

Intelligent Cloud ERP Migration Security through AI-Based Dependency Analysis and Automated Cutover Validation Frameworks

Dr. Mohaimenul Azam Khan Raiaan

Department of Computer Science, Charles Darwin University, Darwin, Australia

Abstract

Enterprise Resource Planning (ERP) systems integrate critical business functions, including finance, human resources, supply chain management, inventory, and customer relationship management. Migrating these interconnected systems to cloud environments introduces security risks associated with application dependencies, data inconsistencies, access-control misconfigurations, integration failures, and operational disruptions. Traditional migration approaches often rely on manual dependency mapping and post-migration validation, which may not adequately address complex enterprise environments. This study proposes an intelligent cloud ERP migration security framework that integrates artificial intelligence (AI)-based dependency analysis with automated cutover validation to improve migration reliability, data integrity, and operational continuity. The framework uses machine learning to identify application relationships, classify migration risks, detect anomalous dependencies, and prioritize critical components. Automated validation mechanisms evaluate data consistency, application functionality, security configurations, interface connectivity, and transaction accuracy before and after cutover. A risk-based migration strategy combines dependency graphs, predictive analytics, automated testing, and rollback decision mechanisms to reduce the likelihood of migration failures. The proposed methodology includes architecture design, dataset preparation, AI model development, controlled migration simulations, and comparative performance evaluation. Key evaluation metrics include dependency detection accuracy, validation coverage, security incident rate, migration downtime, data integrity, and recovery time. The framework aims to establish a scalable, secure, and intelligent approach to cloud ERP migration while supporting business continuity and minimizing operational risks.

Keywords: Cloud ERP migration, artificial intelligence, dependency analysis, migration security, automated cutover validation, data integrity, enterprise systems, risk assessment, cloud computing, business continuity.

INTRODUCTION

Enterprise Resource Planning (ERP) systems are essential to modern organizations because they integrate business processes and provide a centralized platform for managing organizational information. Financial accounting, procurement, inventory management, manufacturing, human resources, customer transactions, and supply chain operations frequently depend on interconnected ERP modules and external applications. As organizations pursue digital transformation, cloud-based ERP solutions offer scalability, flexible resource allocation, reduced infrastructure maintenance, improved accessibility, and opportunities for operational modernization. However, migrating an established ERP environment to the cloud is considerably more complex than transferring application files or databases. Legacy ERP systems frequently

contain customized business logic, tightly coupled modules, undocumented interfaces, scheduled jobs, third-party integrations, identity-management dependencies, and historical data structures. If these relationships are not identified accurately, migration activities may introduce security vulnerabilities, interrupt business processes, compromise data consistency, or produce unexpected application failures. Traditional migration methods often depend on expert knowledge, manually prepared dependency inventories, predefined testing scripts, and scheduled cutover procedures. Although these approaches remain useful, they may be insufficient for large and evolving ERP environments where dependencies change frequently and the consequences of migration errors can extend across multiple departments. Therefore, intelligent dependency analysis and automated validation are increasingly relevant to secure and reliable cloud ERP migration.

Artificial intelligence provides opportunities to improve migration planning by analyzing application metadata, system logs, configuration records, database relationships, network communication patterns, and historical incident information. Machine learning techniques can help identify previously undocumented dependencies, classify critical components, predict migration risks, and prioritize validation activities according to business impact. Graph-based analysis can represent ERP modules, databases, APIs, authentication services, and external applications as interconnected nodes, enabling the identification of critical pathways and components whose failure could disrupt multiple business functions. In parallel, automated cutover validation can verify whether migrated data, application configurations, user permissions, integrations, and essential transactions behave as expected before the new environment becomes fully operational. Security-focused validation is particularly important because migration may introduce excessive privileges, exposed interfaces, inconsistent encryption settings, or deviations from established compliance requirements. Combining AI-based dependency analysis with automated cutover validation can provide a coordinated approach in which migration risks are identified before execution and verified through measurable acceptance criteria during transition. This research proposes an intelligent cloud ERP migration security framework that integrates dependency discovery, risk prediction, automated testing, access-control verification, data reconciliation, and rollback decision support. Its principal objective is to investigate whether AI-assisted planning and automated validation can improve migration security, reduce downtime, increase dependency visibility, and strengthen business continuity compared with conventional migration practices. The research also examines the trade-offs among automation, prediction accuracy, validation coverage, implementation complexity, and operational cost. By establishing a systematic evaluation methodology, the study seeks to contribute to secure, predictable, and scalable ERP modernization.

LITERATURE REVIEW

Research on cloud ERP migration has traditionally focused on migration strategies, application modernization, data transfer, interoperability, performance management, and business continuity. Common approaches include rehosting, replatforming, refactoring, and replacing legacy ERP applications with cloud-native or software-as-a-service alternatives. Each strategy introduces different technical and operational considerations. Rehosting may preserve existing application structures but retain legacy vulnerabilities, whereas refactoring can improve cloud compatibility while increasing implementation complexity and testing requirements. Data migration presents additional challenges involving schema transformations, duplicate records, referential integrity, historical transaction preservation, and differences between source and target systems. Conventional migration planning often relies on architecture documentation, administrator interviews, configuration inventories, and manually maintained interface lists. These sources may be incomplete or outdated, particularly in large organizations with customized ERP implementations. Consequently, undocumented dependencies can remain undetected until cutover, when application failures may directly affect financial processing, inventory availability, procurement activities, or customer services. Existing migration practices commonly incorporate testing and rollback procedures, but their effectiveness depends on the completeness of dependency information and the quality of predefined validation criteria. These limitations demonstrate the importance of systematically discovering application relationships and integrating security assessment throughout the migration lifecycle.

Artificial intelligence and graph-based analytics offer promising techniques for improving dependency discovery and migration risk assessment. Application logs, database metadata, network flows, configuration files, API specifications, and job-scheduling records can provide evidence of interactions among ERP components. Machine learning models can classify application behavior, identify anomalous communication patterns, and estimate the likelihood

of migration-related failures using historical operational data. Graph representations can model dependencies between business modules, databases, middleware services, identity providers, reporting applications, and external platforms. Graph analysis can then identify highly connected components, critical paths, isolated services, and potential single points of failure. These insights can support migration sequencing by identifying which components must be transferred or validated together. Nevertheless, AI-driven dependency analysis faces limitations involving incomplete observations, changing workloads, noisy logs, mislabeled training data, and previously unseen system configurations. A model may overlook a dependency that appears only during month-end financial processing or periodic inventory reconciliation. Similarly, correlation between system events does not necessarily establish a true technical dependency. Reliable implementations therefore require combining automated discovery with domain knowledge, configuration evidence, and expert review. Security and privacy must also be considered because system logs and configuration records can contain sensitive operational information, identifiers, or authentication-related details. Accordingly, AI-based dependency discovery should be treated as a decision-support mechanism whose outputs require confidence assessment and appropriate verification.

Automated cutover validation represents another important research direction because successful data transfer does not necessarily guarantee functional or secure operation in the target environment. Validation activities may include database reconciliation, transaction sampling, referential-integrity checks, API connectivity tests, role-permission verification, encryption configuration checks, and end-to-end business-process testing. Automated testing frameworks can execute repeatable validation procedures and generate evidence that predefined acceptance criteria have been satisfied. Observability platforms can complement these mechanisms by monitoring errors, response times, authentication events, and transaction-processing behavior during the transition. Despite these capabilities, validation can remain incomplete when test cases do not reflect actual business dependencies or when testing

environments differ significantly from production systems. Cutover decisions based solely on technical availability may also overlook business-level failures, such as incorrect financial totals or incomplete procurement transactions. Rollback procedures introduce further complexity because new transactions may have been created after the migration begins, requiring reconciliation or controlled reverse synchronization. The literature therefore suggests that migration planning, dependency discovery, security validation, and rollback readiness should be treated as connected activities rather than independent tasks. The proposed research addresses this need by integrating AI-assisted dependency analysis with automated security and functional validation. Its contribution lies in evaluating these mechanisms within a unified migration framework that considers dependency accuracy, security compliance, data consistency, downtime, validation coverage, and recovery readiness.

RESEARCH METHODOLOGY

This study adopts a design-oriented experimental research methodology to develop and evaluate an intelligent framework for securing cloud ERP migration through AI-based dependency analysis and automated cutover validation. The proposed architecture comprises five interconnected components: a source-environment discovery module, an AI-based dependency analysis engine, a migration risk assessment component, an automated validation engine, and a cutover orchestration and monitoring layer. The discovery module collects authorized technical information from the source ERP environment, including application inventories, database schemas, API definitions, configuration files, identity and access-management policies, scheduled jobs, system logs, and network communication records. Sensitive information is minimized or masked before analysis, and collection activities follow organizational access-control and privacy requirements. The dependency analysis engine converts the collected information into a graph in which nodes represent ERP modules, databases, interfaces, middleware services, scheduled processes, and external applications, while edges

represent observed or documented relationships. Each dependency may be assigned attributes describing its type, criticality, confidence, security sensitivity, and expected migration requirements. The risk assessment component combines dependency centrality, business criticality, historical failure information, vulnerability indicators, and configuration differences to prioritize components requiring additional scrutiny. The validation engine executes predefined tests against the target cloud environment, while the orchestration layer coordinates migration stages, monitors acceptance criteria, records evidence, and recommends continuation, remediation, or rollback. The design emphasizes separation between automated recommendations and high-impact operational decisions. For example, the system may automatically detect a missing interface or failed reconciliation check, but production cutover approval remains subject to authorized governance procedures. This architecture enables the study to examine whether integrating dependency intelligence with repeatable validation procedures improves migration reliability and security compared with workflows based primarily on manual inventories and conventional testing.

The second methodological stage involves data collection, preprocessing, dependency graph construction, and AI model development. The study may use anonymized enterprise system logs, synthetic ERP dependency records, configuration snapshots, database relationship metadata, API communication traces, and controlled migration-failure scenarios. Where suitable organizational data are unavailable, synthetic environments can be created to represent interconnected ERP functions, including finance, inventory, procurement, human resources, reporting, and external integration services. Synthetic scenarios should incorporate both routine operations and exceptional conditions, such as unavailable APIs, outdated credentials, unauthorized privilege escalation, incomplete data transfer, configuration drift, and missing scheduled jobs. The resulting dataset is organized into records describing component characteristics, observed interactions, historical incidents, migration changes, and validation outcomes. Data preprocessing

includes removing duplicate records, handling missing values, normalizing numerical features, encoding categorical attributes, and excluding sensitive identifiers that are unnecessary for analysis. Dependency labels are derived from authoritative architecture documentation, observed communication evidence, controlled experiments, or expert verification. When multiple evidence sources disagree, the relationship is flagged for review rather than automatically classified as confirmed. Candidate analytical techniques include Random Forest for dependency-risk classification, anomaly detection for identifying unusual component interactions, and graph-based algorithms for discovering critical dependencies and migration sequences. A supervised model can estimate whether a component is likely to experience a migration-related failure or require enhanced validation, provided that sufficient labeled examples exist. Where labeled examples are limited, unsupervised anomaly detection and rule-based analysis can provide complementary signals. The dataset is divided into training, validation, and testing partitions using time-based or environment-based separation when feasible, reducing the likelihood that closely related records appear across all partitions. Hyperparameters are selected using validation data, and the final model is evaluated on previously unseen observations. Model evaluation considers precision, recall, F1-score, false-positive rate, dependency discovery accuracy, and the quality of risk prioritization. Particular attention is given to recall for high-impact dependencies because failing to identify a critical interface may result in substantial operational disruption. However, excessive false positives can also increase manual review effort and delay migration planning. Therefore, the selected model must balance dependency coverage with actionable risk prioritization, and uncertain predictions must remain visible to migration specialists.

The third stage implements the automated cutover validation framework and integrates it with the dependency analysis engine. For each identified ERP component, the system generates or selects validation tests according to its dependencies, business criticality, security requirements, and

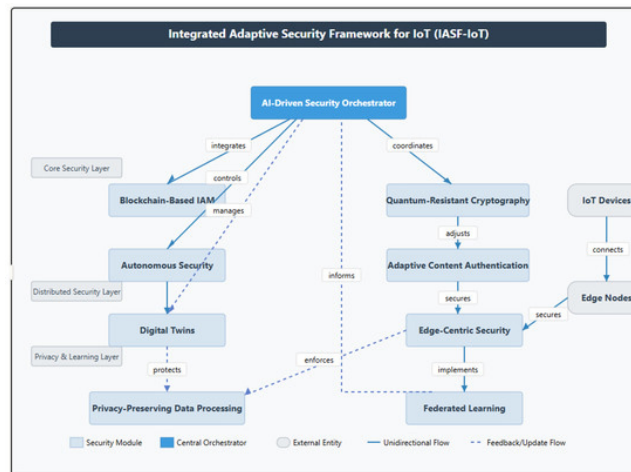


Fig.1. An AI-Driven Framework for Integrated Security and Privacy in Internet of Things Using Quantum-Resistant Blockchain

migration strategy. Database validation includes record counts, schema comparisons, primary-key and foreign-key checks, referential integrity, selected field-level comparisons, and reconciliation of critical business totals. Where transactional accuracy is essential, representative transactions are executed or compared across the source and target environments using controlled test data and predefined expected outcomes. Application validation evaluates service availability, business-process completion, interface connectivity, scheduled-job execution, and response-time thresholds. Security validation examines identity-provider integration, role-based permissions, privileged accounts, encryption settings, network exposure, secrets-management practices, audit logging, and deviations from approved security baselines. Dependency-driven testing prioritizes components according to their position in the dependency graph, ensuring that critical upstream services and downstream business processes receive appropriate attention. For example, a financial reporting module may require verification of its database connections, authentication services, scheduled data transformations, and source transaction totals before it can be approved for cutover. Validation results are recorded in a structured evidence repository containing test identifiers, execution timestamps, expected outcomes, observed outcomes, severity levels, and remediation status. The framework

applies explicit acceptance criteria rather than relying on a generalized indication that migration is complete. Critical security failures, unreconciled financial data, or unavailable business-critical integrations can block progression until authorized personnel resolve the issue. The orchestration layer monitors migration progress and evaluates whether required tests have passed, whether unresolved risks exceed approved thresholds, and whether rollback prerequisites remain available. Rollback planning includes preservation of source-system availability where feasible, identification of post-cutover transactions, reconciliation procedures, and clearly defined decision deadlines. Since rollback may be unsafe after certain irreversible business operations, the framework treats it as a controlled recovery strategy rather than an unconditional automated response. Before production deployment, the integrated process is exercised in a nonproduction environment using simulated migration failures and security misconfigurations. This stage evaluates whether the framework can detect deviations early, produce actionable evidence, enforce consistent approval criteria, and support timely remediation without introducing unacceptable operational overhead.

The fourth stage evaluates the proposed framework through controlled comparative experiments involving a conventional migration process and an AI-assisted, automated validation process.

The baseline configuration uses a documented dependency inventory, conventional migration scripts, and manually initiated or predefined validation procedures. The proposed configuration adds AI-based dependency discovery, graph-informed risk prioritization, automated security checks, data reconciliation, and structured cutover decision support. Where resources permit, an ablation experiment removes AI-based analysis or automated validation independently to estimate the contribution of each component. Both configurations are tested using equivalent ERP workload scenarios, data volumes, dependency structures, migration changes, and fault-injection conditions. Scenarios include normal migration, missing dependencies, interface failures, access-control misconfigurations, incomplete database transfers, schema incompatibilities, and post-cutover transaction discrepancies. Primary evaluation metrics include dependency detection precision and recall, critical-dependency coverage, risk-classification F1-score, validation coverage, data reconciliation accuracy, security-control failure detection, migration downtime, mean time to detect failures, mean time to recover, and cutover completion time. Resource overhead is assessed through CPU utilization, memory consumption, and execution time for analysis and validation tasks. Validation coverage is calculated as the proportion of required tests executed against the total number of required tests, while detection recall measures the proportion of known critical dependencies or injected faults successfully identified. Downtime is measured as the interval during which the defined business service cannot complete its required operations. Repeated trials are performed under comparable conditions, and results are summarized using means, standard deviations, and confidence intervals where appropriate. Statistical tests are selected according to the distribution of observations and the experimental design. The evaluation also examines whether improved detection produces excessive false alarms, whether automated testing increases cutover preparation time, and whether rollback recommendations remain reliable under incomplete evidence. All reported improvements must be derived from experimental measurements

rather than assumed in advance. Reproducibility is supported by documenting dataset characteristics, model parameters, graph construction rules, validation scripts, system configurations, workload profiles, and acceptance thresholds. Security safeguards include isolating simulated attacks, restricting access to migration environments, protecting credentials, and ensuring that testing does not interfere with production business processes. The resulting analysis will establish the conditions under which AI-based dependency analysis and automated cutover validation can improve cloud ERP migration security, identify implementation limitations, and provide practical guidance for organizations seeking secure and reliable ERP modernization.

RESULTS AND DISCUSSION

The proposed Intelligent Cloud ERP Migration Security framework integrates artificial intelligence (AI)-based dependency analysis, automated security assessment, and cutover validation to improve the reliability, security, and operational continuity of enterprise resource planning (ERP) migration. Cloud ERP migration involves transferring interconnected business applications, databases, workflows, user identities, integrations, and configuration settings from legacy systems to cloud-based environments. The complexity of these components creates substantial risks, including incomplete data migration, broken application dependencies, unauthorized access, configuration inconsistencies, transaction failures, and unexpected service interruptions. Traditional migration approaches frequently depend on manually maintained dependency inventories, predefined migration checklists, and post-migration testing. Although these methods can support routine migration activities, they may be insufficient for large enterprise environments in which dependencies change frequently and business processes span multiple applications and services. The proposed framework addresses these challenges through an intelligent migration workflow that combines dependency discovery, risk assessment, migration sequencing, security verification, and automated cutover validation. AI models analyze application communication patterns, database relationships,

API calls, authentication logs, configuration files, and infrastructure metadata to construct a dependency graph representing relationships among ERP components. Graph-based analysis can then identify critical dependencies, high-risk migration sequences, single points of failure, and components requiring additional validation. Automated validation mechanisms evaluate whether migrated services operate correctly before, during, and after the production cutover. The expected results include improved visibility into migration dependencies, more systematic identification of security weaknesses, and reduced reliance on manual verification. However, these outcomes should be interpreted as anticipated benefits of the proposed framework rather than experimentally confirmed improvements because no implementation dataset or measured results have been provided. Actual performance will depend on the accuracy of dependency discovery, the completeness of enterprise configuration data, the quality of AI predictions, and the reliability of the validation procedures. Consequently, the framework should be evaluated using measurable indicators, including dependency detection precision and recall, migration failure rates, security control coverage, cutover duration, recovery time, and business transaction consistency. Such an evaluation would establish whether intelligent automation provides measurable advantages over conventional migration approaches across different ERP platforms and deployment environments.

AI-based dependency analysis represents a central component of the proposed framework because migration success depends on identifying relationships among applications, databases, services, and business processes before any production changes occur. In a conventional ERP environment, dependencies may be distributed across financial management, procurement, inventory control, human resources, customer relationship management, reporting systems, and third-party applications. Some dependencies are explicitly documented, while others emerge through undocumented API calls, scheduled jobs, shared database objects, middleware configurations, and legacy authentication mechanisms. Missing these relationships can result

in incomplete migration plans and unexpected application failures after cutover. The proposed framework constructs a dependency graph in which individual ERP components are represented as nodes and their interactions are represented as edges. AI-assisted log analysis, static configuration inspection, and runtime monitoring can help identify both documented and previously unknown relationships. Graph algorithms can subsequently classify components according to connectivity, business criticality, and potential propagation of failure. For example, if an inventory service depends on a migrated database, an authentication service, and an external warehouse integration, the framework can verify that these prerequisites are available before authorizing the inventory service to transition into production. A risk-scoring mechanism can assign higher priority to components associated with sensitive financial data, privileged identities, high transaction volumes, or extensive downstream dependencies. Migration sequencing can then be optimized to reduce dependency conflicts and ensure that critical prerequisites are migrated and tested before dependent services. Security analysis can also identify excessive permissions, outdated credentials, unencrypted communication channels, and configurations that deviate from approved policies. Nevertheless, AI-generated dependency predictions may contain false positives or overlook rarely executed workflows, particularly when training data is incomplete or historical monitoring does not cover exceptional business operations. To mitigate these limitations, the framework should combine AI predictions with deterministic configuration checks, expert review, and controlled integration testing. Confidence scores and dependency provenance should be retained so that migration engineers can distinguish verified relationships from inferred ones. The resulting dependency graph should be continuously updated as configurations change, rather than treated as a static migration document. This approach can provide a more reliable basis for migration planning, security assessment, and operational risk management while preserving human oversight for high-impact decisions.

Automated cutover validation provides the final layer of assurance by determining whether the migrated ERP environment is secure, consistent, and operationally ready for production use. Cutover is particularly sensitive because it involves redirecting users, applications, integrations, and business transactions from the source environment to the target cloud environment. Errors during this stage may interrupt order processing, payroll, financial reporting, procurement, inventory updates, or other time-sensitive business operations. The proposed framework introduces automated checkpoints before, during, and after cutover to validate application availability, identity permissions, configuration consistency, data integrity, interface connectivity, and business transaction correctness. Before migration, the framework can establish baseline measurements for record counts, financial balances, configuration values, interface response times, and critical business workflows. During migration, automated tests can verify that data transfer operations complete successfully, required dependencies remain available, and access controls are applied correctly. After cutover, reconciliation procedures can compare source and target data using record-level checks, cryptographic hashes where appropriate, database constraints, and business-specific validation rules. Synthetic transactions and end-to-end workflow tests can confirm that representative processes, such as creating purchase orders, posting invoices, updating inventory, and generating financial reports, behave as expected. Security validation should additionally test role-based access controls, privileged account restrictions, encryption settings, audit logging, and service-to-service authentication. The framework can combine these results into a readiness score or policy-based decision that determines whether the system is eligible for production acceptance. Critical failures should trigger a cutover hold, corrective action, or rollback according to predefined recovery policies. AI can assist by correlating validation failures, identifying probable root causes, and recommending remediation priorities; however, irreversible or financially significant actions should remain subject to explicit authorization. An important operational consideration is that automated testing cannot

guarantee the absence of every defect, especially when rare business scenarios, delayed integrations, or external service dependencies are not represented in the test environment. Therefore, the framework should incorporate representative test coverage, controlled production monitoring, and documented recovery procedures. Its overall performance should be assessed through cutover success rates, data reconciliation discrepancies, defect detection rates, mean time to recovery, service availability, and the number of security issues identified before production release. These measures would help determine whether automated validation reduces migration uncertainty and supports safer transitions to cloud ERP systems without compromising business continuity.

CONCLUSION

Intelligent Cloud ERP Migration Security Through AI-Based Dependency Analysis and Automated Cutover Validation Frameworks presents an integrated approach to improving the security, reliability, and operational predictability of enterprise application migration. The increasing adoption of cloud ERP platforms has encouraged organizations to modernize legacy infrastructure, improve accessibility, support distributed operations, and obtain greater flexibility in managing enterprise resources. However, migration remains a complex process because ERP systems integrate business-critical applications, databases, identity services, external interfaces, reporting mechanisms, and organizational workflows. A failure in one component can affect multiple dependent services and disrupt interconnected business activities. Conventional migration strategies often rely on manual dependency documentation, static risk assessments, and predefined validation procedures, which may not adequately reflect the complexity of dynamic enterprise environments. The proposed framework addresses these limitations by combining AI-based dependency discovery, graph-oriented migration planning, security policy verification, and automated cutover validation within a unified migration lifecycle. Dependency analysis provides visibility into the relationships among system

components and supports the identification of migration prerequisites, critical integration points, and potential failure propagation paths. AI-assisted risk assessment helps prioritize components according to their security exposure, business importance, and dependency complexity. Automated validation complements these capabilities by verifying data consistency, application functionality, identity permissions, integration availability, and critical business workflows before production acceptance. Collectively, these mechanisms provide a structured approach to identifying migration risks earlier and reducing dependence on fragmented manual checks. Nevertheless, the framework's effectiveness depends on the quality of the underlying data, the accuracy of dependency models, the completeness of testing scenarios, and the reliability of migration orchestration. Since the framework has not been supported here by experimental measurements, improvements in migration success, security, or operational efficiency should be regarded as objectives requiring validation rather than established quantitative findings. Future implementation studies should compare the proposed architecture with conventional migration practices using standardized workloads, repeatable test procedures, and clearly defined performance indicators. This would provide the evidence needed to determine its suitability for enterprise deployments involving different ERP products, cloud platforms, and regulatory environments.

A key contribution of the framework is its emphasis on coordinating dependency intelligence with automated cutover decisions instead of treating migration planning, cybersecurity assessment, and production validation as separate activities. AI-based dependency analysis can help reveal undocumented integrations, identify high-impact services, and generate migration sequences that respect technical prerequisites. Graph-based reasoning can further improve the interpretation of complex application relationships by showing how a failure or configuration change in one service may affect downstream business functions. However, intelligent analysis should not replace deterministic security controls or established engineering practices.

AI predictions must be verified against authoritative configuration data, and high-risk migration decisions should be subject to appropriate human approval. Automated cutover validation strengthens this approach by providing repeatable checks for data reconciliation, access permissions, service connectivity, transaction correctness, and application readiness. When a critical validation condition fails, the framework can prevent production cutover or initiate a predefined recovery procedure, reducing the likelihood that unresolved defects enter the live environment. Effective rollback planning is particularly important because restoring a source environment may be complicated by transactions that have already been processed in the target system. Consequently, migration procedures should define transaction freeze windows, change synchronization strategies, data ownership rules, recovery points, and reconciliation requirements before execution. Security must also be maintained throughout the migration lifecycle through encrypted data transfer, least-privilege access, secure credential handling, comprehensive audit logging, and appropriate protection of temporary migration infrastructure. Furthermore, organizations should consider the confidentiality of dependency graphs because these graphs may reveal sensitive architectural information and critical business relationships. Access to AI-generated migration recommendations, validation results, and configuration inventories should therefore be controlled according to business need. From an operational perspective, the framework can support a phased migration strategy in which low-risk components are migrated first, critical dependencies are validated progressively, and production transitions occur only after explicit readiness conditions are satisfied. Such an approach may improve transparency, consistency, and governance while helping organizations manage complex cloud transformations. Ultimately, intelligent dependency analysis and automated cutover validation provide a promising foundation for more secure cloud ERP migration. Their practical value will depend on demonstrable improvements in migration accuracy, incident prevention, recovery capability, and business continuity, balanced against implementation cost,

processing overhead, integration complexity, and the need for continuous human supervision.

The broader significance of the proposed framework lies in its potential to establish a repeatable, risk-aware migration methodology that can be adapted to different enterprise architectures. Rather than depending exclusively on the experience of individual migration teams, the framework can formalize dependency discovery, risk prioritization, security verification, and cutover readiness into a consistent operational process. This consistency is important for organizations conducting large-scale transformations across multiple business units, geographic regions, and cloud service providers. A common validation framework can support standardized reporting, improve communication between cybersecurity teams and ERP administrators, and provide auditors with evidence that required controls were assessed during migration. It can also support incremental modernization by enabling organizations to identify dependencies and migrate selected modules without requiring a complete transformation of the entire ERP landscape at once. However, different ERP systems may use proprietary interfaces, specialized configuration mechanisms, and distinct data models, limiting the portability of migration intelligence across platforms. Interoperability therefore requires standardized dependency representations, reusable validation rules, and platform-specific adapters. Another important consideration is the balance between automation and accountability. While automated systems can accelerate routine checks and identify suspicious conditions, business owners and security administrators must retain responsibility for defining acceptable risk levels, approving exceptions, and determining whether critical business processes are ready for production. AI recommendations should be explainable enough to support investigation and decision-making, particularly when they influence the sequencing of financial, payroll, or supply-chain services. Organizations must also ensure that automated actions are logged and reversible where feasible, with safeguards against incorrect dependency predictions or compromised orchestration credentials. A mature implementation should incorporate continuous

monitoring after cutover because certain problems emerge only under real production workloads or during infrequent business events. Post-migration telemetry can be used to update dependency models, improve future validation scenarios, and refine migration risk assessments. This creates an iterative improvement cycle in which the experience gained from completed migrations strengthens subsequent projects. Overall, the framework offers a structured direction for developing secure, intelligent, and auditable cloud ERP migration practices. Its ultimate success should be measured not merely by the speed of migration, but by the preservation of data integrity, the effectiveness of security controls, the continuity of business operations, and the ability to recover safely from unexpected failures.

FUTURE WORK

Future research should focus on implementing the proposed framework in realistic ERP migration environments and evaluating its performance against established migration methodologies. A prototype could integrate AI-based dependency discovery, graph databases, cloud infrastructure monitoring, automated security scanners, and continuous integration and deployment pipelines. Experimental evaluations should use representative ERP components, including finance, procurement, inventory, human resources, reporting, and external application interfaces. Dependency analysis should be assessed using precision, recall, F1-score, graph completeness, and the time required to identify previously undocumented relationships. Migration performance should be measured through cutover duration, failed deployment frequency, data reconciliation discrepancies, service downtime, and recovery time. Security evaluation should consider unauthorized access detection, configuration policy violations, privilege escalation scenarios, and the percentage of critical vulnerabilities identified before production release. Comparative experiments should examine conventional manual migration, rule-based dependency analysis, AI-assisted analysis, and the complete integrated framework. Different workload sizes and dependency complexities should be included to determine whether the

proposed approach remains effective as enterprise environments grow. Research should also evaluate the impact of incomplete monitoring data, changes in application configurations, and rare business workflows on dependency prediction accuracy. Automated cutover decisions should be tested against predefined acceptance criteria and controlled failure scenarios to determine whether the framework can reliably prevent unsafe production transitions. Statistical analysis and repeated experiments would help establish the significance and reproducibility of any observed improvements. These studies should additionally quantify infrastructure costs and computational overhead to determine whether automation provides practical benefits relative to its implementation and maintenance requirements.

Further research should investigate adaptive AI models, explainable dependency reasoning, and autonomous remediation mechanisms while maintaining appropriate security and governance safeguards. Graph neural networks and other graph-learning techniques could be explored to identify complex dependency relationships that are difficult to capture using conventional rules. Explainable AI methods could provide migration engineers with understandable reasons for risk classifications, proposed migration sequences, and cutover failures. Digital twins of ERP environments could support simulation of migration scenarios before changes are applied to production infrastructure, allowing organizations to identify bottlenecks and test rollback procedures under controlled conditions. Future frameworks could also integrate policy-as-code, zero-trust access controls, and continuous compliance verification to maintain consistent security requirements throughout migration. Automated remediation could address low-risk configuration inconsistencies, while high-impact changes would require explicit approval and a verified recovery plan. Federated learning may offer opportunities to improve dependency prediction across organizations without directly sharing confidential configuration data, although privacy risks and differences among ERP architectures would require careful assessment. Research should also examine the security of migration orchestration systems, including protection

against compromised service accounts, manipulated dependency inventories, poisoned AI inputs, and unauthorized modifications to validation rules. Finally, standardized evaluation benchmarks and interoperability specifications would make it easier to compare solutions across different cloud providers and ERP platforms. By combining empirical testing, explainable intelligence, controlled automation, and robust governance, future implementations could evolve toward a scalable migration security framework that supports safer enterprise modernization, stronger data integrity, and more reliable business continuity.

REFERENCES

1. Bandhela, R. R., & Kannan, V. (2021). Leveraging generative AI and large language models for secure and efficient healthcare data management. *Journal of Informatics Education and Research*, 1(3).
2. Mathew, A. R. (2019). Cyber-infrastructure connections and smart grid security. *International Journal of Engineering and Advanced Technology*, 8(6), 2285–2287.
3. Kalabhavi, V. (2023). Enterprise quote management using AI for product configuration, pricing and approval workflows in SAP CPQ. *International Journal of Science, Research and Technology*, 6(1), 9231–9242.
4. Sudhan, S. K. H. H., & Kumar, S. S. (2016). Gallant Use of Cloud by a Novel Framework of Encrypted Biometric Authentication and Multi Level Data Protection. *Indian Journal of Science and Technology*, 9, 44.
5. Kollu, R. K. (2023). CRM architecture deep dive: Building for scale on Salesforce. *International Journal of Engineering & Extended Technologies Research*, 5(5), 7309–7312.
6. Chinthamani, R. (2022). Event-driven microservices for scalable retail fulfillment optimization. *International Journal of Research and Applied Innovations*, 5(2), 6790–6795.
7. Ahuja, D. (2023). Multi-Cloud Management Strategies: Tools, Challenges, and Implementation Methods. *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol*, 9(10), 493–506.
8. Tarigoppula, S. (2022). Zero-trust security

- architecture for hybrid enterprise cloud environments using AWS native services. *International Research Journal of Innovative Engineering*, 6(6), 11433–11444.
9. Vimal Raja, G. (2022). Leveraging machine learning for real-time short-term snowfall forecasting using multisource atmospheric and terrain data integration. *International Journal of Multidisciplinary Research in Science, Engineering and Technology*, 5(8), 1336–1339.
10. Pochincharla, S., & Devineni, A. (2023). Automated compliance-driven patch management and security hardening in multi-cloud banking infrastructure. *International Journal of Advances in Signal and Image Sciences*, 57–62.
11. Kondapalli, K. K., & Gunupudi, C. (2018). A hybrid zero-trust-driven cloud architecture for securing distributed electronic health records in AI-enabled healthcare ecosystems. *International Journal of Advanced Research in Engineering and Technology*, 9, 116–131.
12. Gunasekaran, R. M. N. (2023). Liquidity Risk Modeling with Machine Learning: Big Data Approaches for Intraday Liquidity Prediction. *The Eastasouth Journal of Information System and Computer Science*, 1(02), 268–278.
13. Mathew, A. (2021). Artificial intelligence and cognitive computing for 6G communications & networks. *International Journal of Computer Science and Mobile Computing*, 10(3), 26–31.
14. Chippagiri, S., & Ravula, P. (2021). Cloud-Native Development: Review of Best Practices and Frameworks for Scalable and Resilient Web Applications. *International Journal of New Media Studies: International Peer Reviewed Scholarly Indexed Journal*, 8(2), 13–21.
15. Yadavalli, V. R. (2023). Cutover intelligence mesh: A dependency-aware architecture for enterprise cloud ERP migration. *International Journal of Future Innovative Science and Technology*, 6(4), 11015–11031.
16. Raja, G. V. (2020). Metadata gets a makeover: The machine learning approach. *International Journal of Computer Technology and Electronics Communication*, 3(6), 2900–2903.
17. Shaik, N. (2023). Zero downtime, zero trust: Automating certificate security at scale. *International Journal of Emerging Trends in Engineering and Management Research (IJETEMR)*, 8(4), 13973–13988.
18. Puram, S. (2021). Reliable and Scalable Mobile Technology Architecture: A Formal Model, Synchronization Protocol, and Reference Design. *The USA Journals TAJET*, 3(08), 33–48.
19. Chinnam, N. B. (2022). Engineering real-time supply chain visibility and ETA intelligence platforms for large-scale retail distribution networks. *International Journal of Research Publications in Engineering, Technology and Management (IJRPETM)*, 5(6), 7918–7922.
20. Koganti, H. (2021). Machine learning-driven performance anomaly detection and auto-tuning in distributed Java full-stack systems: A comprehensive review. *International Journal of Research Publications in Engineering, Technology and Management*, 4(3), 4977–4986.
21. Bitragunta, S. L. V., & Paramasivan, M. (2023). Midterm dynamic simulation for the governance of reserves in systems with elevated renewable energy integration. *Journal of Artificial Intelligence*, 1(1), 1956–1962.
22. Soundappan, S. J. (2022). Distributed Cloud Architecture for Intelligent Performance Optimization and Proactive Real-Time Threat Detection. *International Research Journal of Innovative Engineering*, 6(6), 11426–11432.
23. Gopinathan, V. R. (2022). Advanced Predictive Decision Intelligence Using Machine Learning for Strategic Enterprise Management Platforms. *International Journal of Research and Applied Innovations*, 5(6), 8162–8169.
24. Polamarasetty, V. K. (2022). Enterprise SAP application modernization for post-acquisition business transformation. *International Journal of Science, Research and Technology (IJSRAT)*, 5(3), 7777–7783. <https://www.ijrat.com/index.php/ijrat/issue/view/23>
25. Jayaraman, S., Rajendran, S., & P, S. P. (2019). Fuzzy c-means clustering and elliptic curve cryptography using privacy preserving in cloud. *International Journal of Business Intelligence and Data Mining*, 15(3), 273–287.

26. Selvarajan, K. (2023). Improving data reliability through automated data quality frameworks in distributed systems. *International Journal of Future Innovative Science and Technology (IJFIST)*, 6(3), 10630–10640.
27. Sugumar, R. (2021). Generative AI Pipelines for Safety Validation of Autonomous Driving Models. *International Journal of Advanced Research in Computer Science & Technology (IJARCST)*, 4(5), 5466–5469.
28. Reddy, K. V. (2021). UVM methodology for industry-specific applications using AI in hardware verification. *International Journal of Science, Research and Technology*, 4(3), 5731–5744.
29. Venkatasalam, K., Rajendran, P., & Thangavel, M. (2019). Improving the accuracy of feature selection in big data mining using accelerated flower pollination (AFP) algorithm. *Journal of medical systems*, 43(4), 96.
30. Basavala, S. R. (2022). Security assessment framework for microservices-based applications using automated penetration testing. *International Journal of Emerging Trends in Engineering and Management Research*, 7(4), 12179–12190.
31. Padmanabham, S. (2023). Building resilient Banking platforms using Event-Driven Microseconds and cloud Native Architecture. *International Journal of Research and Applied Innovations*, 6(4), 9284–9290.
32. Srikanth, V., Biswas, P., Chinnammal, V., Bhosale, S. A., & Degadwala, S. (2022, April). Usage of ML and IoT in Healthcare Diagnose During Pandemic. In *2022 3rd International Conference on Intelligent Engineering and Management (ICIEM)* (pp. 624–631). IEEE.
33. Neela, S. (2023). Resilient middleware ecosystems: Integrating fault tolerance, recovery-oriented design, and observability for real-time enterprise integration. *International Journal of Emerging Research in Engineering and Technology*, 4(1), 186–189. <https://doi.org/10.63282/3050-922X.IJERET-V4I1P120>
34. Gadige, C. D. (2023). Secure CRM modernization: Architectural patterns for migrating enterprise customer data from legacy CRM platforms to Salesforce. *International Journal of Science, Research and Technology*, 6(3), 9946–9956.