Stout in Computer Science and Technology Studies (SCSTS)

DOI: https://doi.org/10.61424/zngee941
Journal Homepage: https://stoutjournals.org/index.php/SCSTS

Process-Aware LLM-Agent Scaffolds for Metric-Based Microservice Root-Cause Analysis with Evidence-Trace Scoring

Matthew Liao
Computer Science, Rutgers University, New Brunswick, NJ, USA
✉ **Corresponding Author**: Matthew Liao, **E-mail**: mliao_rutgers@outlook.com

ARTICLE INFORMATION

Submitted: April 10, 2026
Accepted: June 23, 2026
Published: July 19, 2026
Volume: 1
Issue: 1

KEYWORDS

AIOps; microservices; root-cause analysis; LLM agents; metric telemetry; evidence grounding; process evaluation; fault diagnosis

ABSTRACT

Root-cause analysis in microservice systems increasingly relies on agents that retrieve telemetry, form hypotheses, and explain a diagnosis. Final-answer accuracy alone does not reveal whether the cited evidence supports the conclusion. This study evaluated a process-aware diagnostic scaffold for large-language-model agents on the metric-only RE1-OB corpus from RCAEval. The corpus contains 125 fault-injection cases arranged as five root services, five fault types, and five repetitions. Each case was represented by equal 300-second pre-injection and post-injection windows. To prevent acquisition and schema leakage, the primary predictors were restricted to 35 CPU, memory, and workload streams present in every case; timestamps, file duration, column count, and missingness patterns were excluded. Five leave-one-repetition-out folds compared six conventional classifiers with a tri-head tree ensemble that combined joint service–fault prediction with separate localization and fault-identification heads. The process-aware model achieved 0.960 joint accuracy (95% bootstrap confidence interval [0.920, 0.992]), 0.957 macro-F1, 0.968 service accuracy, 0.992 fault accuracy, and 0.992 top-three accuracy. Injection-aligned evidence retrieval reached 0.952 root-service recall at five. Reranking evidence against the predicted service increased recall to 1.000, reason score from 0.653 to 0.719, and evidence-trace score from 0.871 to 0.893. Five errors remained, dominated by loss-fault service confusion. The findings show that strong diagnosis and auditable evidence are separable objectives: a structured retrieval and verification layer added measurable grounding without changing the underlying top-one predictions.

1. Introduction

Microservice modularity makes operational failures difficult to diagnose because a fault in one service changes downstream behavior. RCA therefore requires localization—where the fault originated—and identification—what fault occurred. Classical methods use correlations, causal graphs, and topology (Lin et al., 2018; Wu et al., 2020), while later systems combine metrics, logs, and spans (Lee et al., 2023; Zhang et al., 2023). Benchmark design and observability coverage materially affect the reported performance (S. Zhang et al., 2025).

Large language models (LLMs) extend RCA from ranking to interactive investigation. An agent can issue monitoring queries, inspect returned observations, revise a hypothesis, and produce a human-readable conclusion. ReAct established the general pattern of interleaving reasoning and action (Yao et al., 2023), and tool-learning research showed that language models can construct executable API paths (Qin et al., 2024). RCA-specific systems subsequently connected agents to telemetry and incident repositories. Studies of cloud incidents demonstrated diagnostic aggregation and autonomous tool use (Chen et al., 2024; Roy et al., 2024; Wang et al., 2024), while Flow-of-Action constrained multi-agent diagnosis with standard operating procedures (Pei et al., 2025). These systems make the investigation path an operational artifact rather than an invisible prelude to an answer.

This change exposes an evaluation gap. A correct final label can follow irrelevant queries, incorrect intermediate statistics, or unsupported prose (Jin, 2025; R. Zhang et al., 2025). Conversely, an agent can retrieve the right anomalous component yet

misname the fault. Outcome-oriented metrics collapse these behaviors into one score. Recent benchmarks address the gap by separating localization, identification, and reason. AgenticOpsEval uses key evidence and typed causal chains across AIOps2025 and RCA100 (Cai et al., 2026), OpenRCA 2.0 introduces process supervision (Fang et al., 2026), and live environments such as AIOpsLab, ITBench, and SREGym examine tool interaction under executable conditions (Chen et al., 2025; Clark et al., 2026; Jha et al., 2025). Their common implication is that the diagnostic record must be scored as well as the answer.

The present study applies this process-awareness principle to the Online Boutique subset of RCAEval's RE1 corpus (Pham, 2025; Pham et al., 2025). RE1-OB contains metric streams, injection times, and service–fault labels. It does not contain log text, span records, or expert causal-chain annotations. Accordingly, "evidence trace" in this paper denotes an ordered and auditable record of metric queries, returned statistics, hypothesis updates, and the final decision. It does not denote a distributed tracing span. This distinction keeps the empirical claims aligned with the available observations while testing a diagnostic substrate that an LLM agent can call and narrate.

The study makes three contributions. First, it constructs a leakage-resistant, incident-level evaluation from all 125 cases using identical pre/post windows and five held-out-repetition folds. Second, it evaluates a process-aware tri-head decision scaffold against random, centroid, linear, kernel, and tree baselines. Third, it defines evidence-trace scoring that separately measures root-service evidence ranking, claim support, temporal grounding, localization, and fault identification. The resulting analysis answers three questions: how accurately can shared metric channels identify 25 service–fault combinations; how much does ordered contrastive retrieval improve evidence quality; and which fault and service conditions remain unreliable despite high aggregate performance?

Figure 1 summarizes the ordered diagnostic scaffold and the audit record retained from incident anchoring through evidence verification.

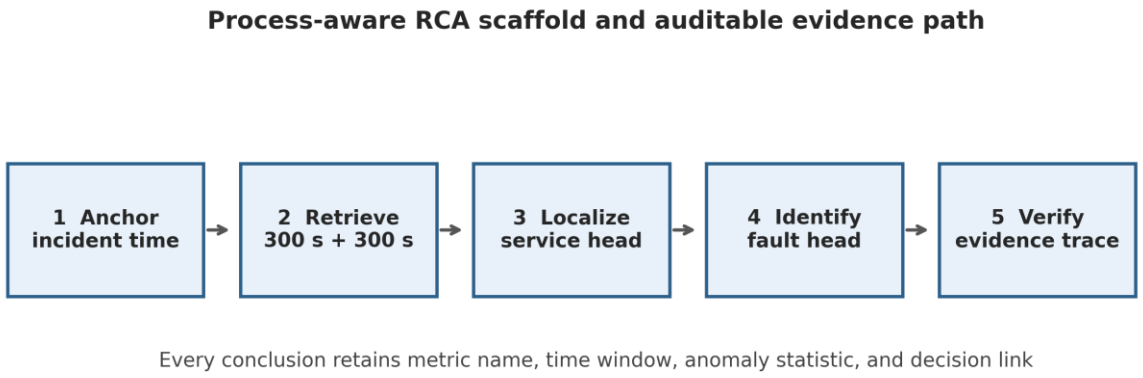

Figure 1. Process-aware diagnostic scaffold and auditable evidence path

2. Literature Review

Early RCA methods modeled propagation through graphs and dependencies. Microscope used causal graphs (Lin et al., 2018), MicroRCA combined metrics with topology-aware correlations (Wu et al., 2020), and MicroRank used trace spectra for latency localization (Yu et al., 2021). Causal-discovery research treated injected faults as interventions (Ikram et al., 2022). Together, these methods show that contextual and temporal relevance is stronger than raw anomaly magnitude.

Eadro, DiagFusion, and Nezha broadened RCA with metrics, logs, traces, and dependency graphs (Lee et al., 2023; Yu et al., 2023; Zhang et al., 2023). Metric-only RCA remains important where tracing is absent. BARO used Bayesian online change-point detection for this setting (Pham et al., 2024), while public dataset research showed that collection regimes affect comparison (Li et al., 2022). The current experiment therefore treats metric-only diagnosis as a bounded task, not a proxy for multimodal performance (Xin, 2025).

LLMs added semantic reasoning over tools and operational knowledge. Chain-of-thought exposed intermediate text without guaranteeing faithfulness (Wei et al., 2022); ReAct grounded steps in action observations (Yao et al., 2023); and ToolLLM studied API selection (Qin et al., 2024). OpsEval assessed operations knowledge (Liu et al., 2023), while OpenRCA tested language-model localization on software failures (Xu et al., 2025). These benchmarks advanced beyond generic question answering but initially emphasized outcomes over step validity.

Operational agents then incorporated domain-specific retrieval and collaboration. RCACopilot aggregated diagnostic information from cloud incidents and generated explanations (Chen et al., 2024). RCAgent used tool-augmented autonomous agents and trajectory self-consistency (Wang et al., 2024). A ReAct-based production study examined how agents searched incident evidence (Roy et al., 2024), and mABC coordinated specialized agents through a bounded collaboration protocol (Zhang et al., 2024). TAMO added tool-assisted processing of multimodal cloud-native observations (X. Zhang et al., 2025). Flow-of-Action most directly motivates the present process design because it uses a standard operating procedure to constrain the sequence of investigative actions (Pei et al., 2025). Reflection-guided multi-agent search extends the same idea by treating diagnosis as movement among evidence-bearing states (Naakka et al., 2026).

AIOpsLab combines deployment, fault injection, telemetry, and agent interfaces (Chen et al., 2025). ITBench spans several operational domains (Jha et al., 2025), and SREGym supplies live noisy failures (Clark et al., 2026). AgenticOpsEval divides diagnosis into localization, identification, and reason through key evidence and causal chains (Cai et al., 2026); OpenRCA 2.0 adds causal-process supervision (Fang et al., 2026). These resources establish the value of process labels, but their multimodal tasks differ from RE1-OB (Zhong et al., 2026).

RCAEval supplies the empirical bridge. Its full benchmark contains three datasets, multiple microservice systems, and reproducible RCA baselines (Pham et al., 2025). RE1 contains 375 metric-only failures, of which RE1-OB contributes 125 Online Boutique cases arranged across five injected services, five fault types, and five repetitions (Pham, 2025). The balanced design supports controlled held-out-repetition testing, yet it also presents risks: small class counts, repeated laboratory conditions, and schema differences across collection periods. This study responds by keeping the case—not individual seconds—as the experimental unit, excluding collection metadata, and distinguishing primary shared-channel results from schema sensitivity. The objective is not to claim a universal agent benchmark. It is to measure how an ordered metric investigation changes both RCA accuracy and evidence quality within a clearly defined corpus.

3. Methodology

The evaluation used every RE1-OB service–fault combination. Directory labels supplied the root service and fault; inject_time.txt supplied the incident anchor. Five services and five fault types—CPU, memory, disk, delay, and loss—each had five repetitions, producing 125 cases. Table 1 reports the factorial structure, and Figure 2 visualizes it.

Table 1. Balanced factorial composition of RE1-OB

Root service	CPU	MEM	DISK	DELAY	LOSS	Total
adservice	5	5	5	5	5	25
cartservice	5	5	5	5	5	25
checkoutservice	5	5	5	5	5	25
currencyservice	5	5	5	5	5	25
productcatalogservice	5	5	5	5	5	25
Total	**25**	**25**	**25**	**25**	**25**	**125**

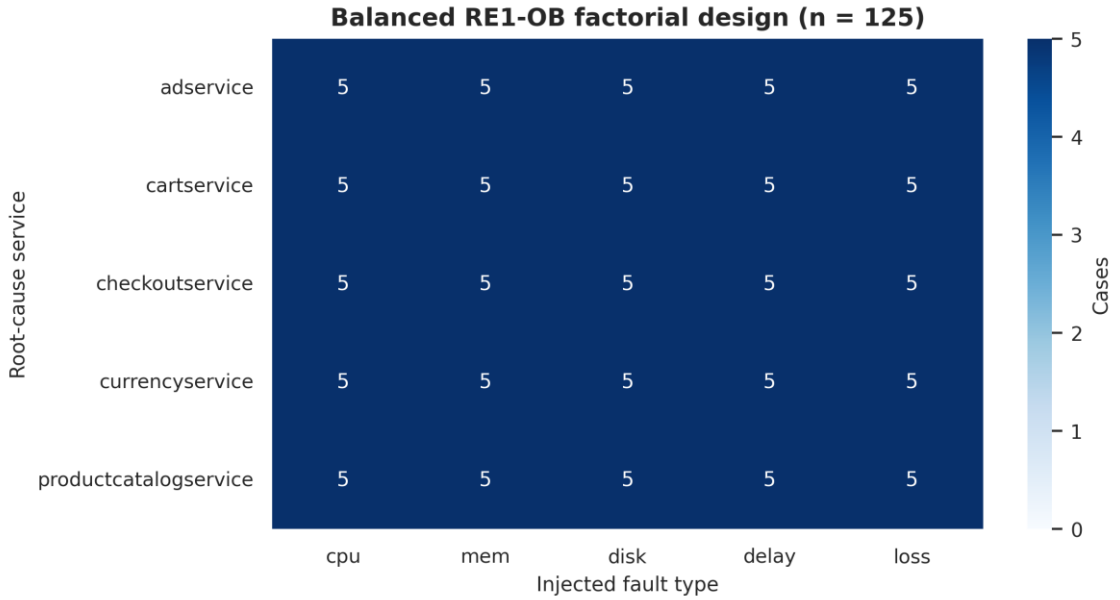

Figure 2. Balanced service–fault composition of the 125-case RE1-OB corpus

The archive contained 262,751 raw CSV rows. Numeric parsing of the first timestamp column retained 262,467 rows and removed 284 rows with invalid timestamps from three CPU cases. Every case still supplied at least 300 valid seconds on both sides of injection. The analytic sample consequently contained exactly 75,000 one-second observations: 600 observations for each of 125 incident-level units. Duplicate timestamp fields were removed. The label aliases "load" and "workload" were normalized to one name. The canonical union contained 84 metric streams, but only 35 streams appeared in all cases: 12 CPU, 12 memory, and 11 workload streams. Table 2 lists these measured quality properties and the corresponding analytic treatment.

Table 2. Measured data quality and leakage-safe harmonization

Measure	Observed value	Analytic treatment
Failure cases	125	Incident-level experimental units
Raw time-series rows	262,751	All CSV observations
Valid timestamp rows	262,467	Primary timestamp parsed numerically
Invalid timestamp rows	284	Excluded before windowing; present in 3 cases
Analytic rows	75,000	600 one-second observations per case
Canonical metric union	84	Used only for within-case evidence retrieval
Common predictor metrics	35	12 CPU, 12 memory, 11 workload
Window	300 s before + 300 s after	Equal duration for every case

The common 35 streams formed the predictor space, preventing learning from file length, timestamp, header width, latency convention, or missingness. The 84-stream union served only for within-case evidence ranking; a sensitivity run admitted it as predictors. Because no common disk-utilization stream exists, disk diagnosis relied on service responses rather than a direct disk gauge.

The process-aware scaffold followed seven deterministic stages: anchor, retrieve, contrast, localize, identify, verify, and report. The anchor fixed the injection time. Retrieval selected [t0−300, t0−1] as baseline and [t0, t0+299] as the observed window. Contrast converted each metric into robust statistics. The localization, identification, and joint heads produced class probabilities. Verification reranked concrete metric observations against the predicted service and fault, and reporting retained the service–

fault claim with five evidence items. Figure 1 presents the flow, while Table 3 records the input, output, and audit fields retained at each stage.

Table 3. Process-aware diagnostic stages and retained audit fields

Stage	Input	Output	Retained fields
Anchor	inject_time.txt	Incident time t0	case, t0
Retrieve	35 common metric streams	Paired pre/post windows	metric, interval
Contrast	Median, dispersion, tail	Five robust statistics per metric	value, direction
Localize	Service and joint heads	Root-service posterior	ranked services
Identify	Fault and joint heads	Fault-type posterior	ranked faults
Verify	Available canonical metrics	Five cited evidence items	metric, score, rank
Report	Diagnosis + evidence	Service–fault claim	ordered action record

For each metric, the pre-injection median defined the center. Scale was the maximum of baseline robust dispersion, standard deviation, 2% of the baseline level, and 10−6, limiting extreme ratios in constant gauges. Five features captured signed shift, mean and tail deviation, dispersion change, and relative shift. Table 4 gives the formulas. Figure 3 plots the resulting trajectories for the first adservice CPU case over its fixed 600-second window.

Table 4. Injection-aligned feature definitions

Feature	Definition	Purpose
Signed shift	clip[(medianpost − medianpre)/s, −50, 50]	Direction and magnitude
Mean absolute deviation	mean(|(xpost − medianpre)/s|)	Sustained change
Tail deviation	Q0.95(|(xpost − medianpre)/s|)	Peak behavior
Dispersion ratio	log[(SDpost + .01s)/(SDpre + .01s)]	Volatility change
Relative shift	(medianpost − medianpre)/(|medianpre| + 10−6)	Scale-relative change
Robust scale s	max(1.4826 MADpre, SDpre, .02|medianpre|, 10−6)	Stable denominator
Evidence anomaly	|signed shift| + .25 mean abs. + .10 tail	Metric ranking

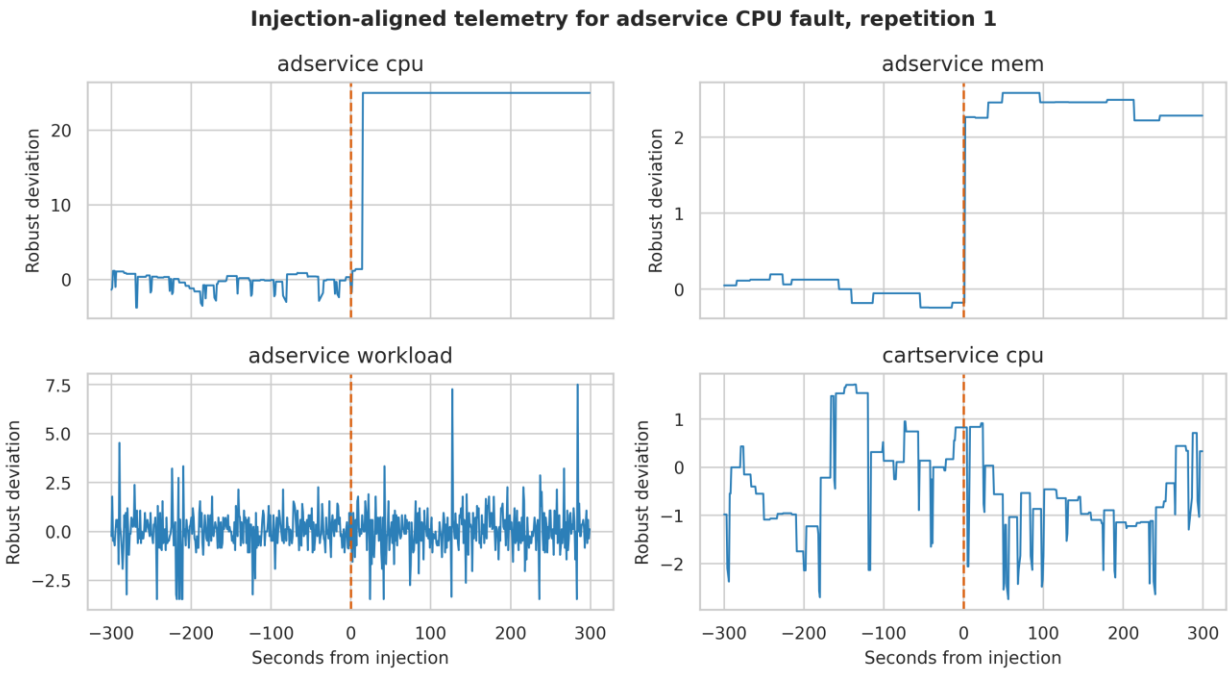

Figure 3. Injection-aligned metric deviations for adservice CPU fault, repetition 1

Seven decision methods were compared. The uniform random classifier established a chance reference. Nearest centroid, multinomial logistic regression, and an RBF support-vector machine represented geometric, linear, and kernel decision rules. Random forest and Extra Trees represented nonlinear ensembles. The process-aware tri-head combined an Extra Trees joint head with separate Extra Trees service and fault heads. The joint head used 500 trees; each decomposed head used 750 trees. For a candidate service–fault pair, the separate-head probability was the product of its service and fault probabilities. The final posterior was 0.75 times the joint posterior plus 0.25 times the decomposed posterior. Table 5 states the complete fixed configurations. No directory name, repetition number, injection timestamp, or file-shape variable was included as a feature.

Table 5. Compared decision models and fixed hyperparameters

Method	Estimator	Configuration
Uniform random	Stratified dummy	Seed 20260717 + fold
Nearest centroid	Standardized Euclidean centroid	Fold-fitted scaling
Logistic regression	Multinomial L2	C=1; balanced; 5,000 iterations
RBF SVM	Kernel classifier	C=1; gamma=scale; balanced
Random forest	Bagged trees	500 trees; sqrt features; leaf=1
Extra Trees	Randomized trees	500 trees; sqrt features; leaf=1
Process-aware tri-head	Joint + service × fault ensemble	500/750 trees; weights .75/.25

Cross-validation left out one repetition index at a time. In fold k, repetition k from all 25 service–fault pairs formed a 25-case test set, and the other 100 cases formed the training set. Median imputation and standardization, when required, were fitted only on the training portion. The five folds yielded one out-of-fold prediction for every case. Primary outcome measures were joint top-one accuracy, joint macro-F1, top-three accuracy, service accuracy, fault accuracy, and log loss. Nonparametric 95% confidence intervals used 5,000 case bootstrap samples with random seed 20260717. Two-sided exact McNemar tests compared paired correctness against selected baselines.

Evidence evaluation used the same out-of-fold predictions. Each available metric received anomaly score a=|signed shift|+0.25(mean absolute deviation)+0.10(tail deviation). Four evidence policies were evaluated. Outcome only returned no metric items. Post-only volatility ranked the coefficient of variation inside the post-injection window. Injection-aligned contrast

ranked a. Process-aware reranking multiplied a by 1.75 when the metric belonged to the predicted service and by 1.15 when its metric family matched the predicted fault label. The top five items were retained.

Evidence relevance used the labeled root service because no expert causal-chain file exists. Precision@5 measured the true-service fraction, Recall@5 any true-service hit, and binary nDCG@5 its rank. Claim support measured predicted-service evidence; temporal grounding required explicit pre/post contrast. Reason was 0.50(nDCG)+0.20(precision)+0.20(claim support)+0.10(temporal grounding). ETS averaged service correctness, fault correctness, and reason, so evidence quality could not conceal an incorrect diagnosis.

4. Results and Discussion

The common-channel experiment separated the models clearly. Table 6 gives out-of-fold performance for all seven methods, and Figure 4 shows accuracy, macro-F1, and top-three accuracy. Uniform random prediction achieved 0.032 accuracy. The RBF SVM reached 0.608, logistic regression 0.776, nearest centroid 0.912, random forest 0.944, and Extra Trees 0.960. The process-aware tri-head preserved the Extra Trees top-one decisions and also achieved 0.960 accuracy, 0.957 macro-F1, and 0.992 top-three accuracy. Its mean fold accuracy was 0.960 with a 0.028 standard deviation. The 95% bootstrap interval for accuracy was [0.920, 0.992], and the macro-F1 interval was [0.914, 0.987].

Table 6. Out-of-fold joint 25-class RCA performance

Model	Accuracy	Macro-F1	Top-3	Log loss	Fold accuracy
Uniform random	0.032	0.030	0.112	34.890	0.032 ± 0.033
Nearest centroid	0.912	0.912	0.984	2.117	0.912 ± 0.018
Logistic regression	0.776	0.774	0.952	0.699	0.776 ± 0.061
RBF SVM	0.608	0.597	0.808	2.067	0.608 ± 0.131
Random forest	0.944	0.941	1.000	0.747	0.944 ± 0.022
Extra Trees	0.960	0.957	0.992	0.504	0.960 ± 0.028
Process-aware tri-head	0.960	0.957	0.992	0.579	0.960 ± 0.028

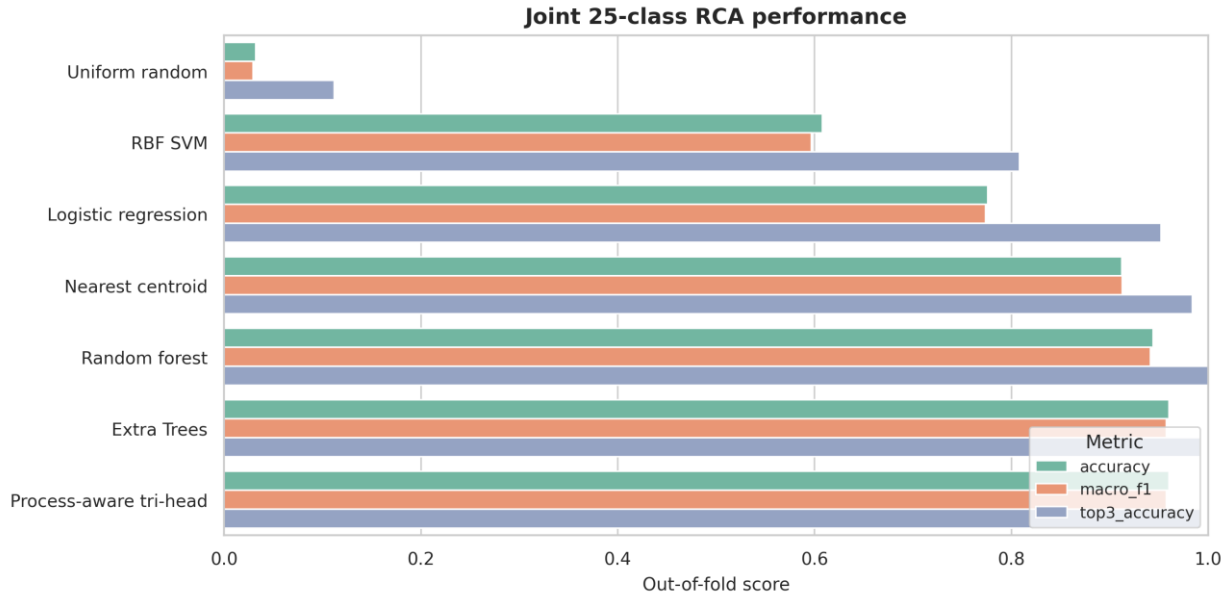

Figure 4. Out-of-fold comparison for joint 25-class RCA

The tree scores require caution: every joint class has only five repetitions from one laboratory deployment. They establish separability inside RE1-OB, not resilience to new services, workloads, or fault mechanisms. Nearest centroid's 0.912 accuracy also shows that many classes occupy distinct contrastive regions.

Decomposed results clarify where the remaining error occurred. Table 7 reports localization and identification for every model. The process-aware model reached 0.968 service accuracy and macro-F1 0.968, while fault accuracy was 0.992 with macro-F1 0.992. Joint accuracy was lower because both decisions had to be correct. Figure 5 displays the two confusion matrices. Four of five joint errors assigned a loss incident to the wrong root service; one adservice CPU incident was identified as disk while preserving the service. The fault head was therefore nearly perfect, but propagation patterns under loss made the initiating service less distinct.

Table 7. Decomposed localization, identification, and joint performance

Model	Svc Acc.	Svc F1	Fault Acc.	Fault F1	Joint Acc.	Joint F1
Uniform random	0.192	0.191	0.248	0.249	0.032	0.030
Nearest centroid	0.936	0.936	0.960	0.960	0.912	0.912
Logistic regression	0.832	0.833	0.920	0.921	0.776	0.774
RBF SVM	0.728	0.731	0.744	0.742	0.608	0.597
Random forest	0.968	0.968	0.976	0.976	0.944	0.941
Extra Trees	0.968	0.968	0.992	0.992	0.960	0.957
Process-aware tri-head	0.968	0.968	0.992	0.992	0.960	0.957

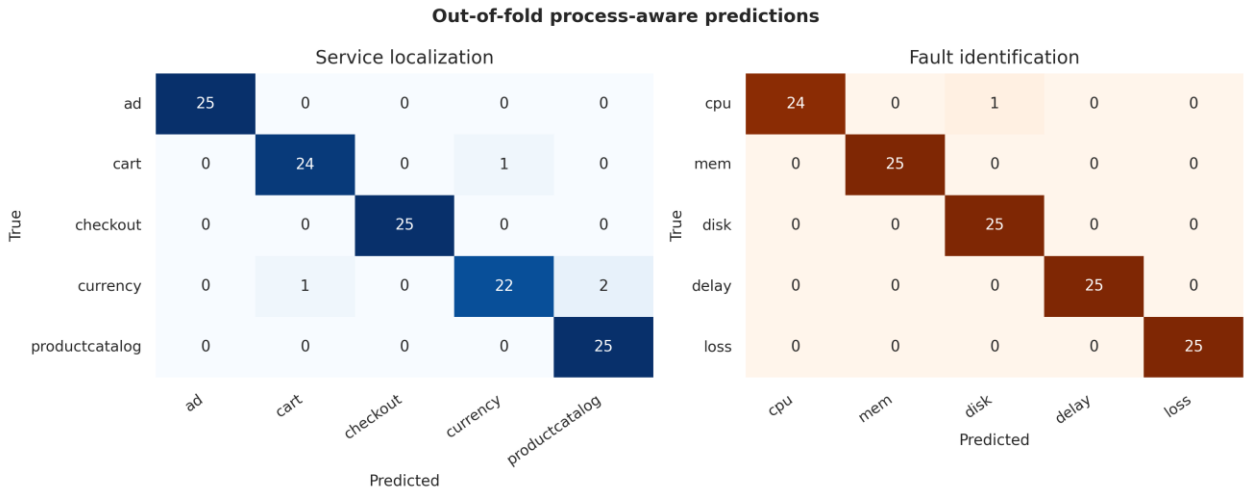

Figure 5. Service and fault confusion matrices for the process-aware tri-head

Table 8 breaks the process-aware results down by fault and service. Memory, disk, and delay achieved 1.000 joint accuracy; CPU reached 0.960; and loss reached 0.840. At the service level, checkoutservice and productcatalogservice reached 1.000, adservice and cartservice reached 0.960, and currencyservice reached 0.880. All three currencyservice errors were loss cases, and the fourth loss localization error originated in cartservice. This concentration shows why a balanced macro score alone is insufficient: the agent remained vulnerable to one propagation regime even though the overall fault classifier was strong.

Table 8. Process-aware performance by injected fault and root service

Grouping	Level	n	Svc Acc.	Fault Acc.	Joint Acc.
Fault	CPU	25	1.000	0.960	0.960
Fault	MEM	25	1.000	1.000	1.000
Fault	DISK	25	1.000	1.000	1.000
Fault	DELAY	25	1.000	1.000	1.000
Fault	LOSS	25	0.840	1.000	0.840
Service	adservice	25	1.000	0.960	0.960
Service	cartservice	25	0.960	1.000	0.960
Service	checkoutservice	25	1.000	1.000	1.000
Service	currencyservice	25	0.880	1.000	0.880
Service	productcatalogservice	25	1.000	1.000	1.000

Evidence policy changed audit quality without changing labels. Table 9 and Figure 6 show the results. Outcome only scored ETS 0.653 with no support. Post-only volatility reached Recall@5 0.504, nDCG 0.118, reason 0.185, and ETS 0.715. Injection-aligned contrast reached 0.952, 0.555, 0.653, and 0.871, respectively. Reranking reached 1.000 recall, 0.644 nDCG, 0.984 claim support, reason 0.719, and ETS 0.893.

Table 9. Evidence-policy comparison on all 125 held-out cases

Policy	P@5	Recall@5	nDCG@5	Claim support	Reason	ETS
Outcome only	0.000	0.000	0.000	0.000	0.000	0.653
Post-only volatility	0.120	0.504	0.118	0.512	0.185	0.715
Injection-aligned contrast	0.450	0.952	0.555	0.928	0.653	0.871
Process-aware reranking	0.499	1.000	0.644	0.984	0.719	0.893

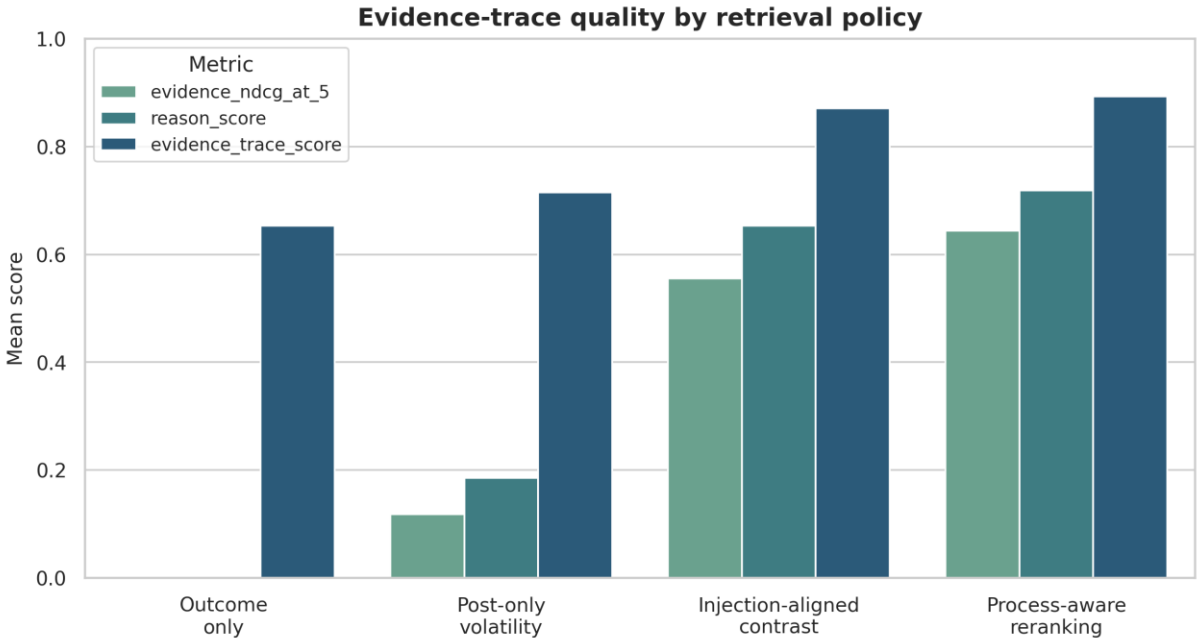

Figure 6. Evidence-trace quality under four retrieval policies

The difference between contrast and reranking was not cosmetic. Contrast identified strong changes wherever they occurred, including downstream services. Reranking connected those changes to the agent's explicit claim, moving predicted-service evidence earlier in the list. When the service prediction was correct, this also moved true root-service evidence upward. When it was wrong, claim support remained high but service correctness and true-service nDCG fell, limiting ETS. The scoring rule therefore distinguished internally consistent error from grounded success. It did not treat agreement with a wrong hypothesis as causal proof.

Feature sensitivity in Table 10 reveals a second caution. Baseline-only profiles achieved 0.624 accuracy—far above the 0.040 theoretical chance level for 25 balanced classes—showing that operating conditions and collection batches carried residual structure even after explicit timestamps were removed. Post-incident profiles reached 0.968, while common pre/post contrasts reached 0.952 in the sensitivity run. CPU and memory contrasts alone reached 0.920; workload contrasts alone reached 0.472. Union-schema contrasts reached 0.976, with or without availability indicators. The union result is reported as sensitivity rather than the primary estimate because latency and error schemas vary across collection groups and can reveal fault era. The shared 35-channel contrast is the more conservative basis for the paper's main claims.

Table 10. Feature and schema sensitivity analysis

Configuration	Accuracy	Macro-F1	Top-3	Log loss
Pre-incident profile	0.624	0.605	0.904	1.046
Post-incident profile	0.968	0.968	0.984	0.258
CPU and memory contrast	0.920	0.919	1.000	0.523
Workload contrast	0.472	0.432	0.752	1.808
All common contrasts	0.952	0.949	0.992	0.501
Union contrasts	0.976	0.976	1.000	0.251
Union + availability	0.976	0.976	1.000	0.273

Uncertainty, calibration, selective diagnosis, and paired comparisons appear in Table 11. The process-aware model was significantly more accurate than logistic regression (exact $p=2.38\times10-7$) and the RBF SVM ($p=1.34\times10-12$), but not random forest ($p=0.500$) or Extra Trees ($p=1.000$). The last comparison is equal by design because the tri-head combination did not change the joint head's top-one decisions. Mean confidence was 0.598 despite 0.960 empirical accuracy, producing 10-bin

expected calibration error 0.362 and multiclass Brier score 0.242. The model was strongly underconfident, so raw probabilities should not be treated as incident risk estimates.

Table 11. Uncertainty, calibration, selective performance, and paired tests

Analysis	Setting	Estimate	Interval / contrast	Details
Bootstrap	accuracy	0.960	95% CI [0.920, 0.992]	5,000 case resamples
Bootstrap	macro_f1	0.957	95% CI [0.914, 0.987]	5,000 case resamples
Calibration	10-bin ECE	0.362	Mean confidence 0.598	Empirical accuracy 0.960
Calibration	Multiclass Brier	0.242	Lower is better	25 classes
Selective	Coverage 0.800	0.990	Risk 0.010	n=100
Selective	Coverage 0.600	1.000	Risk 0.000	n=75
McNemar	Process-aware vs Logistic regression	p=2.38e-07	discordant 23:0	Two-sided exact
McNemar	Process-aware vs RBF SVM	p=1.34e-12	discordant 45:1	Two-sided exact
McNemar	Process-aware vs Random forest	p=0.5	discordant 2:0	Two-sided exact
McNemar	Process-aware vs Extra Trees	p=1	discordant 0:0	Two-sided exact

Evidence-aware selection improved accepted-case performance, as Figure 7 shows. Ranking cases by 70% posterior confidence and 30% reason score produced 0.990 accuracy at 80% coverage and 1.000 at 60% coverage. This result describes a retrospective selective policy over the out-of-fold cases. In deployment, a coverage threshold requires prospective calibration on later incidents. The current evidence score remains useful for routing: low-confidence or weakly supported diagnoses can be escalated to an engineer without discarding the metric trail already collected (Xin, 2026).

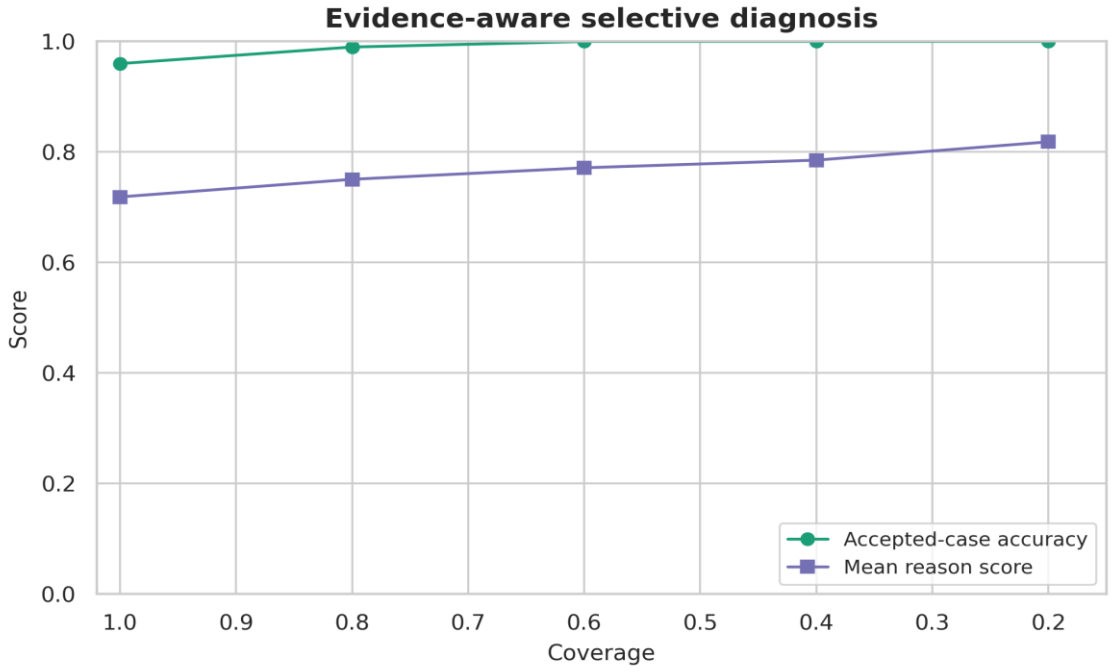

Figure 7. Evidence-aware selective diagnosis as coverage decreases

The results support process-aware agent design in three ways. First, a constrained scaffold made every scored claim traceable to metric names, windows, and recomputable statistics. Second, evidence ranking exposed a failure mode that top-one accuracy concealed: loss faults were identified correctly, yet their originating service was sometimes obscured by propagation. Third, the framework separated language realization from diagnosis. An LLM can summarize the retained record, ask for additional tools, or translate it into an incident report (Zhang et al., 2026), but it does not need authority to alter the underlying measurements. This division directly addresses the concern that fluent reasoning text can appear persuasive without being grounded.

Interpretation is bounded by metric-only telemetry, an oracle injection-time anchor, five repetitions per class, and detectable collection structure. Root-service membership—not an expert causal chain—defined evidence relevance, and disk evidence was indirect. The experiment evaluated the scaffold and record rather than named generative LLMs. Its claims therefore concern a metric-grounded layer for an agent, not language-generation performance.

5. Conclusions

This study evaluated a process-aware RCA scaffold on all 125 RE1-OB incidents with case-level held-out-repetition testing. Equal 300-second baseline and post-injection windows, common-channel predictors, and exclusion of acquisition metadata reduced direct schema leakage. The process-aware tri-head achieved 0.960 joint accuracy, 0.957 macro-F1, 0.968 service accuracy, 0.992 fault accuracy, and 0.992 top-three accuracy. Its 95% bootstrap accuracy interval was [0.920, 0.992]. These values describe controlled Online Boutique fault injections, not unrestricted production incidents.

Evidence scoring added information that outcome accuracy could not supply. Injection-aligned contrast raised ETS from 0.715 for post-only volatility to 0.871, and claim-aware reranking raised it to 0.893 while placing true-service evidence within the top five for every case. The error analysis still identified a concrete weakness: four loss incidents were assigned to the wrong initiating service. High aggregate performance therefore coexisted with a localized propagation problem and poor probability calibration.

The principal design implication is to keep diagnosis, evidence verification, and language realization as distinct layers. A language-model agent can control tools and communicate findings, while deterministic telemetry functions retain the measurements that support each claim. Future evaluation should apply the same scoring to RCA100 or AIOps2025 when their logs, spans, and expert evidence chains are part of the experimental corpus, and should test temporal transfer across new workloads and service versions. Within the metric-only setting examined here, process-aware retrieval produced a more auditable diagnosis without hiding incorrect conclusions behind fluent explanations.

Funding: This research received no external funding
Conflicts of Interest: The authors declare no conflict of interest
Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers
Artificial Intelligence (AI) Use Disclosure: The authors declare that no artificial intelligence tools were used in the preparation of this manuscript

References

[1] Cai, Y., Nie, X., Yin, K., Pei, C., Sun, Y., Zhang, S., Liu, H., Liu, G., Wen, X., Situ, F., & Pei, D. (2026). A multi-dataset benchmark for evaluating LLM agents in microservice failure diagnosis [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2606.29193

[2] Chen, Y., Shetty, M., Somashekar, G., Ma, M., Simmhan, Y., Mace, J., Bansal, C., Wang, R., & Rajmohan, S. (2025). AIOpsLab: A holistic framework to evaluate AI agents for enabling autonomous clouds. Proceedings of Machine Learning and Systems, 7. https://proceedings.mlsys.org/paper_files/paper/2025/hash/d1f9e4a9f109b6e8b75ed362736f22ec-Abstract-Conference.html

[3] Chen, Y., Xie, H., Ma, M., Kang, Y., Gao, X., Shi, L., Cao, Y., Gao, X., Fan, H., Wen, M., Zeng, J., Ghosh, S., Zhang, X., Zhang, C., Lin, Q., Rajmohan, S., Zhang, D., & Xu, T. (2024). Automatic root cause analysis via large language models for cloud incidents. In Proceedings of the Nineteenth European Conference on Computer Systems (pp. 674–688). Association for Computing Machinery. https://doi.org/10.1145/3627703.3629553

[4] Clark, J., Su, Y., Pial, S. M. R., Tian, Y., Gniedziejko, L., Jacobsen, H.-A., Chen, Y., & Xu, T. (2026). SREGym: A live benchmark for AI SRE agents with high-fidelity failure scenarios [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2605.07161

[5] Fang, A., Yang, Y., Shang, J., Lu, Q., Xu, J., Wang, R., Zhang, S., Zhang, Y., Yu, B., & He, P. (2026). OpenRCA 2.0: From outcome labels to causal process supervision [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2606.27154

[6] Ikram, A., Chakraborty, S., Mitra, S., Saini, S., Bagchi, S., & Kocaoglu, M. (2022). Root cause analysis of failures in microservices through causal discovery. Advances in Neural Information Processing Systems, 35, 31158–31170. https://proceedings.neurips.cc/paper_files/paper/2022/hash/c9fcd02e6445c7dfbad6986abee53d0d-Abstract-Conference.html

[7] Jha, S., Arora, R. R., Watanabe, Y., Yanagawa, T., Chen, Y., Clark, J., Bhavya, B., Verma, M., Kumar, H., Kitahara, H., Zheutlin, N., Takano, S., Pathak, D., George, F., Wu, X., Turkkan, B. O., Vanloo, G., Nidd, M., Dai, T., . . . Puri, R. (2025). ITBench: Evaluating AI agents across diverse real-world IT automation tasks. Proceedings of Machine Learning Research, 267, 27134–27197. https://proceedings.mlr.press/v267/jha25a.html

[8] Jin, J. (2025). Evidence-chain reliable RAG: Hallucination detection, source attribution, and deterministic provenance explanations. Journal of Technology Informatics and Engineering, 4(2), 520–533. https://doi.org/10.51903/jtie.v4i2.535

[9] Lee, C., Yang, T., Chen, Z., Su, Y., & Lyu, M. R. (2023). Eadro: An end-to-end troubleshooting framework for microservices on multi-source data. In 2023 IEEE/ACM 45th International Conference on Software Engineering (pp. 1750–1762). IEEE. https://doi.org/10.1109/ICSE48619.2023.00150

[10] Li, Z., Zhao, N., Zhang, S., Sun, Y., Chen, P., Wen, X., Ma, M., & Pei, D. (2022). Constructing large-scale real-world benchmark datasets for AIOps [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2208.03938

[11] Lin, J., Chen, P., & Zheng, Z. (2018). Microscope: Pinpoint performance issues with causal graphs in micro-service environments. In Service-Oriented Computing (pp. 3–20). Springer. https://doi.org/10.1007/978-3-030-03596-9_1

[12] Liu, Y., Pei, C., Xu, L., Chen, B., Sun, M., Zhang, Z., Sun, Y., Zhang, S., Wang, K., Zhang, H., Li, J., Xie, G., Wen, X., Nie, X., Ma, M., & Pei, D. (2023). OpsEval: A comprehensive IT operations benchmark suite for large language models [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2310.07637

[13] Naakka, A., Wang, Y., & Mäntylä, M. V. (2026). LATS-RCA: Language agent tree search for root cause analysis in microservices [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2605.03505

[14] Pei, C., Wang, Z., Liu, F., Li, Z., Liu, Y., He, X., Kang, R., Zhang, T., Chen, J., Li, J., Xie, G., & Pei, D. (2025). Flow-of-Action: SOP enhanced LLM-based multi-agent system for root cause analysis. In Companion Proceedings of the ACM Web Conference 2025 (pp. 422–431). Association for Computing Machinery. https://doi.org/10.1145/3701716.3715225

[15] Pham, L. (2025). RCAEval: A benchmark for root cause analysis of microservice systems [Data set]. Zenodo. https://doi.org/10.5281/zenodo.14590730

[16] Pham, L., Ha, H., & Zhang, H. (2024). BARO: Robust root cause analysis for microservices via multivariate Bayesian online change point detection. Proceedings of the ACM on Software Engineering, 1(FSE), 2214–2237. https://doi.org/10.1145/3660805

[17] Pham, L., Zhang, H., Ha, H., Salim, F., & Zhang, X. (2025). RCAEval: A benchmark for root cause analysis of microservice systems with telemetry data. In Companion Proceedings of the ACM Web Conference 2025 (pp. 777–780). Association for Computing Machinery. https://doi.org/10.1145/3701716.3715290

[18] Qin, Y., Liang, S., Ye, Y., Zhu, K., Yan, L., Lu, Y., Lin, Y., Cong, X., Tang, X., Qian, B., Zhao, S., Hong, L., Tian, R., Xie, R., Zhou, J., Gerstein, M., Li, D., Liu, Z., & Sun, M. (2024). ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In The Twelfth International Conference on Learning Representations. https://openreview.net/forum?id=dHng2O0Jjr

[19] Roy, D., Zhang, X., Bhave, R., Bansal, C., Las-Casas, P., Fonseca, R., & Rajmohan, S. (2024). Exploring LLM-based agents for root cause analysis. In Companion Proceedings of the 32nd ACM International Conference on the Foundations of Software Engineering (pp. 208–219). Association for Computing Machinery. https://doi.org/10.1145/3663529.3663841

[20] Wang, Z., Liu, Z., Zhang, Y., Zhong, A., Wang, J., Yin, F., Fan, L., Wu, L., & Wen, Q. (2024). RCAgent: Cloud root cause analysis by autonomous agents with tool-augmented large language models. In Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (pp. 4966–4974). Association for Computing Machinery. https://doi.org/10.1145/3627673.3680016

[21] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q. V., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems, 35, 24824–24837. https://proceedings.neurips.cc/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html

[22] Wu, L., Tordsson, J., Elmroth, E., & Kao, O. (2020). MicroRCA: Root cause localization of performance issues in microservices. In NOMS 2020—2020 IEEE/IFIP Network Operations and Management Symposium (pp. 1–9). IEEE. https://doi.org/10.1109/NOMS47738.2020.9110353

[23] Xin, Q. (2025). Explaining OpenStack failure-injection log anomalies with retrieved normal prototypes. Emerging Information Science and Technology, 6(2), 125–146. https://doi.org/10.18196/eist.v6i2.31232

[24] Xin, Q. (2026). Log anomaly detection with conformal alert control and evidence-grounded incident ticket generation. Aviation Electronics, Information Technology, Telecommunications, Electricals, and Controls (AVITEC), 8(2), 247–264. https://doi.org/10.28989/avitec.v8i2.3974

[25] Xu, J., Zhang, Q., Zhong, Z., He, S., Zhang, C., Lin, Q., Pei, D., He, P., Zhang, D., & Zhang, Q. (2025). OpenRCA: Can large language models locate the root cause of software failures? In The Thirteenth International Conference on Learning Representations. https://openreview.net/forum?id=M4qNIzQYpd

[26] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. In The Eleventh International Conference on Learning Representations. https://openreview.net/forum?id=WE_vluYUL-X

[27] Yu, G., Chen, P., Chen, H., Guan, Z., Huang, Z., Jing, L., Weng, T., Sun, X., & Li, X. (2021). MicroRank: End-to-end latency issue localization with extended spectrum analysis in microservice environments. In Proceedings of the Web Conference 2021 (pp. 3087–3098). Association for Computing Machinery. https://doi.org/10.1145/3442381.3449905

[28] Yu, G., Chen, P., Li, Y., Chen, H., Li, X., & Zheng, Z. (2023). Nezha: Interpretable fine-grained root causes analysis for microservices on multi-modal observability data. In Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (pp. 553–565). Association for Computing Machinery. https://doi.org/10.1145/3611643.3616249

[29] Zhang, B., Sun, X., Liu, G., & Zhou, B. (2026). LLM-style DevOps copilot for cloud-native troubleshooting: Retrieval-augmented runbook generation and command-safety evaluation. Journal of Technology Informatics and Engineering, 5(2), 104–118. https://doi.org/10.51903/jtie.v5i2.534

[30] Zhang, R., Wen, Z., Wang, C., Tang, C., Xu, P., & Jiang, Y. (2025). Quality analysis and evaluation prediction of RAG retrieval based on machine learning algorithms [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2511.19481

[31] Zhang, S., Jin, P., Lin, Z., Sun, Y., Zhang, B., Xia, S., Li, Z., Zhong, Z., Ma, M., Jin, W., Zhang, D., Zhu, Z., & Pei, D. (2023). Robust failure diagnosis of microservice system through multimodal data. IEEE Transactions on Services Computing, 16(6), 3851–3864. https://doi.org/10.1109/TSC.2023.3290018

[32] Zhang, S., Xia, S., Fan, W., Shi, B., Xiong, X., Zhong, Z., Ma, M., Sun, Y., & Pei, D. (2025). Failure diagnosis in microservice systems: A comprehensive survey and analysis. ACM Transactions on Software Engineering and Methodology. https://doi.org/10.1145/3715005

[33] Zhang, W., Guo, H., Yang, J., Tian, Z., Zhang, Y., Chaoran, Y., Li, Z., Li, T., Shi, X., Zheng, L., & Zhang, B. (2024). mABC: Multi-agent blockchain-inspired collaboration for root cause analysis in micro-services architecture. In Findings of the Association for Computational Linguistics: EMNLP 2024 (pp. 4017–4033). Association for Computational Linguistics. https://doi.org/10.18653/v1/2024.findings-emnlp.232

[34] Zhang, X., Wang, Q., Li, M., Yuan, Y., Xiao, M., Zhuang, F., & Yu, D. (2025). TAMO: Fine-grained root cause analysis via tool-assisted LLM agent with multi-modality observation data in cloud-native systems [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2504.20462

[35] Zhong, Z. S., Li, C., & Rao, H. (2026). Trajectory reliability prediction for generalist AI agents: Tool-use failure analysis and success forecasting on ZClawBench. Journal of Technology Informatics and Engineering, 5(1), 341–360. https://doi.org/10.51903/jtie.v5i1.539