International Journal of Emerging Trends in Engineering and Technology
ISSN: 3107-8699 | Volume 2 Issue 1, 33-41, April-June 2026
DOI: https://doi.org/10.64137/31078699/IJETET-V2I1P106

Received: 03/02/2026 Revised: 25/02/2026 Accepted: 27/02/2026 Published: 01/03/2026

Original Article

AI-Driven Cloud-Native Microservices Framework for Secure Healthcare Prescription Automation, Software Reliability, and Scalable Deployment Optimization

SRIKANTH REDDY GUDI
Cigna Evernorth Health Services Inc, Charlotte, North Carolina, USA.

ABSTRACT: *Healthcare prescription automation is increasingly dependent on interoperable clinical data exchange, cloud-native service orchestration, secure application programming interfaces, and dependable software delivery pipelines. However, current prescription automation platforms often treat clinical decision support, e-prescribing workflow execution, software reliability engineering, cybersecurity governance, and deployment optimization as separate concerns. This separation creates architectural fragmentation in environments where prescription requests must be clinically valid, auditable, resilient to distributed failure, compliant with privacy obligations, and scalable under fluctuating enterprise workloads. This paper proposes a conceptual AI-driven cloud-native microservices framework for secure healthcare prescription automation, software reliability, and scalable deployment optimization. The framework integrates modular prescription services, AI-assisted clinical and operational intelligence, policy-driven security controls, observability-centered reliability engineering, and container-based deployment automation into a unified architecture. The major contribution of this paper is not an empirical claim of clinical superiority, but a structured reference model for designing prescription automation platforms that can support high-assurance workflows in regulated healthcare environments. The study identifies practical gaps in existing approaches, including a weak linkage between prescription standards and runtime reliability, insufficient integration of defect prediction with development pipelines, limited explainability in AI-assisted prescription decisions, and fragmented governance across APIs, containers, and machine learning components. The paper further presents a comparative methodology, architectural layers, implementation considerations, expected analytical outcomes, risk limitations, and future research directions. The proposed framework can guide healthcare enterprises, cloud architects, software reliability engineers, and AI governance teams in designing secure, scalable, and auditable prescription automation systems without binding the architecture to a single vendor or proprietary platform.*

KEYWORDS: *Artificial Intelligence, Cloud-Native Systems, Microservices, Healthcare Prescription Automation, Software Reliability, E-Prescribing, Cyber Security, Cabernets, Mops, Develops, Observability, Clinical Decision Support.*

1. INTRODUCTION

The need for a stronger architecture is amplified by the clinical and operational risks associated with medication automation. Computerized provider order entry and clinical decision support systems have historically been associated with improvements in medication safety. Still, their benefits depend heavily on implementation quality, alert design, workflow integration, and the reliability of the surrounding software ecosystem [29]. An automated prescription system that generates accurate clinical recommendations but fails during service orchestration may still result in delayed, duplicated, or incomplete prescriptions. Similarly, a system that scales technically but lacks privacy and authorization controls may expose sensitive health information. Therefore, secure prescription automation requires a systems-level design that considers clinical logic, software reliability, security, and deployment optimization together rather than as independent layers.

Interoperability standards such as HL7 FHIR and SMART on FHIR have improved the ability of healthcare applications to exchange clinical data and integrate with electronic health record ecosystems [4]. Prescription-specific exchange further depends on e-prescribing standards and implementation guidance such as NCPDP SCRIPT, which supports transaction types for new prescriptions, changes, renewals, cancellations, and related pharmacy communications [13]. However, standards alone do not solve the architectural challenge of building reliable, secure, AI-assisted, cloud-native platforms. Standards define exchange structure and semantic expectations, while runtime platforms must enforce security policies, perform consistency checks, recover from failures, detect anomalies, and scale under variable demand.

Artificial intelligence can contribute to prescription automation in several ways. AI models can support drug interaction risk ranking, prior authorization routing, prescription error detection, anomaly detection, workload prediction, software defect prediction, and operational root cause analysis. Recent work on software fault prediction using neural networks and machine learning demonstrates the potential of AI models to identify defect-prone components and improve software quality assurance

[2]. Hybrid deep learning models that combine convolutional, recurrent, and dense representations further indicate that software reliability signals can be modeled across structural and temporal dimensions [7]. Nevertheless, integrating AI into prescription automation creates new challenges around explainability, drift monitoring, model validation, data privacy, and accountability. A model that predicts a high-risk prescription event must produce information that clinicians and governance teams can interpret. In contrast, an operational AI model that recommends scaling or remediation must remain constrained by safety policies.

Cloud-native deployment technologies provide another necessary but incomplete foundation. Kubernetes and related container orchestration systems support automated scaling, scheduling, service discovery, and deployment management. Lessons from large-scale cluster management systems show that orchestration platforms can improve utilization and operational control when combined with monitoring and policy mechanisms [18]. However, healthcare systems cannot adopt cloud-native automation as a purely infrastructure-driven exercise. They require security controls aligned with protected health information, API governance, auditability, container image assurance, runtime isolation, and policy enforcement. NIST guidance on application container security emphasizes the need to manage container images, registries, orchestrators, hosts, and runtime risks across the full container lifecycle [6].

This paper addresses the following research gap: existing literature and industry architectures often examine healthcare interoperability, AI-assisted clinical decision support, cloud-native microservices, software defect prediction, and deployment automation separately. There is insufficient architecture-level integration of these domains for secure healthcare prescription automation. The absence of a unified framework creates design ambiguity for enterprises that must simultaneously support clinical correctness, system reliability, cybersecurity, observability, and scalable deployment.

This paper makes five primary contributions. First, it proposes a structured AI-driven cloud-native microservices framework for healthcare prescription automation. Second, it compares existing architectural approaches and identifies practical gaps related to security, reliability, AI governance, and deployment optimization. Third, it presents a vendor-neutral reference architecture that can be adapted to public, private, and hybrid cloud environments, as well as regulated enterprise environments. Fourth, it connects prescription automation to enterprise relevance by mapping architectural layers to clinical workflow resilience, protected data handling, software quality, and operational scalability. Fifth, it identifies limitations and future research directions, including validation strategies, model governance, federated learning, and compliance-aware autonomous remediation.

2. BACKGROUND AND RELATED WORK

Microservices emerged as a response to the limitations of monolithic enterprise applications by decomposing functionality into independently deployable services aligned with business capabilities. In prescription automation, this decomposition may separate patient profile retrieval, provider authorization, medication selection, clinical rule evaluation, formulary checking, prescription generation, pharmacy routing, audit logging, notification, and exception handling. The advantage is modular development and independent scaling. The risk is that correctness is distributed across service boundaries, making dependency modeling and failure-propagation analysis essential. Graph-based modeling of service dependencies is particularly relevant because prescription workflows often involve chained interactions one which a degraded service can affect multiple downstream decisions [9].

Healthcare interoperability research provides a second foundation. FHIR supports resource-oriented exchange of clinical data, while SMART on FHIR enables application integration across health information systems [16]. In prescription automation, these standards support patient medication history retrieval, allergy access, encounter context, practitioner identity, and integration with clinical applications. Yet interoperability does not eliminate the need for local validation, access control, data minimization, semantic mapping, and workflow-specific orchestration. Prescription automation must therefore combine standards-based exchange with internal domain services and governance controls.

Clinical decision support software has a long history in medication safety, but AI-assisted prescription systems must be designed carefully. FDA guidance on clinical decision support software distinguishes between functions that may be excluded from device regulation and those that may require regulatory oversight, depending on intended use, user type, and whether the basis for recommendations can be independently reviewed [11]. This distinction is important for AI-driven prescription automation because opaque models can create governance uncertainty. A framework that uses AI must therefore separate advisory intelligence from authoritative clinical action, maintain explainability, and preserve human accountability in high-risk decision points.

Software reliability and AI-based quality assurance provide a third research stream. Machine learning approaches to defect prediction have been applied to classify fault-prone software modules, compare predictive models, and improve software quality prioritization [12]. Additional studies of machine learning techniques for software defect prediction show that model selection, feature engineering, and evaluation strategy influence predictive usefulness [21]. In prescription automation, these

techniques can be extended beyond general software quality to prioritize testing for services with high clinical workflow criticality, high change frequency, or high incident correlation.

AI-driven lifecycle governance connects defect prediction, automated testing, and quality assurance to broader software engineering management [5]. Such work supports the argument that prescription automation should not treat AI as only a clinical layer. AI can be used across the lifecycle: requirements risk classification, code quality prediction, test case prioritization, deployment risk scoring, incident triage, root cause analysis, and capacity forecasting. This broader view is consistent with AI-based systems engineering perspectives that integrate lifecycle governance, predictive quality assurance, automation economics, and cybersecurity intelligence [15].

Cybersecurity guidance is also central. The HIPAA Security Rule establishes administrative, physical, and technical safeguards for protecting electronic protected health information [8]. NIST CSF 2.0 provides a broader risk management framework that helps organizations govern, identify, protect, detect, respond to, and recover from cybersecurity risks [3]. API security is especially relevant because prescription automation platforms expose endpoints for patient data retrieval, prescription submission, status checks, pharmacy communication, and third-party integrations. The OWASP API Security Top 10 identifies common API risks such as broken object-level authorization and inadequate access control, both of which are directly relevant to prescription services [14].

Finally, cloud-native observability and operations research inform deployment optimization. Open Telemetry defines vendor-neutral telemetry specifications for traces, metrics, and logs [30]. Site reliability engineering practices emphasize service level objectives, error budgets, monitoring, automation, and operational discipline for scalable systems [34]. These concepts are essential because prescription automation platforms must not only function under normal conditions but degrade safely, recover predictably, and provide auditable evidence during incidents.

3. PROBLEM STATEMENT AND RESEARCH MOTIVATION

The central problem addressed in this paper is the absence of an integrated architecture for secure, AI-assisted, cloud-native prescription automation that simultaneously accounts for clinical workflow integrity, software reliability, cybersecurity, and scalable deployment. Many enterprises modernize prescription workflows incrementally. They may first add APIs to legacy systems, then introduce microservices, then add AI-based recommendations, and later migrate selected workloads to containers. This sequence may produce short-term delivery gains but can also leave gaps in governance, traceability, resilience, and risk management.

Three practical deficiencies motivate this research. The first is architectural fragmentation. Clinical decision support, e-prescribing standards, service orchestration, security policy, and deployment automation are often governed by different teams and tools. As a result, the correctness of the prescription workflow may depend on implicit assumptions that are not visible in runtime telemetry or deployment policies. The second deficiency is insufficient reliability of intelligence. Traditional monitoring can detect failures after they occur, but prescription automation requires predictive and preventive controls, especially when failures may delay patient care or create an administrative burden. AI-enabled defect prediction and incident analysis can help prioritize quality interventions before production failure occurs [26]. The third deficiency is weak integration of AI governance. AI models used for prescription risk scoring, operational anomaly detection, or deployment recommendation must be evaluated for fairness, drift, explainability, security, and appropriate human oversight. The NIST AI Risk Management Framework is relevant because it frames AI risk as a lifecycle governance problem rather than a one-time model validation issue [22].

The research motivation is therefore to design a framework that treats prescription automation as a socio-technical system. Clinical knowledge, software architecture, AI models, cloud infrastructure, security policy, and enterprise operations must be coordinated. A prescription automation framework should support safe automation while avoiding the unrealistic assumption that AI can replace clinical accountability or that cloud-native platforms automatically provide reliability.

4. PROPOSED CONCEPTUAL FRAMEWORK OR ARCHITECTURE

This paper proposes the AI-Driven Cloud-Native Prescription Automation and Reliability Framework (AICN-PARF). The framework is a vendor-neutral reference architecture composed of seven layers: channel and identity layer, interoperability layer, prescription domain microservices layer, AI intelligence layer, policy and security layer, reliability and observability layer, and deployment optimization layer.

The channel and identity layer manages user access through clinician portals, EHR-integrated applications, pharmacy interfaces, patient applications, and enterprise service clients. This layer enforces authentication, authorization, session controls, and role-aware access. It separates user identity from service identity, allowing clinician actions, machine-to-machine calls, and automated background processes to be audited independently.

The interoperability layer translates and validates external healthcare data exchange. It supports FHIR-based retrieval of patient context, medication history, allergies, conditions, practitioner information, and encounter data. It also supports prescription transaction integration using NCPDP SCRIPT, where appropriate. The layer is not limited to protocol transformation. It also performs semantic validation, schema version management, terminology mapping, and data minimization to reduce unnecessary exposure of protected information.

The prescription domain microservices layer contains bounded services aligned with prescription workflow capabilities. These services include patient medication context, medication catalog, dose validation, allergy and interaction screening, formulary and benefit check, prior authorization routing, prescription assembly, pharmacy routing, cancellation and change management, notification, and audit logging. Each service owns clear interfaces and data responsibilities. The architecture discourages shared database coupling because shared persistence can create hidden dependencies and complicate incident isolation.

The AI intelligence layer provides decision support and operational intelligence. Clinical AI services may rank potential medication conflicts, flag unusual dose patterns, detect duplicate therapy risk, and classify prescriptions requiring additional review. Operational AI services may predict defect-prone services, detect anomalous latency patterns, estimate workload spikes, recommend test prioritization, and support root cause analysis. The framework requires AI outputs to be confidence-scored, explainable, logged, and bounded by policy. AI should advise, prioritize, or route decisions rather than silently execute high-risk clinical actions.

The policy and security layer implements zero-trust service communication, API authorization, consent-aware access rules, encryption, secrets management, container admission control, software supply chain checks, and policy-as-code enforcement. Open Policy Agent or similar policy engines can help externalize authorization and compliance decisions across APIs, Kubernetes admission controls, and CI/CD workflows [31]. Security policies are mapped to prescription workflow context, such as whether a service may access medication history, whether a pharmacy routing request contains the minimum necessary data, or whether an AI model is approved for a specific clinical use case.

The reliability and observability layer collects distributed traces, metrics, logs, audit events, model telemetry, and prescription workflow state transitions. It correlates business-level events, such as prescription submission failures, with technical signals, including service latency, retry exhaustion, circuit breaker activation, dependency timeouts, or database saturation. The goal is not merely monitoring but explainable operational evidence. This layer supports service-level objectives for prescription submission latency, transaction completion, duplicate prevention, completeness of data access audit, and recovery time.

The deployment optimization layer manages containerized delivery, autoscaling, rollout strategies, resource allocation, and resilience testing. Cabernets Horizontal Pod Autoscaler can adjust replicas based on CPU, memory, or custom metrics. Still, prescription automation may require domain-specific scaling signals such as prescription queue depth, pharmacy routing backlog, clinical validation latency, or prior authorization request volume [32]. Deployment strategies should include canary releases, blue-green deployments, progressive delivery, automated rollback, and risk-scored release gates.

5. METHODOLOGY OR COMPARATIVE ANALYSIS

Because this paper is conceptual and architecture-oriented, the methodology is based on comparative architectural analysis rather than experimental benchmarking. The analysis compares four common approaches to modernizing prescription platforms.

The first approach is monolithic prescription automation. Its strength is transactional simplicity and centralized control. Its weakness is limited scalability, slow release cycles, and difficulty isolating failures. A defect in prescription assembly may require redeploying unrelated capabilities, and performance bottlenecks can affect the entire application.

The second approach is API-enabled legacy modernization. This approach exposes legacy prescription capabilities through APIs while preserving underlying systems. It improves integration but often leaves core reliability and scalability limitations unresolved. API gateways may hide legacy complexity, but do not eliminate hidden coupling, poor observability, or brittle batch-oriented workflows.

The third approach is conventional cloud-native microservices. This model improves modularity and deployment flexibility. However, without healthcare-specific policy modeling, AI governance, and domain-level observability, microservices can increase operational complexity. Distributed systems introduce risks of partial failure, eventual consistency, network dependencies, and version compatibility.

The fourth approach is the proposed AICN-PARF model. This approach integrates microservices, AI-assisted intelligence, security policy, observability, and deployment optimization into a single prescription-focused framework. Its strength is

architectural completeness. Its weakness is implementation complexity, governance overhead, and the need for cross-functional maturity.

Comparison criteria include clinical workflow integrity, interoperability readiness, security enforceability, reliability intelligence, deployment scalability, AI governance, auditability, and enterprise adaptability. The proposed framework performs strongest where prescription systems require coordinated governance across clinical, technical, and operational domains. However, it may be excessive for small systems with limited transaction volume or low integration complexity.

6. TECHNICAL DISCUSSION

The technical originality of the proposed framework lies in treating prescription automation as a reliability-governed AI microservice ecosystem rather than as a conventional healthcare application. This distinction changes how services are designed, deployed, monitored, and governed.

First, the framework introduces workflow-aware service boundaries. A medication interaction service should not be designed only as a reusable API. It should expose traceable decision outputs, input provenance, confidence or rule basis, exception codes, and audit identifiers. A pharmacy routing service should not simply transmit messages. It should maintain idempotency keys, retry policies, duplicate detection, delivery status, and compensating workflows for cancellation or change events.

Second, the framework applies AI to both clinical and software reliability domains. Neural network research on software fault detection suggests that architectural features, code metrics, and historical defect patterns can be used to improve the identification of defect-prone components [17]. In AICN-PARF, this capability can be connected to CI/CD pipelines so that high-risk prescription services receive deeper test coverage, stricter approval gates, and targeted observability. However, the framework avoids claiming that AI prediction eliminates the need for testing. It positions AI as a prioritization mechanism for human and automated quality controls.

Third, dependency-aware reliability is essential. Microservice systems often fail through indirect propagation. For example, a latency increase in formulary checking may cause prescription submission timeouts, triggering retries, which may increase message queue load, delaying pharmacy routing. Dependency graphs and causal telemetry can help identify whether failures originate in clinical validation, external payer systems, pharmacy networks, or infrastructure saturation. Reliability patterns such as circuit breakers, bulkheads, timeouts, idempotent retries, and backpressure remain necessary for production systems [33].

Fourth, security must be embedded into runtime and delivery workflows. Container images should be scanned, signed, and governed before deployment. Kubernetes clusters should enforce least privilege, network policies, secret protection, admission control, and runtime monitoring [27]. API endpoints should verify object-level authorization because prescription data is highly sensitive and patient-specific. Security should not rely only on perimeter gateways; it must be enforced at service, data, and workflow levels.

Fifth, AI governance must be operationalized. The framework recommends model cards, data lineage, feature drift monitoring, bias review, clinical validation boundaries, and rollback procedures for AI models. AI-assisted prescription recommendations should expose the basis for the recommendation when used by clinicians. Operational AI models should be constrained by deployment policies so that scaling or rollback recommendations do not override safety rules.

7. PRACTICAL IMPLEMENTATION OR ENTERPRISE RELEVANCE

An enterprise implementation of AICN-PARF can proceed through phased modernization. The first phase is domain decomposition. Healthcare enterprises should identify prescription workflow capabilities, map service dependencies, define data ownership, and classify services by clinical criticality. Services involved in medication validation, authorization, pharmacy routing, and audit logging should receive higher reliability and security controls than low-risk administrative services.

The second phase is interoperability alignment. Enterprises should implement FHIR interfaces for the clinical context and, where relevant, NCPDP-aligned prescription transaction handling. The ONC Cures Act Final Rule strengthens the importance of secure electronic health information access and standards-based interoperability in healthcare ecosystems [19]. Therefore, prescription automation platforms should be designed for standards-based exchange rather than for closed integration.

The third phase is the integration of develops and mops. Source code, infrastructure definitions, policy rules, model artifacts, and deployment manifests should be versioned and promoted through controlled pipelines. Software engineering for machine learning differs from traditional software because models depend on data, features, training pipelines, evaluation criteria, and operational feedback loops [24]. Hidden technical debt in machine learning systems can arise from unstable data dependencies,

configuration complexity, undeclared consumers, and feedback loops [25]. These risks are amplified in prescription automation, where data quality and model interpretation affect clinical workflow.

The fourth phase is observability and reliability engineering. Enterprises should instrument services with distributed tracing, business event correlation, and service level indicators. Observability should include prescription-specific metrics such as validation completion time, pharmacy routing success rate, duplicate prescription prevention rate, model inference latency, policy denial rate, and exception resolution time. ISO/IEC 25010:2023 provides a useful quality model for reasoning about software product characteristics such as reliability, security, maintainability, compatibility, and performance efficiency [28].

The fifth phase is deployment optimization. Cabernets-based deployments can support horizontal scaling, rolling updates, and workload isolation, but enterprises should tune scaling metrics to align with prescribed workload patterns. For example, the medication validation service may need low-latency scaling during clinic peak hours, while batch reconciliation services may use lower-priority compute classes. Deployment should incorporate canary analysis, automated rollback, policy checks, and chaos testing in non-production environments.

The sixth phase is governance and audit readiness. Every automated prescription decision should preserve traceability from input data to service execution, model output, policy decision, and final transaction status. This audit chain supports compliance, incident investigation, and clinical accountability. AI should be implemented as governed decision support rather than unbounded automation.

8. RESULTS, EXPECTED OUTCOMES, OR ANALYTICAL INSIGHTS

Since this paper proposes a conceptual framework rather than reporting experimental results, the outcomes are expressed as analytical insights and expected evaluation dimensions. The proposed framework is expected to improve architectural clarity by giving enterprises a structured model for aligning prescription services, AI intelligence, security controls, observability, and deployment automation. It is also expected to improve reliability governance by connecting defect prediction, dependency modeling, telemetry, and release risk scoring.

A second expected outcome is a stronger security posture. By embedding policy enforcement into APIs, service-mesh boundaries, cabernets admission, and CI/CD workflows, the framework reduces reliance on manual review and on isolated gateway controls. The approach aligns with cybersecurity governance principles that require risk management across identification, protection, detection, response, and recovery activities [3].

A third insight is that AI should be used differently across clinical and operational contexts. Clinical AI should be conservative, explainable, and human-supervised. Operational AI can be more autonomous in low-risk domains, such as test prioritization or anomaly triage, but should remain bounded when recommending production remediation. AI/ML-powered root cause analysis and automated remediation can support faster operational response, but remediation must be constrained by safety policies in healthcare systems [26].

A fourth insight is that scalable deployment optimization should be domain-aware. Generic CPU-based autoscaling may not adequately represent prescription workload pressure. Queue depth, validation latency, external dependency saturation, and prescription exception volume may provide better custom scaling indicators. Therefore, cloud-native scalability should be evaluated not only by infrastructure utilization but also by workflow completion and clinical service continuity.

A fifth insight is that prescription automation requires an evidence-oriented architecture. Enterprises should be able to reconstruct why a prescription was accepted, modified, routed, rejected, retried, delayed, or escalated. This evidence should include clinical data inputs, policy checks, service traces, AI outputs, and user actions. Such traceability is essential for reliability engineering, compliance review, and trust.

9.CHALLENGES, LIMITATIONS, AND RISK CONSIDERATIONS

The proposed framework has several limitations. First, it is conceptual and requires empirical validation through prototype implementation, simulation, or enterprise case studies. The paper does not claim measured reductions in prescription errors, latency, or security incidents. Such claims would require controlled evaluation using representative workflows, datasets, operational traces, and safety review.

Second, AI model governance remains difficult in healthcare. Prescription data may be incomplete, biased, institution-specific, or constrained by privacy requirements. Models trained in one health system may not generalize to another. Federated learning may reduce centralization of sensitive data by allowing model training across distributed data sources, but it introduces communication, heterogeneity, privacy, and governance challenges [35].

Third, cloud-native microservices can increase complexity. Each service boundary adds network communication, version management, monitoring requirements, and possible failure points. Without disciplined platform engineering, microservices can become harder to operate than the monolith they replaced. The framework, therefore, requires mature DevSecOps, observability, incident response, and governance capabilities.

Fourth, security risks remain significant. APIs may expose sensitive prescription resources, containers may inherit vulnerable dependencies, and AI pipelines may introduce data leakage or adversarial risks. NIST container security guidance and OWASP API security practices reduce risk but do not eliminate the need for continuous security assessment [14].

Fifth, regulatory classification can be complex. AI-assisted prescription features may fall into different regulatory categories depending on intended use, user population, explainability, and degree of automation. Enterprises must involve legal, compliance, clinical safety, and cybersecurity stakeholders before production deployment.

10. FUTURE RESEARCH DIRECTIONS

Future research should validate the proposed framework through prototype implementation and controlled evaluation. One direction is to build a reference implementation using FHIR-based patient context, synthetic prescription workflows, containerized services, distributed tracing, and AI-assisted defect prediction. Evaluation metrics could include prescription transaction latency, service recovery time, duplicate prevention accuracy, policy enforcement coverage, test prioritization efficiency, and incident detection time.

A second direction is to develop benchmark datasets for evaluating the reliability of prescription automation. Current healthcare datasets may support clinical modeling, but fewer datasets capture service traces, prescription workflow exceptions, deployment events, and reliability incidents. Synthetic yet realistic datasets could help researchers evaluate AI-driven operational intelligence without exposing protected health information.

A third direction is explainable AI for supporting prescription workflows. Future models should produce clinically interpretable outputs and distinguish between rule-based alerts, probabilistic recommendations, and workflow routing suggestions. Explainability is especially important when AI recommendations influence medication safety.

A fourth direction is compliance-aware autonomous remediation. Current AIOps systems may recommend or execute remediation, but healthcare environments require stronger controls. Future research should explore policy-constrained remediation agents that can restart services, adjust scaling, or roll back deployments only when safety and compliance conditions are satisfied.

A fifth direction is federated and privacy-preserving learning for healthcare software reliability. Federated approaches may allow multiple healthcare organizations to improve defect prediction, anomaly detection, or prescription risk models without centralizing sensitive data. However, governance, model drift, privacy leakage, and fairness require further study.

11. CONCLUSION

This paper proposed an AI-driven cloud-native microservices framework for secure healthcare prescription automation, software reliability, and scalable deployment optimization. The study argued that prescription automation should not be treated as a narrow e-prescribing transaction problem or as a generic cloud migration exercise. It is a high-assurance socio-technical system that requires clinical workflow integrity, standards-based interoperability, secure API design, AI governance, software reliability engineering, observability, and cloud-native deployment maturity.

The proposed AICN-PARF architecture integrates seven layers: channel and identity; interoperability; prescription domain micro services; AI intelligence; policy and security; reliability and observability; and deployment optimization. The framework provides a vendor-neutral reference model that healthcare enterprises can adapt to regulated environments. It identifies practical gaps in current approaches, including fragmented governance, insufficient dependency-aware reliability, weak integration of AI with developers, and a lack of domain-specific deployment optimization.

The analysis indicates that AI can strengthen prescription automation when used as governed decision support and operational intelligence rather than as unbounded automation. Cloud-native microservices can improve modularity and scalability when paired with strong security, observability, and reliability patterns. The paper also acknowledges limitations, including the need for empirical validation, the complexity of AI governance, regulatory uncertainty, and operational maturity requirements. Future research should focus on reference implementations, synthetic reliability benchmarks, explainable prescription intelligence, compliance-aware remediation, and privacy-preserving multi-institution learning.

REFERENCES

[1] N. Dragoni *et al.*, "Microservices: Yesterday, Today, and Tomorrow," *Present and Ulterior Software Engineering*, pp. 195–216, 2017, doi: https://doi.org/10.1007/978-3-319-67425-4_12.

[2] S. K. Gunda, "Advancing software fault detection: A comparative study of neural network architectures," *PROCEEDINGS OF THE 2025 12TH INTERNATIONAL CONFERENCE ON MECHANICS, MATERIALS AND MANUFACTURING: ICMMM2025*, p. 020212, Jan. 2026, doi: https://doi.org/10.1063/5.0298095.

[3] NIST, "The NIST Cyber Security Framework (CSF) 2.0," *The NIST Cyber Security Framework (CSF) 2.0*, vol. 2.0, no. 29, Feb. 2024, doi: https://doi.org/10.6028/nist.cswp.29.

[4] J. C. Mandel, D. A. Kreda, K. D. Mandl, I. S. Kohane, and R. B. Ramoni, "SMART on FHIR: a standards-based, interoperable apps platform for electronic health records," *Journal of the American Medical Informatics Association*, vol. 23, no. 5, pp. 899–908, Feb. 2016, doi: https://doi.org/10.1093/jamia/ocv189.

[5] S. D. Sivva, R. R. Thalakanti, S. S. G. Bandari, and S. D. R. Yettapu, "AI-Driven Decision Intelligence for Agile Software Lifecycle Governance: An Architecture-Centered Framework Integrating Machine Learning Defect Prediction and Automated Testing," *International Journal of Emerging Trends in Computer Science and Information Technology*, vol. 4, pp. 167–172, 2023, doi: https://doi.org/10.63282/3050-9246.ijetcsit-v4i4p118.

[6] M. Souppaya, J. Morello, and K. Scarfone, "Application container security guide," *Application Container Security Guide*, Sep. 2017, doi: https://doi.org/10.6028/nist.sp.800-190.

[7] S. K. Gunda, "A Hybrid Deep Learning Model for Software Fault Prediction Using CNN, LSTM, and Dense Layers," *Communications in Computer and Information Science*, pp. 282–290, Oct. 2025, doi: https://doi.org/10.1007/978-3-032-05144-8_21.

[8] U.S. Department of Health and Human Services, "The Security Rule," *HHS.gov*, Oct. 20, 2022. https://www.hhs.gov/hipaa/for-professionals/security/index.html

[9] N. Mutyam, "Graph-Based Modeling of Service Dependencies for Predicting Failure Propagation in Distributed Systems," *International Journal of Multidisciplinary Evolutionary Research*, vol. 5, no. 1, pp. 113–116, 2024, doi: https://doi.org/10.54660/ijmer.2024.5.1.113-116.

[10] B. Burns, B. Grant, D. Oppenheimer, E. Brewer, and J. Wilkes, "Borg, Omega, and cabernets," *Queue*, vol. 14, no. 1, pp. 70–93, Jan. 2016, doi: https://doi.org/10.1145/2898442.2898444.

[11] Center for Devices and Radiological Health, "Clinical Decision Support Software - Draft Guidance," *U.S. Food and Drug Administration*, 2019. https://www.fda.gov/regulatory-information/search-fda-guidance-documents/clinical-decision-support-software

[12] S. K. Gunda, "Comparative Analysis of Machine Learning Models for Software Defect Prediction," pp. 1–6, Oct. 2024, doi: https://doi.org/10.1109/icpects62210.2024.10780167.

[13] National Council for Prescription Drug Programs, *SCRIPT Implementation Recommendations*, Version 1.76, May 2026. https://ncpdp.org/NCPDP/media/pdf/SCRIPT-Implementation-Recommendations.pdf

[14] OWASP, "OWASP Top 10 API Security Risks – 2023," *owasp.org*, 2023. https://owasp.org/API-Security/editions/2023/en/0x11-t10/

[15] S. D. Sivva, "An End-to-End AI-Based Systems Engineering Paradigm for Lifecycle Governance, Predictive Quality Assurance, Automation Economics, and Cyber Security Intelligence," *Journal of Frontiers in Multidisciplinary Research*, vol. 4, no. 1, pp. 600–604, 2023, doi: https://doi.org/10.54660/.jfmr.2023.4.1.600-604.

[16] National Institutes of Health, "HL7® Releases FHIR® v5.0 | Data Science at NIH," *Nih.gov*, 2023. https://datascience.nih.gov/content/hl7%C2%AE-releases-fhir%C2%AE-v50

[17] R. R. Thalakanti, "Enhancing Convergence in Fully Connected Neural Networks via optimized back propagation," *2025 2nd International Conference on Computing and Data Science (ICCDS)*, pp. 1–6, Jul. 2025, doi: https://doi.org/10.1109/iccds64403.2025.11209625.

[18] A. Verma, L. Pedrosa, M. Korupolu, D. Oppenheimer, E. Tune, and J. Wilkes, "Large-scale cluster management at Google with Borg," *Proceedings of the Tenth European Conference on Computer Systems*, Apr. 2015, doi: https://doi.org/10.1145/2741948.2741964.

[19] "ONC's Cures Act Final Rule," *ASTP - Assistant Secretary for Technology Policy*, Jan. 15, 2026. https://healthit.gov/regulations/cures-act-final-rule/

[20] M. Balerao, "A Converged Artificial Intelligence Architecture for Innovation, Software Lifecycle Optimization, and Cyber Security Risk Mitigation," *International Journal of Multidisciplinary Futuristic Development*, vol. 4, no. 1, pp. 117–120, 2023, doi: https://doi.org/10.54660/ijmfd.2023.4.1.117-120.

[21] S. K. Gunda, "Analyzing Machine Learning Techniques for Software Defect Prediction: A Comprehensive Performance Comparison," *2024 Asian Conference on Intelligent Technologies (ACOIT)*, pp. 1–5, Sep. 2024, doi: https://doi.org/10.1109/acoit62457.2024.10939610.

[22] NIST, "AI Risk Management Framework," *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*, vol. 1, no. 1, Jan. 2023, doi: https://doi.org/10.6028/nist.ai.100-1.

[23] A. Avizienis, J.-C. Laprie, B. Randell, and C. Landwehr, "Basic concepts and taxonomy of dependable and secure computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 11–33, Jan. 2004, doi: https://doi.org/10.1109/tdsc.2004.2.

[24] S. Amershi *et al.*, "Software Engineering for Machine Learning: A Case Study," *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, May 2019, doi: https://doi.org/10.1109/icse-seip.2019.00042.

[25] D. Sculley *et al.*, "Hidden Technical Debt in Machine Learning Systems," *Neural Information Processing Systems*, 2015. https://papers.nips.cc/paper_files/paper/2015/hash/86df7dcfd896fcaf2674f757a2463eba-Abstract.html

[26] V. K. R. Mittamidi, "AI/ML Powered Intelligent Root Cause Analysis and Automated Remediation for Multi-System Data Integrity Issues," *International Journal of AI, BigData, Computational and Management Studies*, vol. 6, pp. 133–141, 2025, doi: https://doi.org/10.63282/3050-9416.ijaibdcms-v6i4p115.

[27] "Security," *cabernets*. https://kubernetes.io/docs/concepts/security/

[28] ISO, "ISO/IEC FDIS 25010," *ISO*, 2023. https://www.iso.org/standard/78176.html

[29] R. Kaushal, K. G. Shojania, and D. W. Bates, "Effects of Computerized Physician Order Entry and Clinical Decision Support Systems on Medication Safety," *Archives of Internal Medicine*, vol. 163, no. 12, p. 1409, Jun. 2003, doi: https://doi.org/10.1001/archinte.163.12.1409.

[30] Open Telemetry Specification 1.43.0, "Open Telemetry Specification 1.43.0," *Open Telemetry*, 2019. https://opentelemetry.io/docs/specs/otel/

[31] "Introduction | Open Policy Agent," *Openpolicyagent.org*, 2025. https://openpolicyagent.org/docs

[32] Horizontal Pod Auto scaling, "Horizontal Pod Auto scaling," *cabernets*, Nov. 23, 2025. https://kubernetes.io/docs/concepts/workloads/autoscaling/horizontal-pod-autoscale/

[33] M. Nygard, "Release It!: Design and Deploy Production-Ready Software," *Torrossa.com*, pp. 1–376, Mar. 2025, Accessed: Mar. 19, 2025. [Online]. Available: https://www.torrossa.com/gs/resourceProxy?an=5241265&publisher=FZP531

[34] "Beyer, B., Jones, C., Petoff, J., and Murphy, N.R. (2016). Site Reliability Engineering: How Google Runs Production Systems. O'Reilly Media. - References - Scientific Research Publishing," *Scirp.org*, 2016. https://www.scirp.org/reference/referencespapers?referenceid=4019415

[35] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," *proceedings.mlr.press*, Apr. 10, 2017. https://proceedings.mlr.press/v54/mcmahan17a.html.