European Journal of Computer Science and Information Technology, 13(42),147-155, 2025
Print ISSN: 2054-0957 (Print)
Online ISSN: 2054-0965 (Online)
Website: https://www.eajournals.org/
Publication of the European Centre for Research Training and Development -UK

A Framework for Self-Healing Enterprise Applications Using Observability and Generative Intelligence

Goutham Yenuganti
Independent Researcher, USA

Citation: Yenuganti G. (2025) A Framework for Self-Healing Enterprise Applications Using Observability and Generative Intelligence, *European Journal of Computer Science and Information Technology*, 13(42),147-155, https://doi.org/10.37745/ejcsit.2013/vol13n42147155

Abstract: *Enterprise applications operating in distributed cloud environments face significant reliability challenges that traditional monitoring systems cannot adequately address. This framework presents a novel solution combining structured observability with generative artificial intelligence to create autonomous self-healing capabilities. The proposed system integrates multi-layered architecture encompassing telemetry collection, intelligent processing, and controlled execution layers. Generative intelligence models serve as reasoning engines that interpret system anomalies and synthesize appropriate remediation strategies within carefully defined safety boundaries. The framework implements hierarchical anomaly detection methodologies that minimize false positives while maintaining sensitivity to genuine system issues. Automated remediation workflows incorporate risk assessment logic and human-in-the-loop approval processes for complex scenarios. Multiple safety mechanisms including circuit breakers, canary deployments, and automatic rollback triggers ensure system integrity during autonomous operations. The framework transforms enterprise application reliability from reactive incident response to proactive self-maintenance, significantly reducing mean time to recovery while minimizing operational burden on engineering teams.*

Keywords: Self-healing systems, observability, generative intelligence, autonomous remediation, enterprise applications

INTRODUCTION

Modern enterprise applications face unprecedented complexity in distributed cloud environments, where traditional reactive monitoring approaches fall short of ensuring optimal system reliability. Organizations today experience significant financial impact from system downtime, with costs varying dramatically based on industry sector and application criticality [1]. The convergence of advanced observability platforms and generative artificial intelligence presents a transformative opportunity to evolve beyond passive monitoring

toward active, autonomous system repair. AI-based insights are fundamentally transforming observability by enabling predictive analytics, automated pattern recognition, and intelligent correlation of system events that were previously impossible with traditional monitoring approaches [2].

This article presents a comprehensive framework for implementing self-healing enterprise applications that leverage structured observability data to feed generative intelligence models capable of detecting anomalies, synthesizing corrective actions, and executing repairs within carefully defined control boundaries. The proposed framework addresses the fundamental challenge of reducing mean time to recovery while minimizing human intervention in routine system failures. By establishing clear rollback mechanisms and safety controls, organizations can build trust in autonomous remediation systems that not only detect issues faster than human operators but can also implement fixes with precision and consistency. This paradigm shift represents a movement from reactive incident response to proactive system self-maintenance, fundamentally redefining enterprise application reliability strategies.

Architectural Foundation and Core Components

The self-healing framework operates on a multi-layered architecture that separates concerns while maintaining tight integration between observability, intelligence, and execution layers. This architectural approach draws from established principles in distributed systems design, where hierarchical processing structures enable scalable and maintainable solutions for complex computational problems [3]. The foundational layer consists of comprehensive telemetry collection systems that capture metrics, logs, traces, and custom business indicators across all application components. This telemetry substrate must provide high-fidelity, real-time data streams with consistent formatting and semantic enrichment to enable effective pattern recognition across distributed microservice architectures.

The intelligence layer processes this observability data through a combination of statistical analysis engines, machine learning models, and generative AI systems. Statistical engines handle baseline establishment and threshold-based alerting, implementing sophisticated algorithms that can adapt to changing system behaviors and seasonal patterns. Machine learning models focus on anomaly detection and pattern classification, utilizing advanced techniques that can identify subtle correlations and emerging issues before they manifest as user-visible problems. The generative intelligence component serves as the reasoning engine, interpreting detected anomalies within business context and synthesizing appropriate remediation strategies based on historical successful interventions and organizational best practices.

The execution layer implements a controlled automation framework with multiple safety mechanisms including approval gates, rollback triggers, and blast radius limitations. This layer maintains a comprehensive library of validated remediation patterns, each with associated risk profiles, success metrics, and contextual applicability criteria. Modern microservice telemetry data provides rich insights into system dynamics, enabling sophisticated analysis of service interactions, dependency relationships, and cascading failure patterns that inform remediation strategy selection [4]. The integration between these layers occurs through well-defined APIs that maintain comprehensive audit trails and enable human oversight at every

decision point, ensuring transparency and accountability in autonomous operations while preserving the ability for manual intervention when necessary.

Observability Data Architecture and Telemetry Strategy

Effective self-healing capabilities require observability data that extends beyond traditional infrastructure metrics to include application-specific indicators, user experience signals, and business process health markers. The telemetry strategy must establish consistent data models that enable correlation across different system layers and time horizons, implementing standardized approaches that facilitate effective monitoring and analysis. OpenTelemetry best practices emphasize the importance of structured data collection, semantic conventions, and consistent instrumentation patterns that enable comprehensive system visibility while minimizing performance overhead and operational complexity [5]. This includes implementing semantic labeling systems that allow the intelligence layer to understand the business impact of technical anomalies, translating low-level system events into meaningful business context.

The data architecture employs a multi-tier storage strategy optimized for both real-time processing and historical analysis, balancing performance requirements with cost considerations and retention policies. Hot tier storage maintains recent high-resolution data for immediate anomaly detection, utilizing high-performance storage systems that can support sub-second query responses for real-time alerting and automated remediation triggers. Warm and cold tiers preserve longer-term patterns essential for baseline establishment and trend analysis, implementing compression and aggregation strategies that maintain analytical value while optimizing storage costs. The system implements comprehensive data quality gates that validate telemetry completeness and accuracy, ensuring that automated decisions are based on reliable information through sophisticated validation algorithms and anomaly detection at the data ingestion layer. Stream processing frameworks handle real-time data transformation and enrichment, adding contextual metadata that enables more sophisticated analysis and correlation across distributed system components. Correlation identifiers play a crucial role in linking related events across distributed system components, enabling trace reconstruction and impact analysis that spans multiple services and infrastructure layers [6]. This includes severity classifications based on historical impact analysis, automated tagging systems that categorize events based on business impact, and contextual enrichment that helps prioritize remediation efforts based on organizational objectives and service level agreement requirements.

European Journal of Computer Science and Information Technology, 13(42),147-155, 2025
Print ISSN: 2054-0957 (Print)
Online ISSN: 2054-0965 (Online)
Website: https://www.eajournals.org/
Publication of the European Centre for Research Training and Development -UK

Table 1: Observability Data Architecture Components [5, 6]

Storage Tier	Data Retention	Query Performance	Use Case	Storage Optimization
Hot Tier	Real-time to recent data	Sub-second response	Immediate anomaly detection	High-performance SSD arrays
Warm Tier	Medium-term historical	Second-level response	Baseline establishment	Compressed storage with indexing
Cold Tier	Long-term archives	Minute-level response	Trend analysis	Highly compressed with aggregation
Stream Processing	Continuous flow	Real-time transformation	Event correlation	In-memory processing clusters

Generative Intelligence Integration and Model Architecture

The integration of generative intelligence into the self-healing framework requires careful model selection and training strategies that balance reasoning capability with response latency requirements and operational constraints. Performance evaluation of large language models demonstrates significant variability in their effectiveness across different operational contexts, with specialized fine-tuning and domain adaptation proving essential for reliable performance in production environments [7]. Large language models serve as the primary reasoning engine, interpreting structured observability data alongside unstructured log entries and error messages to understand system state and identify root causes through sophisticated natural language processing and pattern matching capabilities.

These models undergo domain-specific fine-tuning using historical incident data, successful remediation patterns, and organizational operational procedures, creating specialized knowledge bases that reflect organizational context and established best practices. The training process incorporates feedback loops that continuously improve model performance based on real-world outcomes and operator feedback, implementing reinforcement learning techniques that reward successful remediation strategies while penalizing approaches that lead to system instability or user impact.

The model architecture implements a multi-agent approach where specialized models handle different aspects of the healing process, leveraging recent advances in multi-agent systems and domain-specific language model applications [8]. Diagnostic agents focus on root cause analysis and impact assessment, utilizing specialized training data that emphasizes causal reasoning and system behavior analysis.

Remediation agents generate specific corrective actions based on established operational procedures, incorporating safety constraints and organizational policies into their decision-making processes. Validation agents review proposed actions against safety criteria and historical success rates before execution approval, implementing multi-layer validation that considers technical feasibility, business impact, and risk assessment.

To ensure reliability and consistency, the framework employs model ensembling techniques that combine outputs from multiple models and validation layers, implementing consensus mechanisms that improve overall accuracy and reduce the risk of inappropriate automated actions. This approach reduces the risk of hallucinated or inappropriate remediation suggestions while maintaining the creative problem-solving capabilities that make generative intelligence valuable for handling novel failure scenarios and edge cases that haven't been encountered in historical data.

Table 2: Generative Intelligence Model Architecture [7, 8]

Agent Type	Primary Function	Specialization	Validation Method	Performance Metric
Diagnostic Agent	Root cause analysis	System behavior analysis	Historical pattern matching	Accuracy in cause identification
Remediation Agent	Solution generation	Operational procedures	Safety constraint validation	Success rate of generated solutions
Validation Agent	Action approval	Risk assessment	Multi-layer consensus	Rejection rate of unsafe actions
Ensemble Coordinator	Decision integration	Consensus building	Cross-validation	Overall system reliability

Anomaly Detection and Automated Remediation Workflows

The anomaly detection system operates through a hierarchical approach that combines multiple detection methodologies to minimize false positives while maintaining sensitivity to genuine issues that could impact system reliability or user experience. This multi-layered approach draws from established research in anomaly detection that demonstrates improved performance through ensemble methods and hierarchical classification structures [9]. Statistical methods handle well-understood metrics with established baselines, implementing adaptive thresholding algorithms that can accommodate normal variations in system behavior while detecting genuine anomalies that indicate potential problems.

Machine learning models identify complex patterns and correlations that indicate emerging problems, utilizing advanced techniques including time series analysis, clustering algorithms, and deep learning approaches that can detect subtle patterns in high-dimensional telemetry data. These models are continuously trained on streaming data to adapt to changing system behaviors and emerging failure patterns, implementing online learning techniques that maintain model accuracy in dynamic environments. The generative intelligence layer provides contextual interpretation that distinguishes between benign anomalies and those requiring immediate intervention, applying sophisticated reasoning that considers business context, historical patterns, and organizational priorities.

Automated remediation workflows follow a structured decision tree that evaluates detected anomalies against a comprehensive library of known patterns and approved response procedures, implementing sophisticated matching algorithms that consider symptom similarity, environmental context, and historical success rates. Each workflow includes comprehensive risk assessment logic that considers multiple factors including system criticality, user impact potential, remediation complexity, and organizational risk tolerance before determining the appropriate response level. Machine learning-based health monitoring systems provide the foundation for these risk assessments, utilizing predictive models that can estimate the probability and potential impact of different remediation approaches [10].

Low-risk, high-confidence scenarios trigger immediate automated remediation through well-tested procedures that have demonstrated consistent success rates, while higher-risk situations initiate human-in-the-loop approval processes that balance automation benefits with safety requirements. The workflow execution engine maintains detailed audit trails that capture decision rationale, actions taken, and outcomes achieved, implementing comprehensive logging that supports both operational analysis and compliance requirements. This historical data feeds back into the learning system to improve future decision-making and expand the library of automated remediation patterns through continuous improvement processes.

Control Boundaries, Safety Mechanisms, and Rollback Strategies

Establishing robust control boundaries is essential for maintaining system safety while enabling autonomous operation, requiring comprehensive safety frameworks that can prevent automated systems from causing additional harm while pursuing remediation objectives. The framework implements multiple layers of safety controls including pre-execution validation, runtime monitoring, and post-execution verification, drawing from established safety engineering principles that emphasize defense-in-depth approaches and fail-safe design patterns [11]. Pre-execution validation checks proposed actions against organizational policies, system constraints, and historical risk assessments, implementing rule-based systems that can quickly identify potentially dangerous or inappropriate actions before they are executed.

Runtime monitoring tracks execution progress and system response indicators to detect unexpected behaviors or adverse impacts during remediation attempts, utilizing real-time analytics that can identify when automated actions are not producing expected results or are causing unintended side effects. This monitoring capability implements sophisticated pattern recognition that can distinguish between normal

remediation effects and problematic responses that indicate the need for immediate intervention or rollback procedures.

Safety mechanisms include circuit breakers that halt automated remediation when error rates exceed acceptable thresholds, implementing adaptive algorithms that can adjust sensitivity based on system criticality and operational context. Canary deployment patterns limit the scope of changes to minimize potential impact, gradually expanding remediation actions based on observed success indicators and safety metrics. Automatic rollback triggers are activated by degraded performance metrics, implementing comprehensive monitoring that can detect various forms of system degradation and trigger appropriate recovery procedures.

The system maintains multiple rollback strategies ranging from simple configuration reversions to complex multi-step restoration procedures, each tailored to specific types of interventions and system architectures. These strategies incorporate lessons learned from safety engineering research that emphasizes the importance of tested recovery procedures and clear rollback criteria [12]. Control boundaries also encompass human oversight requirements, with escalation procedures that engage on-call engineers when automated remediation attempts fail or when anomalies fall outside established parameters. The framework provides comprehensive dashboards and alerting systems that keep human operators informed of autonomous activities while maintaining the ability to intervene or override automated decisions when necessary, ensuring that human expertise remains available for complex or unprecedented situations.

Table 3: Safety Mechanisms and Control Boundaries [11, 12]

Safety Mechanism	Trigger Condition	Response Action	Rollback Strategy	Escalation Path
Circuit Breaker	Error rate threshold exceeded	Halt automated remediation	Immediate process suspension	Alert operations team
Canary Deployment	Gradual change validation	Limit scope of modifications	Incremental rollback	Progressive escalation
Runtime Monitor	Unexpected system behavior	Real-time intervention	Context-specific reversion	Automated notification
Pre-execution Validator	Policy violation detected	Block action execution	No changes applied	Compliance team review

CONCLUSION

The framework for self-healing enterprise applications fundamentally transforms organizational approaches to system reliability and operational efficiency through the strategic integration of structured observability and generative intelligence within controlled automation boundaries. Organizations implementing this framework achieve substantial reductions in mean time to recovery while simultaneously decreasing operational burden on engineering teams. Success requires organizational commitment to data quality standards, comprehensive model training protocols, and robust safety procedure development, yet the resulting improvements in system reliability and operational efficiency demonstrate clear return on investment. This self-healing paradigm will become increasingly essential as system complexity continues expanding and user availability expectations intensify. Organizations successfully deploying autonomous remediation capabilities gain competitive advantages through enhanced operational efficiency, superior customer satisfaction, and improved engineering productivity. The framework's emphasis on comprehensive safety controls and human oversight mechanisms enables gradual, secure transition to autonomous systems while preserving intervention capabilities for exceptional circumstances. These developments extend beyond individual organizational benefits to influence industry standards, vendor product offerings, and operational best practices across the enterprise technology landscape. As self-healing capabilities gain widespread adoption, standardization will emerge in observability data formats, remediation pattern libraries, and safety control mechanisms. This evolution redefines platform engineering roles from reactive problem-solving to proactive system architecture focused on designing resilient, self-maintaining infrastructure capable of autonomous adaptation and healing.

REFERENCES

1. Unitrends "Downtime: Causes, Costs and How to Minimize It," Unitrends, 2021. [Online]. Available: https://www.unitrends.com/blog/downtime-causes-costs-and-how-to-minimize-it/
2. Joydip Kanjilal, "How AI-Based Insights Can Transform Observability," DevOps, 2025. [Online]. Available: https://devops.com/how-ai-based-insights-can-transform-observability/
3. Marco Bertini, et al. "Multi-scale and real-time non-parametric approach for anomaly detection and localization," Computer Vision and Image Understanding, 2012. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S1077314211002104
4. Abdullah Al Maruf, et al., "Using Microservice Telemetry Data for System Dynamic Analysis," ResearchGate, 2022. [Online]. Available: https://www.researchgate.net/publication/364544943_Using_Microservice_Telemetry_Data_for_System_Dynamic_Analysis
5. Ayooluwa Isaiah, "Essential OpenTelemetry Best Practices for Robust Observability," Better Stack, 2025. [Online]. Available: https://betterstack.com/community/guides/observability/opentelemetry-best-practices/
6. Microsoft,"Correlation ID Implementation Guide," Engineering Fundamentals Playbook. 2024. [Online]. Available: https://microsoft.github.io/code-with-engineering-playbook/observability/correlation-id/
7. Dr. Divyakant Meva and Hirenkumar Kukadiya, "Performance Evaluation of Large Language Models: A Comprehensive Review," ResearchGate, 2025. [Online]. Available:

https://www.researchgate.net/publication/390272225_Performance_Evaluation_of_Large_Language_Models_A_Comprehensive_Review

8. Venkatesh Balavadhani, et al., "The Ultimate Guide to Fine-Tuning LLMs from Basics to Breakthroughs: An Exhaustive Review of Technologies, Research, Best Practices, Applied Research Challenges and Opportunities," arXiv:2408.13296v1 [cs.LG], 2024. [Online]. Available: https://arxiv.org/html/2408.13296v1
9. Andrey Kharitonov, et al.,"Comparative analysis of machine learning models for anomaly detection in manufacturing," Procedia Computer Science, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050922003398
10. K. Gnana Sheela and Anu Rose Varghese, "Machine Learning based Health Monitoring System," ResearchGate, 2020. [Online]. Available: https://www.researchgate.net/publication/341747187_Machine_Learning_based_Health_Monitoring_System
11. Romain Cuer et al., "A formal framework for the safe design of the Autonomous Driving supervision," Reliability Engineering & System Safety, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0951832017305914
12. Björn Wahlström and Carl Rollenhagen "Safety management – A multi-level control problem," Safety Science, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0925753513001318