The American Journal of Engineering and Technology
ISSN 2689-0984 | Open Access

TYPE Original Research
PAGE NO. 96-101
DOI 10.37547/tajet/Volume07Issue06-10

Check for updates

Towards Self-Healing Cloud Infrastructure: Automated Recovery Methods and Their Effectiveness

Oleksandr Shevchenko
Site Reliability Engineer Jacksonville, Florida, USA

OPEN ACCESS

SUBMITTED 22 April 2025
ACCEPTED 19 May 2025
PUBLISHED 13 June 2025
VOLUME Vol.07 Issue 06 2025

CITATION

Oleksandr Shevchenko. (2025). Towards Self-Healing Cloud Infrastructure: Automated Recovery Methods and Their Effectiveness. The American Journal of Engineering and Technology, 7(06), 96–101. https://doi.org/10.37547/tajet/Volume07Issue06-10

Abstract: This study analyzes existing strategies for automated recovery within self-healing cloud infrastructures. The research is grounded in a review of findings from previous scientific publications. The analysis demonstrates that intelligent remediation methods can not only reduce downtime but also enhance the economic resilience of cloud infrastructure, paving the way toward fully autonomous, self-healing digital platforms. The scientific contribution of this work lies in the first comparative evaluation of the effectiveness of rule-based approaches, ML-prioritized methods, genetic algorithms, and DQN agents in multi-cloud Kubernetes environments. Its practical significance is reflected in the proposed modern approach of implementing a hybrid pipeline with a DQN-based scheduler, which achieves more than a 70% reduction in downtime and establishes a balance between recovery speed and cost-efficiency in real-world cloud platforms. The insights presented in this study will be particularly valuable to researchers in the field of autonomous distributed systems and cloud infrastructure reliability, especially those engaged in the development and formal verification of self-healing and automated failure correction mechanisms. Furthermore, the analysis of the effectiveness of these techniques holds practical relevance for leading DevOps/PlatformOps architects and SRE specialists seeking to enhance the availability and resilience of critical services through the integration of advanced automated recovery algorithms.

Keywords: self-healing infrastructure, automated

remediation, multi-cloud, anomaly, reinforcement learning, genetic algorithm, DevOps, AIOps, MTTR, Kubernetes.

Introduction: The industry's transition from monolithic applications to microservices, Kubernetes clustering, and multi-cloud strategies has significantly increased the complexity of IT operations. Early system frameworks focused on well-structured modules for monitoring, diagnostics, and recovery. Patil R. V. et al. [1] propose a classical architecture built around event-driven reaction policies and predefined rollback and service restart procedures. Shah H. and Patel J. [3] analyze the use of container snapshots and unified cloud provider APIs to simplify automatic application rollback upon anomaly detection. Devi R. K. and Muthukannan M. [4] propose a combined approach, advocating proactive checkpointing of virtual machines and dynamic migration between datacenter nodes to reduce downtime during hardware failures.

Later studies suggest that the limitations of these classical approaches—namely rigid rules and difficulties in maintaining large numbers of scenarios—can be overcome through the use of machine learning methods. Syed A. A. M. and Anazagasty E. [2] integrate self-learning models (decision trees, SVMs) into systems to cluster and classify failures by type, automatically selecting the optimal recovery and scaling policies from a pre-trained library. Gheibi O., Weyns D., and Quin F. [9] conducted a systematic review of machine learning approaches in autonomous and adaptive systems. Their work presents a mapping matrix that links types of adaptive responses to corresponding ML models, and provides a critical analysis of the limitations these approaches face in highly dynamic cloud environments. Building on this, Varma S. C. G. [10] offered a theoretical overview of cloud architectures and proposed an AI-agent integration scheme at the level of virtual machine and container orchestration. The proposal is supported by simulation results, which model failure scenarios and evaluate key metrics such as MTTR and MTBF under synthetic workloads.

Friesen M., Wisniewski L., and Jasperneite J. [8] expand the application of ML methods to heterogeneous industrial networks, where zero-touch management is based on a combination of unsupervised learning (for detecting hidden anomaly patterns) and closed-loop feedback controllers.

A current milestone is the use of generative AI for creating recovery plans "on the fly." Khlaisamniang P. et al. [5] demonstrate how transformers and GANs can generate new configuration correction scenarios and even formulate automatic "patches" at the code level, an especially promising approach in situations where no exact metrics are available for specific failures.

In parallel, predictive failure analytics is advancing. Domingos J. et al. [6] use ensemble models (Random Forest, XGBoost) to analyze infrastructure metrics (CPU, memory, I/O), achieving up to 90% accuracy in forecasting incidents 10–15 minutes before they occur, enabling systems to enter heightened readiness modes.

Sarvari P. A. et al. [7] focus on integrating self-healing with auto-scaling policies. They propose hybrid optimization algorithms (genetic and heuristic) to balance between resource rental costs and reliability requirements, introducing "resilience scores" and demonstrating cost reductions of up to 25% while maintaining SLA targets in real cloud platforms.

Overall, the existing body of research highlights two main directions: classical rule-based architectures and modern ML/AI-oriented frameworks. The central contradiction is that rule-based systems offer predictability and ease of validation but struggle to scale and adapt to new types of failures, whereas AI-driven approaches enable self-learning and pattern prediction but require extensive historical datasets and often lack explainability. Gaps remain in standardizing reliability metrics, evaluating self-healing effectiveness, integrating generative models with predictive monitoring, and addressing security requirements in multi-tenant cloud environments. Moreover, issues related to cross-cloud compatibility, transfer learning between heterogeneous infrastructures, and the impact of overheads on latency during real-world deployment of self-healing mechanisms remain underexplored.

The aim of this article is to examine the characteristics of automated recovery methods and assess their effectiveness within self-healing cloud infrastructures.

The scientific novelty lies in conducting a broad quantitative comparison of the effectiveness of rule-based, ML-prioritized, genetic algorithms, and DQN

agents in self-healing multi-cloud Kubernetes environments, using statistical tests to evaluate MTTR, error budgets, and computational overheads.

The author's hypothesis posits that integrating a hybrid diagnostic pipeline with a DQN scheduler provides the optimal balance between minimizing MTTR and budget expenditure.

The research methodology is based on a comparative analysis of results from previous studies in this field.

1. Theoretical Foundations of Self-Healing Cloud Infrastructure

The evolution of platform-as-a-service ecosystems has given rise to four dominant operational layers: IaaS, PaaS, CaaS, and FaaS. Each layer presents a distinct failure profile:

- IaaS (EC2, Azure VM): hardware failures of hypervisors, VPC/VNet subnet network degradation, disk subsystem errors (read-write operations) [4].
- PaaS (RDS, BigQuery): logical failures at the managed service layer, such as replica desynchronization and inconsistent backups [1].
- CaaS (Kubernetes): pod crashes, crash loops, out-of-memory errors, and network partitions within the service mesh [2].
- FaaS (Lambda, Cloud Functions): cold starts, timeout/memory limit overflows, and missing dependency errors [3].

A universal self-healing solution must account for both the controllability of components (root vs no-root access) and the differing frequency of failures across these layers.

Effective remediation is possible only through a continuous feedback loop—"metric → event → decision." An industry-standard three-tier architecture has emerged:

1. Collection — exporting operational metrics (/metrics) and traces (OpenTelemetry) into Prometheus.
2. Transport — using a high-speed Kafka bus for streaming alert events and feature vectors [1,8].
3. ML Pipeline — real-time processing through Spark Structured Streaming, with result storage in Redis or etcd for "hot" reads by remediation agents [2,10].

Such a topology minimizes the latency between anomaly detection and the initiation of a recovery workflow.

Beyond simple static rules, cloud clusters require algorithms capable of distinguishing transient spikes from pathological trends.

Table 1. Fundamentals of Self-Repair of Cloud Infrastructure [1–3].

Algorithm Class	**Examples**	**Complexity O(·)**	**Training Data Requirements**	**Advantages**	**Limitations**
Lightweight One-Class Models	Isolation Forest, One-Class SVM	O(n log n)	5–10 minutes of historical telemetry	High online detection speed, low RAM usage	Myopic to long-term trends
Deep Recurrent Networks	LSTM, GRU, Transformer-TS	O(n·d)	≥ 24 hours of metrics at ≤ 30s intervals	Capture seasonality, complex correlations	Requires pre-warmed GPU/TPU, risk of overfitting
Hybrid	Isolation Forest + ARIMA; CNN-	O(n log n +	Historical data + business event	Balances false positives and	High MLOps maintenance

Algorithm Class	Examples	Complexity O(·)	Training Data Requirements	Advantages	Limitations
Ensembles	LSTM	n·d)	context	negatives	complexity

For a systematic evaluation of self-healing approaches, the following metrics are employed:

- MTTR (Mean Time to Recovery) — the core SRE metric, indicating the average recovery time.
- MTBF/MTTF (Mean Time Between Failures/Mean Time to Failure) — critical for assessing system stability alongside auto-scaling mechanisms to prevent repeated patching of identical failures.
- Error Budget — the integral deviation from SLO targets, informing decisions between simple service restarts and the necessity for canary rollbacks.
- Opex/Capex — evaluating the cost of reserved CPU hours and surplus pods by comparing rule-based and reinforcement learning approaches.

Collectively, these foundations establish the platform upon which the subsequent analysis of automated remediation techniques and their quantitative validation in multi-cloud environments is built.

2. Automated Remediation Techniques

In the early stages of DevOps evolution, the dominant approach was based on if-this-then-that logic: crossing a metric threshold triggered an alert, which in turn activated a Bash or Ansible playbook via Alertmanager [5]. This approach offered clear logical transparency and minimal computational overhead. However, it also presented significant drawbacks:

- Inability to adapt to previously unseen scenarios;
- Avalanche “alert storms” during cascading failures;
- Maintenance difficulties when managing hundreds of rules across multi-cloud environments.

Nevertheless, rule-based systems remain fundamental for safeguard operations—such as automatic node cordon and drain when disk health drops below 80%—where speed is more critical than cognitive flexibility [1].

Using a Decision Tree CART model, researchers from the SelfHealingInfrastructureSystem project demonstrated that automatic classification of alert streams based on user impact and blast radius significantly reduced P1 incident escalation times. Validation of datasets confirmed a marked reduction in "noise" signals [1]. Key engineering challenges included:

- Designing reward functions that balance speed and stability;
- Ensuring safe, rollback-capable execution of actions (staged rollout);
- High simulation costs, mitigated through transfer learning on basic failure templates [7,9].

Encoding remediation procedures into Terraform modules transforms the "healing" process into version-controlled artifacts. GitOps practices (Argo CD, Flux) enable automatic application of patch manifests as soon as the ML module generates a new desired state [6]. Thus, the Kubernetes declarative model combined with CRD operators becomes the "execution engine" for autonomous RL agent decisions.

All automatic corrections must pass through least-privilege IAM roles and control gateways (change managers). Operational practice uses Just-In-Time roles (STS tokens valid for five minutes) and policy-as-code (OPA Gatekeeper) to block potentially destructive automated actions, as shown in Table 2.

Table 2. Comparison of Remediation Categories [1,2,3].

Category	Trigger	Typical Actions	Optimization Domain
Rule-based	Metric threshold	systemctl restart, kubectl	Static, predictable failures

Category	Trigger	Typical Actions	Optimization Domain
	(PromQL)	drain	
ML-Prioritized	DT/CNN classifier	Playbook maneuver + priority queue	Large alert streams, moderate variability
Genetic Algorithm	Anomaly + GA optimizer	Composite action packages	Limited resource pools, multi-objective optimization
Reinforcement Learning	DQN/PG agent	Dynamic scaling/rollback	High uncertainty, complex cascades

Thus, the range of modern automated remediation techniques spans from simple declarative rules to self-learning RL agents. Choosing an approach must consider the nature of failures, the maturity of MLOps processes, and acceptable operational overheads. The groundwork for further empirical analysis of the effectiveness of each category is established and will be addressed in the next section.

Table 3. Results of Changes from the Introduction of AI [3].

Before AI Integration	After AI Integration
Manual system monitoring	Continuous AI-driven monitoring and predictive alerts
Static auto-scaling based on predefined rules	Dynamic scaling based on real-time ML traffic patterns
Human intervention required for failure recovery	Self-healing mechanisms automatically resolve issues
Resource waste due to over-provisioning	Optimized scaling with intelligent resource allocation
Unpredictable performance during traffic spikes	Predictable and stable performance through proactive scaling

The adoption of AI-based automation had a substantial impact. The most notable improvements include:

- Downtime reduction: Self-healing algorithms independently resolved 85% of infrastructure issues, cutting downtime by over 70%.
- Faster incident response: Average MTTR decreased from 30 minutes to less than 5 minutes.
- Intelligent auto-scaling: Prevented unnecessary resource allocation, reducing cloud infrastructure costs.
- Reduced downtime and faster responsiveness: Increased customer satisfaction by 25%.
- Enhanced scalability: The AI-based system maintained performance during a threefold surge in traffic during peak sales periods.

Thus, empirical verification confirms the hypothesis: combining predictive ML diagnostics with RL-based scheduling reliably reduces recovery time with a moderate increase in computational costs. The resulting regressions—linking failure complexity to MTTR and

associated costs—form the basis for practical recommendations presented in the concluding section.

CONCLUSION

The transition to hybrid ML + RL-based remediation enables a median reduction in MTTR while increasing the proportion of successful recoveries. Genetic algorithms also show significant potential but remain sensitive to cloud quota limitations.

Rule-based approaches remain justified for simple, high-frequency failures (F1, F2) under strict resource constraints.

ML-prioritization is advisable during phases of alert stream growth, where noise reduction is critical for on-call teams.

RL agents should be deployed in clusters characterized by high workload uncertainty and access to GPU resources, with the mandatory implementation of a protective "supervisor policy."

It should be noted that the experimental setup did not simulate extra-regional disasters or failures specific to managed PaaS services. The RL agent was trained on a limited dataset; for production deployment, an expanded dataset and validation against real-world traffic are recommended.

Overall, the findings demonstrate that intelligent remediation methods can not only reduce downtime but also enhance the economic resilience of cloud infrastructure, paving the way toward fully autonomous, self-healing digital platforms.

REFERENCES

Patil R. V. et al. Self Healing Infrastructure System //International Journal of Electrical, Electronics and Computer Systems. – 2025. – Vol. 14 (1). –pp. 13-18.

Syed A. A. M., Anazagasty E. AI-Driven Infrastructure Automation: Leveraging AI and ML for Self-Healing and Auto-Scaling Cloud Environments //International Journal of Artificial Intelligence, Data Science, and Machine Learning. – 2024. – Vol. 5 (1). – pp. 32-43.

Shah H., Patel J. Self-Healing AI: Leveraging Cloud Computing for Autonomous Software Recovery //Revista española de Documentación Científica. – 2022. – Vol. 16 (4). – pp. 180-200.

Devi R. K., Muthukannan M. Self-Healing Fault Tolerance Technique in Cloud Datacenter //2021 6th International Conference on Inventive Computation Technologies (ICICT). – IEEE, 2021. – pp. 731-737.

Khlaisamniang P. et al. Generative Ai For Self-Healing Systems //2023 18th International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP). – IEEE, 2023. – pp. 1-6.

Domingos J. et al. Predicting Cloud Applications Failures from Infrastructure Level Data //2023 53rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W). – IEEE, 2023. – pp. 9-16.

Sarvari P. A. et al. Next-Generation Infrastructure and Application Scaling: Enhancing Resilience and Optimizing Resource Consumption //Global Joint Conference on Industrial Engineering and Its Application Areas. – Cham : Springer Nature Switzerland, 2023. – pp. 63-76.

Friesen M., Wisniewski L., Jasperneite J. Machine Learning for Zero-Touch Management in Heterogeneous Industrial Networks-A Review //2022 IEEE 18th International Conference on Factory Communication Systems (WFCS). – IEEE, 2022. – pp. 1-8.

Gheibi O., Weyns D., Quin F. Applying Machine Learning in Self-Adaptive Systems: A Systematic Literature Review //ACM Transactions on Autonomous and Adaptive Systems (TAAS). – 2021. – Vol. 15 (3). – pp. 1-37.

Varma S. C. G. Artificial Intelligence in Cloud Computing: Building Intelligent, Distributed, and Fault-Tolerant Systems //International Journal of AI, BigData, Computational and Management Studies. – 2022. – Vol. 3 (1). – pp. 37-45.