International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Human-Governed Artificial Intelligence Agents for Intelligent Incident Triage in Private Cloud Environments

Shaileshbhai Revabhai Gothi

Abstract: Governance architecture, not model capability, determines whether artificial intelligence agents improve or complicate incident triage outcomes in enterprise private cloud environments. Software-defined data center (SDDC) platforms integrate virtualization, microservices, container orchestration, and AI inference workloads into unified operational ecosystems — each component generating distinct telemetry that engineers must synthesize manually when incidents occur. The result is an information-retrieval bottleneck that extends mean time to triage in proportion to infrastructure scale and heterogeneity. This article proposes a three-tier agent authorization framework that distinguishes agent-eligible knowledge retrieval from human-supervised diagnostic tasks and human-only remediation decisions. An event-driven dispatch mechanism connects infrastructure monitoring telemetry to agent workflows without requiring manual initiation, while a retrieval-augmented generation (RAG) layer grounds all agent outputs in cited operational documentation. Evidence from published AIOps deployments indicates measurable reductions in mean time to triage and knowledge retrieval latency when governance boundaries are explicitly designed into agent workflows. The framework is directly applicable to SDDC lifecycle management environments where event-driven automation infrastructure already exists, enabling incremental AI agent adoption without redesigning operational accountability structures.

Keywords: *AIOps, event-driven automation, human-AI collaboration, incident triage, private cloud infrastructure, software-defined data center management*

1. Introduction

Diagnosing a production incident in a modern software-defined data center is not primarily a technical challenge — it is an information synthesis challenge at enterprise scale. An SDDC platform integrates virtualization layers, software-defined networking, container orchestration, high-performance storage, and AI inference workloads into a single operational ecosystem where events in one layer produce symptoms two or three dependency hops away. A GPU resource contention event in an AI training workload surfaces as an authentication failure in a dependent identity service; a storage volume rebalance triggers microservices timeouts that appear, initially, to be an application-layer regression. Engineers responding to these incidents must construct a coherent diagnostic picture from telemetry distributed across monitoring dashboards, configuration management databases, log aggregation platforms, distributed tracing systems, and documentation repositories — systems that hold overlapping operational context but do not automatically synthesize it.

What makes this difficult is not the absence of data. Modern observability tooling captures infrastructure state at high resolution across all relevant dimensions. The problem is retrieval under time pressure: relevant information exists in too many places, in too many formats, for engineers to assemble it efficiently during an active incident. Senior engineers in mature operations organizations develop pattern recognition that allows them to navigate this fragmentation faster than less experienced colleagues, but that capability is difficult to transfer, impossible to scale, and concentrated in ways that create operational risk when those engineers are unavailable.

Artificial intelligence agents offer a tractable response to this retrieval problem — but only under specific architectural conditions. Deploying agents without governance boundaries in production SDDC environments does not eliminate the fragmentation problem; it replaces it with a different one. Agent recommendations that are plausible but unverified, acted upon under operational pressure, can produce cascading failures more difficult to diagnose than the original incident. This failure mode is what current AIOps architectures inadequately address — and it is the problem this article targets.

The governance-first framework proposed here defines three authorisation tiers: agent-eligible knowledge retrieval, human-supervised diagnostic work, and human-only remediation authority. An event-driven dispatch mechanism connects monitoring telemetry to agent workflows without delegating diagnostic decision-making. A retrieval-augmented generation architecture ensures all agent outputs are grounded in cited, versioned operational documentation. Section 2 establishes the background: operational complexity in modern SDDC environments, the structural limitations of conventional triage, and the current state of AIOps research. Section 3 presents the framework. Section 4 examines the evidence and addresses observed failure modes. Section 5 synthesizes the argument and its operational implications.

Independent Researcher, USA

2. Background And Related Work

2.1 Operational Complexity in Private Cloud Environments

What distinguishes software-defined data centre management from earlier infrastructure paradigms is not scale alone — it is interdependency density. When compute, storage, networking, and virtualization are managed as a unified lifecycle system, operational events propagate across component boundaries in ways that isolated monitoring tools cannot track. Researchers examining enterprise observability architectures contend that the heterogeneity of data sources in modern private clouds creates fragmentation that qualitatively exceeds what traditional correlation approaches can address [1], [2]. The adoption of AI inference and training workloads intensifies this problem: GPU scheduling, distributed storage access patterns, and high-throughput network demands introduce failure modes that differ fundamentally from conventional enterprise application behavior.

Lifecycle management automation has addressed part of this complexity. Desired-state platforms — which continuously reconcile actual infrastructure state against a declared target configuration — reduce the manual configuration burden substantially. They do not, however, address incident triage directly. A desired-state system knows that a host is in a non-compliant state; it does not synthesize the why, identify which other components contributed, or surface the historical context relevant to diagnosis. This distinction between lifecycle automation and incident intelligence is underappreciated in current SDDC operations practice and motivates the framework proposed here.

2.2 Limitations of Conventional Incident Triage

Runbook-based triage assumes that the incident at hand resembles incidents that have occurred before, that relevant procedures are documented, and that an engineer can locate and apply them within the time constraints of an active event. Boddupally establishes that all three assumptions fail with increasing frequency as infrastructure heterogeneity grows — incident patterns diverge from documented cases, documentation accumulates faster than it is curated, and retrieval becomes the primary bottleneck before analysis even begins [3]. The effect is measurable: engineers in complex environments spend a substantial proportion of incident response time locating knowledge rather than applying it.

Knowledge concentration presents a related but structurally distinct risk. When institutional expertise resides primarily in senior engineers rather than accessible documentation, those engineers become unavoidable bottlenecks during high-severity events — regardless of how sophisticated the monitoring infrastructure is. Ahmed et al. demonstrate that the value of AI-assisted knowledge surfacing is highest precisely in these situations: when the relevant expertise is latent in the corpus but not in the room [4]. Runbook staleness compounds both problems: procedures accurate for one software version may actively mislead after an upgrade, and no operational documentation system automatically flags deprecation at the field level [5].

2.3 AIOps and the Governance Problem

Andenmatten's foundational framing of AIOps as the application of data analytics and machine learning to operational event streams established the conceptual space that subsequent architectural work has populated [1]. The research trajectory since has moved from alert correlation and threshold-based detection toward LLM-assisted root cause analysis and generative knowledge synthesis. Chen et al. demonstrate that large language model inference over incident histories and service dependency graphs produces root cause identifications that are accurate and actionable at production scale [6]. Zhang posits that integrating log anomaly detection with LLM-generated runbook outputs reduces diagnostic overhead measurably in mature AIOps pipelines [5].

The governance problem has not received equivalent systematic attention. Most published frameworks treat human oversight as a fallback: a review layer that activates when automated processes fail or produce low-confidence outputs. Zha et al. argue that alert aggregation via LLMs reduces noise substantially — but do not specify decision authority structures for acting on aggregated outputs [7]. This silence on governance architecture is consequential for production environments. In SDDC contexts where automated actions can trigger multi-tier cascades within minutes, the absence of explicit human authority boundaries is not a design gap; it is a risk category.

3. Method: Human-Governed Ai Agent Integration Framework

3.1 Framework Design Principles

Three constraints define the architecture, and their ordering is deliberate. Governance precedes capability: agent action scope is specified by explicit, pre-authorized boundary conditions before any model is selected or deployed. This inverts the approach of most AIOps implementations, which select capable models first and apply governance as a secondary control layer. In SDDC environments, the cost of an unauthorized agent action is asymmetric with the cost of a delayed action — a production host incorrectly remediated may require hours of recovery, while delayed knowledge retrieval costs seconds. That asymmetry justifies the inversion.

Event-driven dispatch follows from governance. Triggering agent workflows on infrastructure telemetry events — metric anomalies, log pattern matches, distributed trace deviations — removes manual-initiation latency from the knowledge retrieval phase without delegating diagnostic authority [8], [9]. The agent responds to an event and produces evidence; the engineer interprets that evidence and decides. Tiered accountability completes the architecture: each operational risk level carries a

defined human authority requirement, and no action at a higher tier can proceed without completing the decision gate at the previous one.

3.2 Three-Tier Agent Authorization Architecture

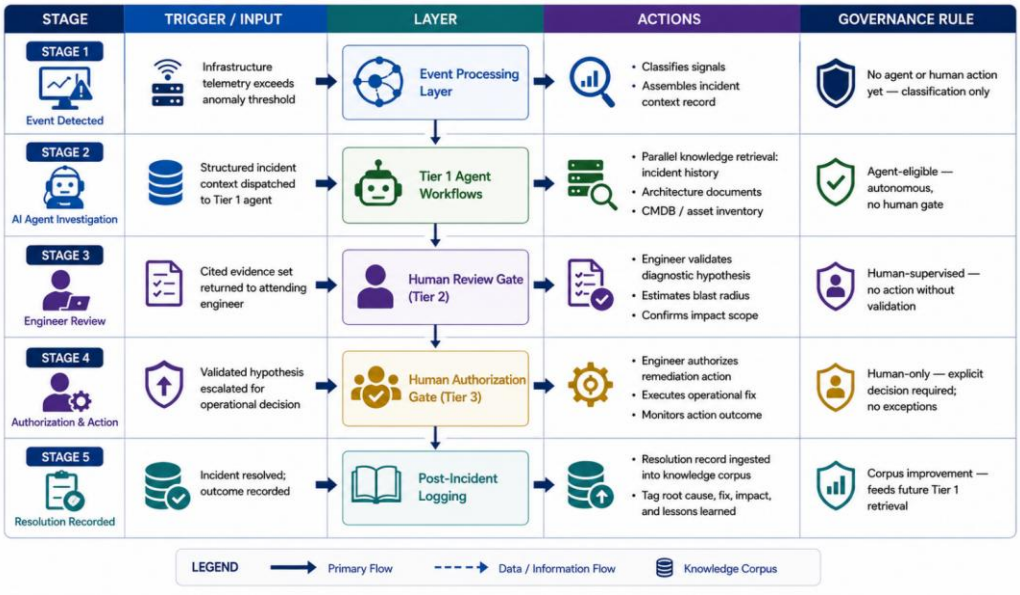
The three tiers reflect a deliberate separation between actions that are informational, actions that are diagnostic, and actions that are operational. Tier 1 encompasses agent-eligible actions: knowledge retrieval, log summarization, configuration state querying, historical incident pattern matching, and distributed trace correlation. These actions are inherently reversible — presenting a document does not modify infrastructure state — and the agent executes them autonomously on event trigger, presenting outputs as attributed, cited evidence.

Tier 2 encompasses human-supervised actions: diagnostic hypothesis generation, multi-source anomaly correlation, blast radius estimation, and failure classification. The agent produces structured analytical output — probable root causes ranked by confidence, related historical incidents, affected component scope — but the engineer validates this output before it informs any decision. Acting on unvalidated Tier 2 output is a governance violation, not a fallback option. Tier 3 is human-only: production environment changes, remediation execution, escalation routing, and any action affecting live service state. Agent involvement at Tier 3 is limited to presenting validated Tier 2 context. Every Tier 3 authorization requires a human decision. No exceptions.

Table 1. Agent Action Authorization Matrix

Incident Category	Tier 1: Agent-Eligible	Tier 2: Human-Supervised	Tier 3: Human-Only	Decision Gate
Network incidents	Log collection, alert correlation, topology query	Root cause hypothesis, path analysis review	Firewall rule changes, routing updates	Tier 2 review → Tier 3 explicit auth
Compute incidents	Host metric snapshot, VM config query	CPU/memory anomaly correlation, blast radius estimate	Host power-off, VM migration, capacity reallocation	Tier 2 review → Tier 3 explicit auth
Storage incidents	Volume state query, I/O trace retrieval	Latency anomaly root cause, dependency mapping	Volume rebalance, snapshot initiation, policy change	Tier 2 review → Tier 3 explicit auth
Application layer	Log summary, trace correlation, error pattern match	Service failure classification, dependency graph review	Deployment rollback, service restart, config override	Tier 2 review → Tier 3 explicit auth
AI workload infra	GPU utilization snapshot, job queue state retrieval	Resource contention hypothesis, cross-job impact estimate	Job preemption, resource reallocation, scheduler override	Tier 2 review → Tier 3 explicit auth

Figure 1. Three-Tier Governance Flowchart — Incident Event to Resolution



3.3 Event-Driven Agent Dispatch Mechanism

The dispatch mechanism operates as a reactive pipeline structured in three stages. Stage one: infrastructure monitoring telemetry — including metrics from compute, storage, and networking layers; log streams from application services and platform components; and distributed trace records from microservices interfaces — is consumed by an event processing layer that classifies incoming signals against incident detection patterns. Stage two: when a signal set meets threshold conditions (configurable per component type and incident severity tier), the event processor assembles a structured incident context record containing affected component identifiers, telemetry snapshots at the moment of anomaly, current configuration state, and alert metadata. Stage three: the context record is dispatched to the appropriate Tier 1 agent workflow, which initiates knowledge retrieval queries in parallel against the operational knowledge corpus.

Parallel dispatch merits specific emphasis. Sequential manual investigation queries documentation systems one at a time. The event-driven agent framework dispatches retrieval queries to multiple knowledge sources simultaneously — incident history, architecture documentation, configuration records, security advisories — and returns a consolidated evidence set within seconds of the incident event. In SDDC environments where incident propagation across service tiers can occur within minutes of the originating event, this compression of the knowledge retrieval phase is operationally material [8], [10].

3.4 Knowledge Retrieval Architecture

The RAG integration layer connects agent dispatch outputs to an operational knowledge corpus. This corpus encompasses architecture documentation maintained in version control, historical incident postmortems with resolution records, configuration management database exports, runbooks, change management records, and security advisories. When a Tier 1 agent workflow receives an incident context record, it formulates semantically targeted queries against the corpus using the affected component identifiers and telemetry anomaly patterns as retrieval keys [11].

Every agent output from this layer carries source attribution: each retrieved passage is linked to its origin document, document version, and retrieval timestamp. This serves two purposes. First, it enables the attending engineer to verify the relevance and currency of each retrieved passage before treating it as diagnostic evidence. Second, it creates an auditable record of the knowledge accessed during the incident, which supports post-incident review and corpus improvement. The hallucination problem that characterizes unconstrained language model inference does not apply in this architecture because the agent produces no assertions — it surfaces cited evidence [12]. This distinction between evidence surfacing and claim generation is the framework's critical operational safety property, and it is what differentiates the RAG integration from standard LLM-assisted operations.

Table 2. Knowledge Retrieval Mechanism Comparison: Keyword Search vs. RAG-Based Agent Retrieval

Criterion	Keyword Search	RAG-Based Agent Retrieval
Retrieval latency	Proportional to query complexity; sequential across repositories	Near-zero initiation; parallel dispatch to multiple corpus segments
Relevance precision	Dependent on keyword match; misses semantic equivalents	Semantically targeted; component and anomaly pattern keys improve precision
Hallucination risk	N/A — returns verbatim document excerpts only	Eliminated in citation-grounded mode; agent surfaces passages, not generated assertions
Source attribution	Filename/URL returned; no version or retrieval timestamp	Full attribution: origin document, document version, retrieval timestamp
Multi-repository integration	Requires separate query per repository; no cross-source synthesis	Single query dispatched across all corpus segments; consolidated result set returned

4. Results And Discussion

4.1 Operational Effectiveness

Published empirical evidence on LLM-assisted triage converges on a consistent finding: agent involvement in knowledge retrieval reduces mean time to triage across cloud environments at scale, and the primary mechanism is retrieval latency compression, not superior diagnostic reasoning [6], [4]. Engineers accessing AI-synthesized, cited knowledge arrive at the diagnostic phase of investigation faster than engineers conducting manual multi-repository searches — the agent eliminates the information-retrieval work that precedes diagnosis [3], [13].

The three-tier structure contributes a measurement advantage that single-tier approaches lack. Because agent involvement at Tier 1 is independently measurable from human decision quality at Tier 2, operations teams can assess the framework's contribution at each stage. Tier 1

retrieval latency improvements can be tracked without waiting for end-to-end resolution time data — which depends on factors well outside the framework's scope. This incremental measurability is itself an operational advantage: it enables evidence-based decisions about when and whether to expand agent involvement to Tier 2 diagnostic tasks.

Table 3. Key Performance Indicators: Conventional vs. AI Agent-Assisted Incident Triage

KPI	Conventional Triage	AI Agent-Assisted (Governed)	Evidence
Mean Time to Triage (MTTT)	Baseline (manual multi-repository search)	Measurably reduced; primary driver is knowledge retrieval latency compression	[6], [4]
Mean Time to Knowledge (MTTK)	Proportional to corpus size and query complexity	Near-zero via event-driven parallel dispatch; sustained across incident types	[8], [11]
False Positive Escalation Rate	Baseline (unfiltered alert volume)	Reduced through LLM-assisted alert aggregation at Tier 1	[7]
Incorrect Remediation Rate	Elevated in fully autonomous systems	Substantially reduced with human validation gate at Tier 2/3 boundary	[14]

4.2 Governance Boundary Effectiveness

The case for explicit governance boundaries is strongest when examined through the failure modes of unbounded automation. Zhang et al. demonstrate in the CloudRCA deployment that human validation of root cause hypotheses before remediation significantly reduces incorrect remediation attempts — and that the operational cost of incorrect remediation in production cloud environments substantially exceeds the cost of a validation delay [14]. D'Antonio and Xie establish that human-in-the-loop runbook improvement processes produce measurably more reliable operational guidance than autonomous generation processes, reinforcing the argument that human judgment adds value at the Tier 2 boundary that agent capability alone cannot replicate [13].

The governance architecture also produces a compliance advantage that is operationally significant for enterprise SDDC deployments. Production environments operating under regulatory or contractual requirements mandating human authorization for infrastructure changes are structurally compatible with the three-tier model from the outset. Organizations implementing governance-first AIOps systems avoid the retrofit complexity that arises when continuous learning pipelines must be retroactively constrained to accommodate audit requirements [15]. Building the decision gates in from the start is architecturally simpler than adding them later.

4.3 Limitations and Failure Modes

Three operational failure modes require explicit acknowledgment. Alert fatigue is the most immediate: when Tier 1 dispatch thresholds are set too broadly, the volume of agent-generated knowledge summaries can exceed engineers' capacity to process them productively, recreating the cognitive overload the framework was designed to reduce [7]. Threshold calibration is an operational tuning problem that the framework's architecture cannot solve in the abstract — it requires ongoing adjustment against the specific incident patterns of the deployment environment.

Knowledge corpus quality represents a deeper structural dependency. Retrieval-augmented generation performs no better than the corpus it queries. Documentation that is outdated, incomplete, or inconsistently structured degrades retrieval relevance regardless of model quality [16], [17]. Addressing this requires organizational investment in knowledge management that is entirely independent of the technical framework — it is a governance challenge, not a technology challenge.

Organizational resistance should not be dismissed as irrational. Engineers who have developed reliable manual investigation workflows have legitimate reasons to approach agent-assisted alternatives with caution. The appropriate response is incremental deployment with transparent output attribution, beginning with Tier 1 knowledge retrieval only [18]. Trust is built through demonstrated reliability, not through proclamation.

4.4 Discussion

The central practical implication of this analysis is straightforward: organizations implementing AIOps capabilities face a governance design problem before they face a technology selection problem. The models and retrieval architectures required to implement the proposed framework exist as deployable, production-grade components. What organizations must design is the boundary architecture — where agent authority ends, where human authority begins, and how decisions flow between them. That work is primarily operational and organizational.

SDDC lifecycle management environments that already implement event-driven automation have a specific structural advantage. The telemetry streams that drive lifecycle management decisions — metrics, configuration

events, state reconciliation signals — are structurally identical to the streams that trigger agent dispatch in the proposed framework. The event processing infrastructure already exists; the integration point is a matter of directing existing telemetry to a new consumer [2], [19], [20]. Organizations that have invested in SDDC automation are therefore positioned to adopt governed AI triage incrementally, without redesigning their operational architecture from the ground up.

5. Conclusion

The operational challenge of incident triage in private cloud environments is fundamentally an information architecture problem: the relevant knowledge exists, distributed across systems that do not automatically synthesize it, and the engineers who need it are under time pressure. AI agents designed to retrieve and surface cited evidence from distributed knowledge sources address this problem directly — but only within governance structures that preserve human authority over the diagnostic and remediation decisions that determine incident outcomes.

The three-tier authorization framework presented here defines those structures explicitly. Agent-eligible knowledge retrieval operates within a governance boundary that prevents agent involvement from expanding into diagnostic judgment or remediation action. Event-driven dispatch eliminates retrieval initiation latency without eliminating the human decision gates that production SDDC environments require. Retrieval-augmented generation ensures that every agent output is grounded in cited documentation, not generated inference.

Governance architecture is the design variable that determines whether AI agents improve incident triage. Not model capability. Not retrieval system sophistication. The organizations that will realize the operational benefits of AI-assisted triage most durably are those that invest in governance design first — and treat model selection as a secondary consideration.

Acknowledgments

The author acknowledges the broader community of researchers and practitioners whose published work on AIOps, distributed systems observability, and human-AI collaboration frameworks provided the empirical foundation for this analysis.

Author Contributions

Conceptualization, S.R.G.; methodology, S.R.G.; writing — original draft, S.R.G.; writing — review and editing, S.R.G.

Conflicts Of Interest

The author declares no conflict of interest.

References

[1] Cisco, "What is AIOps?," HMD Praxis der Wirtschaftsinformatik, vol. 56, no. 5, pp. 345–346, 2019. https://www.cisco.com/site/us/en/learn/topics/artificial-intelligence/what-is-aiops.html

[2] L. Ben-Shimol, O. Raz, G. Mishne, and M. Shanbhag, "Observability and Incident Response in Managed Serverless Environments Using Ontology-Based Log Monitoring," IEEE Trans. Cloud Comput., 2025. https://ieeexplore.ieee.org/document/11122875

[3] H. L. Boddupally, "Automating Incident Triage and Root Cause Intelligence Through Large Language Model–Driven Correlation of System Logs and Operational Metrics in Large-Scale Distributed Environments," Int. J. Eng. Extended Technol. Res. (IJEETR), vol. 5, no. 6, 2023. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=6736978

[4] T. Ahmed, S. Ghosh, C. Bansal, T. Zimmermann, X. Zhang, and S. Rajmohan, "Recommending Root-Cause and Mitigation Steps for Cloud Incidents using Large Language Models," in Proc. 2023 IEEE/ACM 45th Int. Conf. Softw. Eng. (ICSE), 2023. https://arxiv.org/abs/2301.03797

[5] H. Zhang, "A Unified AIOps Pipeline for Joint Log–KPI Anomaly Detection, Graph-Based Root Cause Localization, and LLM-Generated Runbooks," J. Adv. Comput. Syst., vol. 4, no. 1, 2024. https://scipublication.com/index.php/JACS/article/view/277

[6] Y. Chen, Z. Kang, C. Liu, T. Li, Y. Su, Y. Zhao, X. Qu, Y. Xu, C. Bi, Z. Wang, and X. Zhang, "Automatic Root Cause Analysis via Large Language Models for Cloud Incidents," in Proc. Nineteenth Eur. Conf. Comput. Syst. (EuroSys '24), ACM, 2024. https://arxiv.org/abs/2305.15778

[7] J. Zha, X. Shan, J. Lu, J. Zhu, and Z. Liu, "Leveraging Large Language Models for Efficient Alert Aggregation in AIOPs," Electronics, vol. 13, no. 22, p. 4425, 2024. https://www.mdpi.com/2079-9292/13/22/4425

[8] A. Rahmatulloh, F. Nugraha, R. Gunawan, and I. Darmawan, "Event-Driven Architecture to Improve Performance and Scalability in Microservices-Based Systems," in 2022 IEEE Int. Conf. Advancement in Data Science, E-learning and Information Systems (ICADEIS), pp. 01–06, 2022. https://ieeexplore.ieee.org/document/10037390

[9] Z. Purfallah Mazraemolla and A. Rasoolzadegan, "An effective failure detection method for microservice-based systems using distributed tracing data," Eng. Appl. Artif. Intell., 2024. https://www.sciencedirect.com/science/article/abs/pii/S0952197624007164

[10] M. Raeiszadeh, A. Ebrahimzadeh, A. Saleem, R. H. Glitho, J. Eker, and R. A. F. Mini, "Real-Time Anomaly Detection Using Distributed Tracing in Microservice Cloud Applications," in 2023 IEEE 12th Int. Conf. Cloud Networking (CloudNet), 2023.

https://lup.lub.lu.se/search/files/165150749/Real_Time_Anomaly_Detection_Using_Distributed_Tracing_in_Microservice_Cloud_Applications.pdf

[11] S. Zhang, D. Fan, and L. He, "Large Language Models Empowered Online Log Anomaly Detection in AIOps," in Proc. 2024 31st Asia-Pacific Softw. Eng. Conf. (APSEC), pp. 402–411, 2024. https://ieeexplore.ieee.org/document/10967310

[12] X. Wang, X. Liu, P. Xu, and H. Du, "Graph-based Anomaly Detection and Root Cause Analysis for Microservices in Cloud-Native Platform," in 2024 IEEE Int. Conf. Sustainable Computing and Communications (SustainCom), 2024. https://ieeexplore.ieee.org/document/10917910

[13] R. D'Antonio and H. Xie, "Human-in-the-Loop Runbook Improvement with Agentic Support Automation," in 2025 IEEE 7th Int. Conf. Cognitive Machine Intelligence (CogMI), 2025. https://ieeexplore.ieee.org/document/11417021

[14] Y. Zhang, X. He, X. Zheng, M. Qiu, Q. Lin, H. Zhang, Q. Zhang, L. Li, S. Dang, M. R. Lyu, and D. Zhang, "CloudRCA: A Root Cause Analysis Framework for Cloud Computing Platforms," in Proc. 30th ACM Int. Conf. Inf. & Knowl. Mgmt. (CIKM), pp. 4107–4116, 2021. https://arxiv.org/abs/2111.03753

[15] L. An, A.-J. Tu, X. Liu, and R. Akkiraju, "Real-time Statistical Log Anomaly Detection with Continuous AIOps Learning," in Proc. 12th Int. Conf. Cloud Comput. and Services Sci. (CLOSER), 2022. https://www.scitepress.org/Papers/2022/110692/110692.pdf

[16] L. M. Barata, S. Sequeira, E. Lopes, P. R. M. Inácio, and M. M. Freire, "Anomaly detection and root-cause identification in microservices: a survey," Cluster Comput., Springer, 2026. https://link.springer.com/article/10.1007/s10586-026-06095-9

[17] P. Moens, B. Andriessen, M. Sebrechts, B. Volckaert, and S. Van Hoecke, "Edge Anomaly Detection Framework for AIOps in Cloud and IoT," in Proc. 13th Int. Conf. Cloud Comput. and Services Sci. (CLOSER), 2023. https://www.scitepress.org/Papers/2023/118386/118386.pdf

[18] D. Toprani and V. K. Madisetti, "LLM Agentic Workflow for Automated Vulnerability Detection and Remediation in Infrastructure-as-Code," IEEE Access, 2025. https://ieeexplore.ieee.org/document/10965635

[19] M. Soualhia and F. Wuhib, "Automated Traces-based Anomaly Detection and Root Cause Analysis in Cloud Platforms," in 2022 IEEE Int. Conf. Cloud Engineering (IC2E), 2022. https://ieeexplore.ieee.org/document/9946356

[20] X. Peng, "Large-Scale Trace Analysis for Microservice Anomaly Detection and Root Cause Localization," in Proc. Federated Africa and Middle East Conf. Softw. Eng. (FAMSE), 2022. https://dl.acm.org/doi/10.1145/3531056.3542765