$\mathbf{g_k}$, $\boldsymbol{\omega_k}$	Governance dimension score and its weight	[0,1]

6. Intelligence, Learning, and Adaptation

Intelligent agents must possess some degree of adaptive capabilities in order to fulfill their designated responsibilities more effectively over time [20]. Learning and adaptation in the multi-agent architecture can occur at three different levels: by individual agents (i.e., agent-specific learning), at the level of the cloud environment (i.e., learning in the Cloud layer), or for the other two environments (DevOps and enterprise platforms) in an organised manner (i.e., cross-environment learning) [21]. These three forms of learning manifest themselves in very different ways.

The agents are capable of modifying their own models or pipelines using the data they perceive online and experience gained in performing their inherent activities. Each agent can track a complex model or a set of pipelines [22]. Adaptation occurs on the agent side, guided by a self-bias through reinforcement learning or similar learning mechanisms [23]. Although the agent-specific learning processes are typically hidden from the outside world, their results in the form of refined pipelines are made available to other agents, permitting improved operations on the other objects in the Cloud layer [24]. The persistency of these refined pipelines and their applicability in time is normally governed by an explicit quality assurance process.

TABLE II. COMPARISON OF ARCHITECTURAL PARADIGMS FOR AUTONOMOUS RETAIL (SECTION 3)

Criterion	Centralized	Distributed	Hybrid
Coordination latency	Lowest at small N	Higher fixed floor	Near-minimal, stable
Scalability	Orchestrator saturates	Scales with N	Scales per cluster
Fault tolerance	Single point of failure	Degrades gracefully	Cluster-level isolation
Consistency	Strong, single writer	Eventual	Strong in-cluster
Governance auditability	Simple, one ledger	Complex, dispersed	Federated per domain
Best-fit workload	Contract & master data	Edge perception, store ops	Cross-domain retail ops

7. Coordination Strategies Across Environments

Agents operating across cloud, DevOps, and enterprise platforms, each with specific goals, must coordinate [25]. This can be achieved by defining high-level delegation strategies that distribute workloads across the platforms—such as automatically scaling a cloud service when deployment activity in a DevOps pipeline exceeds a designated threshold—or by orchestrating execution at runtime [26]. Communication between cross-platform agents tackles synchronisation of goals or actions and resolution of possible conflicting instructions.

Coordination may also focus on policy, governance, or compliance compliance [27]. Policies defined in a centralised enterprise platform can be pushed to development pipelines, for example, using agents in DevOps tools to ensure security scans of code and resources, update system inventories, or verify the application of governance rules such as licensing constraints on deployed resources. User-defined compliance slates in the enterprise can be followed by agents in the other environments, supporting operations with financing, certification, safety, or ethical considerations.

TABLE III. REFERENCE ARCHITECTURE LAYERS AND AGENT FUNCTIONS (SECTION 4)

Layer	Core component	Agent role	Representative output
Perception	Data fabric	Sensing / ingestion agent	Normalized event & telemetry stream
Decision	Agent framework	Planner, negotiator	Task allocation per Eqs. (1)-(2)
Actuation	Orchestration engine	Deployment / scaling agent	Scaling action per Eq. (13)
Governance	Policy engine, monitoring	Supervisor, auditor	Compliance verdict per Eq. (12)

8. Conclusion

Autonomous systems empower the replacement of humans, not only for high-risk tasks but increasingly also for routine and repetitive jobs. NL-PV disciplines augment retail agencies with reliable and redundant low-cost functionality, allowing deployment for operations that are still exceptionally difficult (virtual try-on) or expensive (personalization, security), Natural-Linguistic-Perception and Vision, and coordination among domains such as Cloud, DevOps and Enterprise Platforms, These technologies enable multilayered autonomous solutions such as moving shelves for stores without staff, intelligent personal shoppers for tourism experiences, cloud services for companies with a presence at the edge or on the environment itself, and bank branches that can also function as tourist information centers without operational costs, etc [28].

NLPV deployments can integrate these systems reducing their operating costs and risk of errors. The formal analysis techniques, capabilities of LTL spaces, agreement in automatically deployed products and the exploration of possible decision trees for possible different paths in evolution with decision points distributed among agents are differential elements that can provide justification and procedures for their use in additional situations and domains. In practice–commercial, tourism, stock processing, among others–integrating dedicated technological solutions and NLPV–with Cross Domains DevOps–with dedicated agents managing all these resources can represent an LPAA and AI agent oriented architecture [29].

REFERENCES

1. Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., 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, 186345.
2. Kalisetty, S., Lakkarasu, P., Singireddy, S., Burugulla, J. K. R., Challa, K., & Gadi, A. L. (2025, August). Next-Gen Payment Gateways: Leveraging Federated Learning for Fraud Detection in Cross-Border Transactions. In International Conference on Artificial Intelligence: Theory and Applications (pp. 200-211). Cham: Springer Nature Switzerland.
3. Hong, S., Zhuge, M., Chen, J., Zheng, X., Cheng, Y., Wang, J., Zhang, C., Wang, Z., Yau, S. K. S., Lin, Z., Zhou, L., Ran, C., Xiao, L., Wu, C., & Schmidhuber, J. (2024). MetaGPT: Meta programming for a multi-agent collaborative framework. International Conference on Learning Representations.
4. Sivanand, R., Kumar, D. P., Nagabhyru, K. C., Natarajan, E. P., Pamisetty, V., & Kapila, D. (2025, September). IoT and AI for Real-Time Monitoring in Substation Automation. In 2025 International Conference on Computing and Communications (COMPUTINGCON) (pp. 1-5). IEEE.
5. Qian, C., Liu, W., Liu, H., Chen, N., Dang, Y., Li, J., Yang, C., Chen, W., Su, Y., Cong, X., Xu, J., Li, D., Liu, Z., & Sun, M. (2024). ChatDev: Communicative agents for software development. Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics, 15174–15186.
6. Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., Awadallah, A. H., White, R. W., Burger, D., & Wang, C. (2024). AutoGen: Enabling next-gen LLM applications via multi-agent conversation. Proceedings of the First Conference on Language Modeling.
7. Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., & Yao, S. (2024). Reflexion: Language agents with verbal reinforcement learning. Advances in Neural Information Processing Systems, 36, 8634–8652.
8. Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. International Conference on Learning Representations.
9. Schick, T., Dwivedi-Yu, J., Dessì, R., Raileanu, R., Lomeli, M., Zettlemoyer, L., Cancedda, N., & Scialom, T. (2023). Toolformer: Language models can teach themselves to use tools. Advances in Neural Information Processing Systems, 36, 68539–68551.
10. Shen, Y., Song, K., Tan, X., Li, D., Lu, W., & Zhuang, Y. (2023). HuggingGPT: Solving AI tasks with ChatGPT and its friends in Hugging Face. Advances in Neural Information Processing Systems, 36, 38154–38180.
11. Li, G., Hammoud, H. A. A. K., Itani, H., Khizbullin, D., & Ghanem, B. (2023). CAMEL: Communicative agents for "mind" exploration of large scale language model society. Advances in Neural Information Processing Systems, 36, 51991–52008.
12. Nakajima, S., & Buyya, R. (2021). Deep learning for cloud computing: Trends and directions. Journal of Cloud Computing, 10, 39.
13. Li, Z., Zhao, Y., Liu, R., & Pei, D. (2021). Robust and rapid clustering of KPIs for large-scale anomaly detection. IEEE/ACM Transactions on Networking, 29(1), 1–14.
14. Soldani, J., Tamburri, D. A., & Van Den Heuvel, W.-J. (2018). The pains and gains of microservices: A systematic grey literature review. Journal of Systems and Software, 146, 215–232.
15. Kummari, D. N., Challa, S. R., Pamisetty, V., Motamary, S., & Meda, R. (2025). Unifying temporal reasoning and agentic machine learning: A framework for proactive fault detection in dynamic, data-intensive environments. Metallurgical and Materials Engineering, 31(4), 552-568.
16. Balalaie, A., Heydarnoori, A., & Jamshidi, P. (2016). Microservices architecture enables DevOps: Migration to a cloud-native architecture. IEEE Software, 33(3), 42–52.
17. Chen, L. (2015). Continuous delivery: Huge benefits, but challenges too. IEEE Software, 32(2), 50–54.
18. Bandi, V. D. V. K. AI-Based Anomaly Detection Frameworks in Distributed Enterprise Data Systems.
19. Jennings, N. R. (2001). An agent-based approach for building complex software systems. Communications of the ACM, 44(4), 35–41.
20. Wooldridge, M., & Jennings, N. R. (1995). Intelligent agents: Theory and practice. The Knowledge Engineering Review, 10(2), 115–152.
21. Jennings, N. R., Sycara, K., & Wooldridge, M. (1998). A roadmap of agent research and development. Autonomous Agents and Multi-Agent Systems, 1, 7–38.
22. Stone, P., & Veloso, M. (2000). Multiagent systems: A survey from a machine learning perspective. Autonomous Robots, 8, 345–383.
23. Ande, R., Mehta, D., Pandugula, C., Krishna AzithTejaGanti, V., & Kalisetty, S. (2025). Modeling the Som Kamla Amba sub-watershed using Artificial Neural Networks (ANN) and the SWAT tool. Available at SSRN 5341755.
24. Dorri, A., Kanhere, S. S., & Jurdak, R. (2018). Multi-agent systems: A survey. IEEE Access, 6, 28573–28593.
25. 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.

26. Dragoni, N., Giallorenzo, S., Lafuente, A. L., 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.
27. Ebert, C., Gallardo, G., Hernantes, J., & Serrano, N. (2016). DevOps. IEEE Software, 33(3), 94–100.
28. Chen, T.-H., & Jiang, M. (2017). Towards finding and understanding anomalous log file entries. Proceedings of the 39th International Conference on Software Engineering, 124–134.
29. Bandi, V. D. V. K. (2025). Self-Optimizing Data Pipelines Using Machine Learning for Cloud Workloads. Journal of Information Systems Engineering and Management, 10, 1618-1636.