International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 18(21s) 2026 pp. 442-451
https://doi.org/10.70917/ijcisim-2026-5319
Article

AI-Assisted Test Execution as an Augmentation Layer in Enterprise Quality Engineering

Rejenish Kiran

Independent Researcher, USA

Abstract: Artificial intelligence has introduced new capabilities for enterprise quality engineering, yet the absence of structured governance frameworks limits its practical adoption in regulated software environments. This paper presents the Probabilistic Augmentation and Governance Model (PAGM), a three-tier framework that formally allocates responsibility between AI agents and human testers across autonomous execution, confidence-gated escalation, and human-led verification. PAGM draws on a structured review of literature spanning AI quality assurance, autonomous system testing, UI-resilient automation, explainable AI, and defect prediction to identify governance gaps in existing approaches. The model addresses eight documented AI testing challenges, including interpretability, absence of formal specifications, oracle determination, and dynamic operational environments. Unlike prior approaches that treat AI test execution as a performance optimization problem, PAGM positions governance, traceability, and bounded autonomy as first-class design requirements. Its application is demonstrated through a regulated property insurance release pipeline in which PAGM enables a full quarterly regression cycle within a five-day service level agreement while preserving mandatory human accountability for premium calculation and claims adjudication workflows. Quantitative evidence from the literature supports the model's constituent mechanisms, including a 95% UI change handling rate for contextual recognition methods compared to 40 to 80 percent achieved by conventional tools and AI-based classifiers outperforming classical approaches by 25 to 50 percent across software quality metrics. PAGM provides a governance-ready foundation for organizations seeking to deploy AI-assisted testing at enterprise scale within regulated or auditable environments.
Keywords: AI-Assisted Test Execution, Enterprise Quality Engineering, Intelligent Test Agents, Probabilistic Reasoning, Governance Frameworks, Confidence-Gated Escalation, Bounded Autonomy, Explainable AI

1. Introduction

Artificial intelligence has fundamentally altered how enterprise quality engineering and software testing teams operate. Historically, enterprise software quality relied on manual and deterministic testing processes. As software systems have grown in complexity, AI-driven capabilities have emerged as an advanced augmentation layer over traditional testing models, though effective monitoring and scope governance remain essential to realizing efficiency improvements [1].

With advances in machine learning and natural language processing, intelligent test agents based on probabilistic logic can now interpret requirements, understand application user interfaces, and validate system behavior across diverse and evolving system states with minimal procedural programming. Their deployment has demonstrated potential for increased defect detection rates and reduced regression validation cycles [2]. A fundamental distinction between traditional software systems and AI-based systems is that the former are deterministic and designed to be fault-free, whereas AI-based systems are engineered to perform within an acceptable accuracy envelope under expected operating conditions [1].

Despite this potential, however, the literature suggests that AI testing research has lagged behind. Most research has focused on algorithmic success, while governance infrastructure for enterprise adoption in regulated domains is lacking. The definition of quality engineering has increasingly focused on governance of automated software testing

at scale across the software stack, rather than code writing for procedures [2]. This will, however, require a new model to take into account the capabilities of smart agents and the need for accountability in regulated industries.

This paper presents the Probabilistic Augmentation and Governance Model (PAGM), a three-tier governance framework for guiding and regulating the use of AI in executing tests:

- Tier 1 - Execution: Structured regression checks, sanity checks and deterministic workflow traversals are executed by smart agents that rely on high-confidence pattern detection algorithms.
- Tier 2 - Confidence-Gated Escalation: Agents assign a confidence score to tasks. Those exceeding a set threshold are executed automatically, while those under the threshold are escalated to a human practitioner.
- Tier 3 - Human/Manually Verified: All tasks that require a human tester (for exploratory testing, testing business logic, or when the pipeline needs a full auditable judgment trail).

PAGM is the first framework in the AI testing literature to integrate governance into the design phase, treating accountability boundaries and regulatory compliance as primary design features rather than post-hoc additions. AI-based systems are quality-assured from three perspectives: the nature of the artifact, the process model used, and the quality property targeted [1]. PAGM extends this multi-view logic into an operational governance structure suitable for regulated enterprise deployment.

The remainder of this paper is organized as follows: Section 2 provides background information required to understand PAGM, including the literature analysis method used and the architecture of AI test executors, Section 3 presents the results of the research including the PAGM governance structure and a comparison between AI automation and conventional automation. Finally, Section 4 discusses enterprise applications, research implications, and future directions. Section 5 concludes this paper.

2. Materials and Methods

2.1 Literature Review Methodology

The development of PAGM was grounded in a structured review of peer-reviewed literature drawn from four primary source categories: AI quality assurance and testing frameworks, autonomous and robotic system testing, UI-resilient automation techniques, and AI trustworthiness and governance. Sources were identified through database searches of the ACM Digital Library, IEEE Xplore, arXiv, and institutional repositories using search terms including AI-assisted testing, intelligent test agents, enterprise test automation, AI governance frameworks, explainability in testing, and regression test selection.

Inclusion criteria required that sources: (1) address AI or machine learning techniques in software testing or quality assurance contexts, (2) be published in peer-reviewed venues or as institutionally archived technical reports, and (3) provide empirical findings, systematic reviews, or formal frameworks applicable to enterprise testing environments. Sources were excluded if they addressed testing of AI systems purely from a benchmarking perspective without relevance to enterprise deployment governance.

Nine primary sources were retained for synthesis. These were organized into thematic clusters to identify coverage gaps and inform PAGM's architectural design decisions. Table 1 summarizes the thematic mapping and the specific governance gaps each source leaves unaddressed, which together constitute the justification for PAGM.

Table 3. Thematic Literature Mapping and Identified Governance Gaps

Thematic Area	Key Sources	Identified Gap
AI quality assurance frameworks	Felderer and Ramler [1]; Li et al. [5]	No governance architecture for test execution in enterprise contexts

Autonomous and robotic system testing	Araujo et al. [3]	Limited application to software test execution systems
UI-resilient test automation	Yandrapally et al. [4]	No confidence-gated escalation or governance integration
Explainability in AI systems	Adadi and Berrada [6]	Taxonomy not mapped to testing governance requirements
Regression test selection	Engstrom et al. [7]	No enterprise-scale validation; limited AI integration
Developer behavior and debugging	Beller et al. [8]	Does not address AI augmentation in quality engineering roles
AI-based defect prediction models	Xu et al. [9]	Focuses on prediction accuracy, not governance or traceability
Digital transformation of software development	Laato et al. [2]	Broad scope; no structured model for testing governance

This review confirmed that no existing framework simultaneously addresses bounded autonomy, confidence-gated governance, and regulatory traceability as integrated design requirements for testing AI in enterprises. PAGM was designed specifically to close this gap.

2.2 Architectural Basis for AI Test Executors

AI-assisted test execution differs architecturally from traditional automation: while automation scripts are deterministic instruction sequences, intelligent agents employ machine learning models trained on system behavior, user behavior, and test validity criteria to convert natural language specifications into executable plans, even when procedures are not fully specified in advance.

AI test executor architectures generally comprise three modules: a visual perception module for the application under test; a decision-making module typically based on reinforcement learning or transformer architectures; and an action execution module interacting directly with the application through automated protocols. Because robotic and autonomous systems share this architectural pattern, eight quality assurance challenges apply equally to AI test executors [3]: interpretability, absence of formal specifications, generation of validation data and test inputs, oracle definition for expected outcomes, accuracy and correctness measurement, non-functional properties assessment, self-adaptation and self-learning, and dynamic operational environments.

Intelligent agents leverage probabilistic reasoning to address uncertainties encountered during execution, for example, when multiple UI elements match a target context or when an unexpected dialog appears. Confidence scores enable agents to proceed with element selection or escalate the scenario to a human practitioner, forming the operational foundation for PAGM's Tier 2 confidence-gated escalation. This avoids the brittleness of customary automation scripts made using record-replay, attribute-based finders, XPath selectors, or image processing techniques which are fragile to small variations in the structure of the web page or attributes of web elements [4].

Contextual clue-based recognition utilizes labels, pictures, and patterns of behavior to identify interface elements when no strong semantic match is present. Across 47 test cases spanning 428 steps across five enterprise web applications, 73% of human-specified anchors were directly identified through contextual clue-based recognition, and an additional 26% of cases successfully resolved anchors from surrounding contextual signals [4].

For AI test execution systems, quality validation must encompass both the properties of the system under test and the properties of the learning components themselves. A survey of 195 publications on testing robotic and autonomous systems found that temporal logic variants were the most prevalent approach to requirements specification, while state machines and transition systems were the most common system modeling formalism [3]. Approximately one-third of surveyed papers report that no quantitative measures were available for proposed testing

techniques, complicating the establishment of formal validation requirements for the use of AI in enterprise testing systems.

2.3 PAGM Framework Design

PAGM was designed as a three-tier responsibility-allocation architecture in which the assignment of a validation task to an AI agent or a human practitioner is determined by a combination of task classification, agent confidence scoring, and regulatory designation. The framework's design principles are drawn from the eight quality assurance challenges documented in the literature [1][3] and the trustworthiness dimensions defined for AI systems operating in regulated contexts [5].

The three tiers are operationalized as follows. Tier 1 covers tasks characterized by high structural determinism, well-defined acceptance criteria, and stable UI contexts. Tier 2 is triggered by the agent's confidence scores, which normally need to be below a threshold of 85% (in the enterprise scenarios studied) as determined by human review prior to task initiation. Tier 3 is triggered by the task's regulatory classification: all tasks involving financial calculations, claims adjudication, or tasks at any stage of the pipeline that are subject to external audit are assigned to human practitioners independent of agent confidence scores.

This structure means that the effectiveness of the governance structure is not dependent on the performance of the agents but rather on the taxonomy, allowing for greater accountability as agent performance evolves.

3. Results

3.1 PAGM Governance Framework

Enterprise AI-assisted test execution systems require governance and controls to ensure accountability, transparency, and operation within acceptable risk thresholds. Governance is particularly relevant in highly regulated sectors such as financial services, healthcare, and aerospace, where the traceability of AI-assisted decisions poses the largest challenge, especially when the supporting AI is not sufficiently explainable [5]. PAGM meets these requirements through its three-tier architecture, in which Tier 3 human ownership covers all validation tasks requiring documented, auditable judgment.

Explainability is a foundational governance requirement for AI testing systems. Multiple classes of explainable AI methods exist, including visual explanation, knowledge extraction, influence methods, and example-based methods, classified by whether explanations are post-hoc or real-time and whether they describe model behavior globally or locally [6]. The most common category of XAI publications on artificial neural networks among the 381 studies was model-agnostic interpretability methods. Local XAI methods made up 5% of the reviewed publications. In practice, human reviewers are more likely to trust and effectively oversee AI systems when provided with explanations. Human defect detection performance correlates positively with the quality of AI's decision rationale.

Key aspects of trustworthy AI systems include predictive performance, robustness to adversarial examples, generalization to unknown data, explainability, lifecycle transparency, reproducibility, absence of systemically biased outcomes, and privacy. To enable accountability for the execution of AI tests in regulated domains, a systemic view is required that includes the design, development, deployment, operation and maintenance of the AI system throughout its lifecycle including the relevant documentation and auditing processes [5]. Table 2 maps these trustworthiness dimensions to their corresponding governance requirements within PAGM.

Table 2. Eight Dimensions of Trustworthy AI Systems in Enterprise Test Execution [5]

Dimension	Definition	Governance Requirement for AI Test Execution
Robustness	Resilience to adversarial attacks and distributional changes	Validation against edge cases and malicious inputs in test scenarios

Generalization	Capability to perform on unseen data	Verification of test agent performance on novel application states
Explainability	Understandability of prediction decisions	Clear reasoning for test pass/fail determinations and action selections
Transparency	Disclosure of system lifecycle information	Complete documentation of training data, model architecture, and decision logic
Reproducibility	Replicability of computational procedures	Deterministic test result generation for audit and compliance purposes
Fairness	Mitigation of systematic bias	Unbiased validation across diverse application scenarios and user workflows
Privacy	Protection against unauthorized data use	Secure handling of sensitive test data and application information
Accountability	Alignment with human normative values	Traceable responsibility for test decisions and outcomes in regulated environments

PAGM's Tier 2 confidence threshold influences agent governance frameworks. It determines if an agent continues to operate autonomously or relinquishes control to a human operator. Governance frameworks also determine what an agent should do if its Tier 2 confidence is below a threshold. For example, the agent may pause its execution, create a saved state, and call human testers.

The eight main QA challenges in autonomous AI test execution are identified in the literature [3]. They correspond to the design objectives of PAGM's three-tier architecture. Table 1 below identifies these eight challenges and their impacts on enterprise test execution.

Table 3. Quality Assurance Challenges for Autonomous AI Test Execution Systems [3]

Challenge Category	**Description**	**Impact on Enterprise Test Execution**
Interpretability	Understanding how AI models make testing decisions and reach conclusions	Affects trust, debugging capabilities, and regulatory compliance in test failures
Lack of Specifications	Absence of formal requirements and defined test criteria for AI behavior	Complicates validation of AI agent correctness and expected behavior
Validation Data Generation	Creating appropriate test datasets and input scenarios for AI training	Limits test coverage effectiveness and AI learning quality
Oracle Determination	Defining correct outcomes for AI test validation and verification	Challenges establishing ground truth for AI test decision verification
Accuracy and Correctness	Quantifying testing effectiveness, reliability, and precision levels	Hinders objective performance assessment of AI test agents
Non-functional Properties	Assessing performance, security, scalability, and usability aspects	Results in incomplete quality evaluation beyond functional testing

Self-adaptation	Managing evolving AI behavior and continuous learning over time	Introduces unpredictability and version control challenges in test execution
Dynamic Environment	Handling changing application contexts, states, and runtime conditions	Requires continuous adaptation to evolving system behaviors and interfaces

3.2 Comparative Analysis: AI-Assisted vs. Traditional Test Automation

Traditional automated tests are deterministic as they involve sequences of actions, assertions, and conditionals. They are repeatable and verifiable but easily fail when application interfaces and workflows change. Nevertheless, a systematic review of regression test selection found that only 3% of empirical studies are about large-scale systems, though regression testing is often automated. It was concluded that standard regression test selection techniques have not been validated for enterprise scale software [7]. Furthermore, all reviewed regression test selection techniques require continuous calibration and empirical validation, representing a significant and ongoing maintenance cost for enterprise quality engineering organizations [7].

When appropriately designed and deployed, AI-assisted intelligent agents offer meaningful adaptability advantages. Contextual perception and probabilistic reasoning accommodate UI drift, data drift, and workflow drift that would otherwise require manual script remediation. The contextual clue-based recognition approach described in Section 2.2 handled 95% of synthetic interface changes including element attribute modifications, structural changes, and combined variations, compared to a 40 to 80 percent handling rate achieved by leading conventional tools [4].

Kernel-based neural network architectures have demonstrated superior computational performance relative to backpropagation networks and support vector machines in controlled benchmark settings. In a study on 44 software projects, kernel-based classifiers have average F-measure, G-measure, MCC and AUC which are 25%, 50%, 42%, and 14% higher than the classical classifiers respectively. This shows that for software quality problems whose distributions are not Gaussian (e.g. metric distributions), AI based approaches can help improve performance. Figure 2 shows the improvements of the kernel classifiers. Figure 1 illustrates these performance improvements.

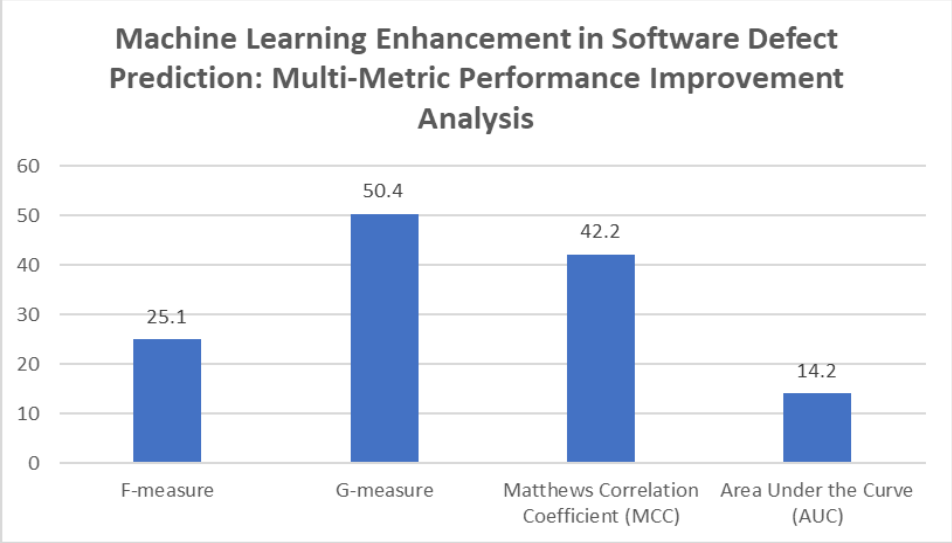

Fig. 1. Quantified AI Advantage: Percentage Improvements Across F-measure, G-measure, MCC, and AUC Metrics [9]

As shown in Figure 2, less than 3% of real-world regression test selection studies target enterprise scale systems, creating a validation gap for large enterprises to utilize empirical test selection techniques.

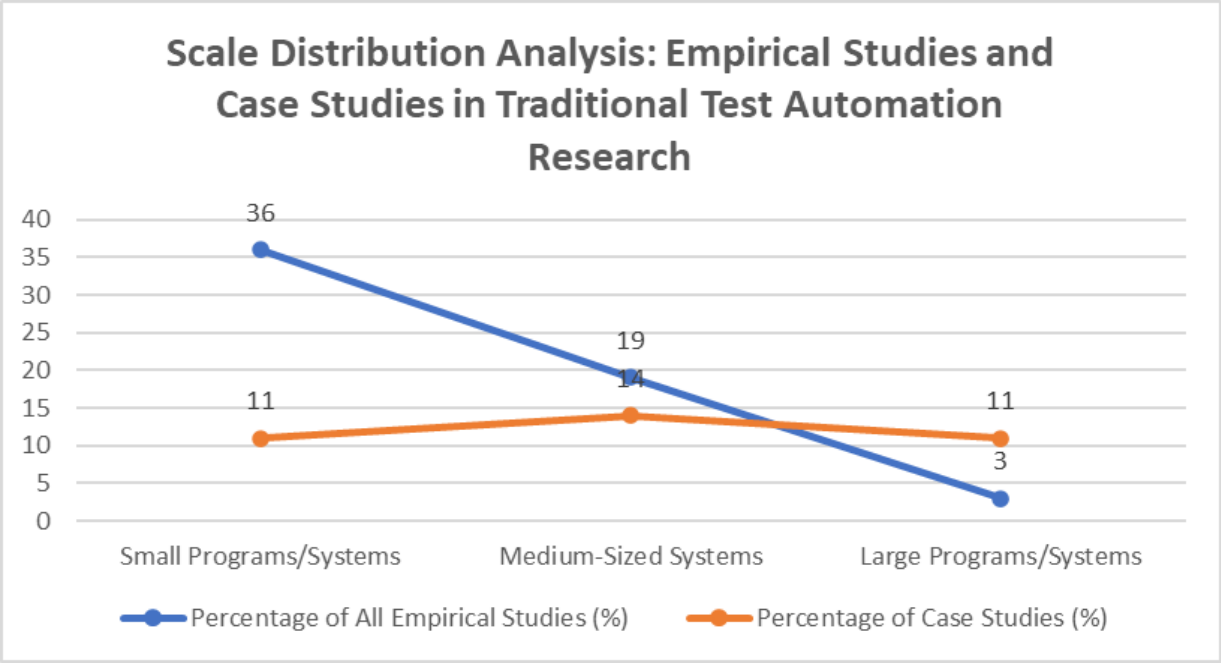

Fig. 2. Scalability Gap in Traditional Test Automation: Limited Enterprise-Scale Validation Evidence [7]

The benefit is especially salient in a control architecture like PAGM, which requires clever agents to alternate between limited degrees of autonomy rather than acting as unconstrained automated actors. The learned behavior models also generalize better to application states than those prescribed by hard-coded rules and scripts. The brittleness of path encoding is a common shortcoming of all forms of hybrid automation.

3.3 Enterprise Application: Regulated Insurance Release Pipeline

To show the enterprise applicability of PAGM, this section describes the application of PAGM to a regulated property insurance organization where regulatory audit is a business drive.

Problem: A property insurance company has subsystems for sales, claims, and underwriting rules that undergo quarterly releases. Each of the several hundred regression tests that need to run on the subsystems takes an average of five business days to run. Regulatory requirements include having human sign-off prior to releasing any calculation of a premium to production.

PAGM's three-tiered governance model was applied to manage this release pipeline in the following ways.

Tier 1 — Autonomous Execution: Using recognition of contextual clues, clever agents run regression tests on all UI navigational paths, authentication paths, and data entry paths to detect differences between builds.

Tier 2 — Confidence-Gated Escalation: For multi-factor data states such as premium calculations based on property risk scores, agent-defined criteria pause and redirect steps whose confidence scores fall below 85%. The steps are diverted for human practitioner review before the agent is allowed to proceed. This maintains the throughput of the automated solution, while also providing human checkpoints in recognized areas of concern to ensure that business critical data paths are not bypassed.

Tier 3 — Human-Led Verification: Test cases related to premium calculations or claims adjudication workflows are always assigned to human practitioners regardless of agent confidence scores. Outcomes are formally attested to by a named QA Engineer, providing documented human accountability for regulatory reporting and full traceability for financial calculation verification.

This model completes the full regression test cycle within the five-day SLA. It spans the interface drift gap that cannot be addressed with deterministic script-based automation, and generates regulatory audit documentation for electronic submission. The model delivers three concrete benefits: audit readiness through automated continuous documentation of test execution; reduced regulatory risk through preservation of human accountability where the regulatory framework requires it; and increased release confidence through higher test coverage and near real-time adaptation, combined with human judgment for areas of high complexity.

4. Discussion

4.1 Research Implications

AI-driven enterprise quality engineering remains an active and largely open research domain. Analysis of the existing literature indicates that fewer than 30% of proposed AI-supported testing methodologies have been evaluated at enterprise scale [8], and systematic reviews confirm that test suite maintenance represents a substantial ongoing cost for large organizations, with limited understanding of enterprise-scale complexity profiles.

Learning from developer behavior can inform how we can better support quality engineering using AI. A study found that developers spend 14% of their productive time in the IDE on debugging, and each debugging episode takes 42.3 seconds on average [8]. These findings complicate the bogey engineering workers across many industries have often used: that 30 to 90 percent of a developer's time is spent debugging code. The work suggests that AI tools may be well-suited to routine validation tasks, while higher-complexity diagnosis and judgment may remain human jobs.

Performance improvements by AI-based classifiers for software quality prediction (see Section 3.2) indicate that, besides UI automation, the impact of AI on quality engineering/cyber-physical production systems can also include defect prediction and risk-based test prioritization. Progress in these areas will depend on dataset size, hardware infrastructure, and problem complexity [9]. Additional evaluations across diverse enterprise contexts would provide more accurate characterizations of variance in real-world deployments.

4.2 Limitations

PAGM is presented as a conceptual governance framework grounded in a structured literature review. It has not yet been subject to longitudinal empirical validation across multiple enterprise implementations. The confidence threshold values in the insurance example and others, e.g., the 85% escalation threshold, are illustrative and need calibration according to the organization's requirements, rules and the complexity of the application.

Moreover, this architecture assumes that AI test agents can produce high-quality confidence scores for their own actions, which may not hold true for all agent architectures. Organizations deploying PAGM should validate confidence score calibration as part of the agent selection and onboarding process.

4.3 Future Directions

Addressing the gap between research-scale findings and enterprise-scale validation will require coordinated progress across several areas: inter-domain generalization through transfer learning; benchmark datasets that authentically represent enterprise complexity; hybrid human-AI collaboration frameworks with responsibility allocation governed by task complexity and regulatory requirements; and transparency and auditability frameworks designed to satisfy evolving regulatory standards. PAGM provides a governance-ready foundation for the development and evaluation of such models.

Future work can further analyze how confidence thresholds are defined over the lifetime of agents as they gain experience, how the PAGM can be integrated into continuous integration and MLOps pipelines, or how to define standardized audit trail formats for different regulatory regimes.

5. Conclusions

This paper presented the Probabilistic Augmentation and Governance Model (PAGM), a three-tier governance framework for AI-assisted test execution in enterprise and regulated software environments. PAGM formalizes the allocation of validation responsibility between intelligent agents and human testers across autonomous execution, confidence-gated escalation, and human-led verification, with enterprise traceability, regulatory compliance, and auditability as foundational design principles.

By addressing the intrinsic brittleness of script-based automation through appropriately constrained intelligent agents, PAGM provides a practical mechanism for enterprises performing regression, sanity, and structured workflow validation across multi-component systems. This was demonstrated through the regulated insurance release pipeline, in which PAGM enables completion of a full quarterly regression cycle within a five-day SLA while preserving mandatory human accountability for financial calculations.

Practical limits on prompt complexity, training data availability, and the verification of advanced business-critical logic support the hybrid human-machine approach that PAGM formalizes: AI agents are best suited to high-volume, structured validation tasks, while human practitioners retain authority over exploratory testing and high-consequence validations. PAGM meets accountability needs through explainability mechanisms, confidence thresholds, and well-defined processes of human review and escalation.

Organizations adopting this model can leverage clever agents to the enterprise level while still keeping humans in the loop for risk analysis, complex logic, or regulatory review. PAGM offers a governance-ready foundation for the continued development of AI-assisted quality engineering frameworks in regulated environments.

6. Funding Information

The author declares that no funding was received in support of this work and that no external individuals or organizations contributed to the research beyond what is cited in the references.

7. Author Contribution

Rejenish Kiran: Conceptualization, methodology, framework development, formal analysis, writing — original draft, writing — review and editing.

8. Conflict of Interest

The author declares no conflict of interest.

9. Data Availability Statement

This paper presents a conceptual governance framework developed through a structured literature review. No primary datasets were generated or analyzed. All source materials are cited in the references section and are publicly accessible via the URLs provided.

10. List of Abbreviations

AI: Artificial Intelligence

AUC: Area Under the ROC Curve

BDI: Belief-Desire-Intention

IDE: Integrated Development Environment

MCC: Morris Criteria Coefficient (also Matthews Correlation Coefficient)

MLOps: Machine Learning Operations

PAGM: Probabilistic Augmentation and Governance Model

QA: Quality Assurance

SLA: Service Level Agreement

UI: User Interface

XAI: Explainable Artificial Intelligence

XPath: XML Path Language

Biographical Sketch

Rejenish Kiran is a Senior Software Quality Engineering Leader with over 25 years of experience in enterprise QA strategy, test automation modernization, cloud migration validation, and AI-enabled testing. He currently serves as Manager of Software QA Engineering at Citizens Property Insurance Corporation in Jacksonville, Florida, where he leads manual, automation, and backend QA teams supporting mission-critical insurance platforms and enterprise modernization programs. His applied work includes initiating and leading an AI-Powered Test Execution Enablement Initiative, designing proof-of-concepts for AI-driven regression support, and establishing responsible AI adoption guidelines for enterprise testing contexts. He holds an MBA in Information Technology Management from Jain University, a B.Sc. in Computer Science from the University of Kerala, and certifications including PMP, PSM-I, SAFe Agilist, ISTQB Foundation Level, and an AI Foundations Certification from the University of North Florida. His research interests center on governance frameworks for AI-assisted testing, probabilistic reasoning in test execution, and hybrid human-AI collaboration models for regulated industries, including insurance, financial services, and healthcare.

REFERENCES

1. M. Felderer and R. Ramler, "Quality Assurance for AI-based Systems: Overview and Challenges," arXiv preprint arXiv:2102.05351, 2021. Available at: https://arxiv.org/pdf/2102.05351
2. S. Laato, M. Mäntymäki, T. Birkstedt, N. Islam, "Digital Transformation of Software Development: Implications for the Future of Work," ResearchGate, 2021. Available at: https://www.researchgate.net/publication/354124774
3. H. Araujo, M.R. Mousavi, M. Varshosaz, "Testing, Validation, and Verification of Robotic and Autonomous Systems: A Systematic Review," ACM Transactions on Software Engineering and Methodology, vol. 32, no. 2, 2023. Available at: https://dl.acm.org/doi/full/10.1145/3542945
4. R. Yandrapally, S. Thummalapenta, S. Sinha, S. Chandra, "Robust test automation using contextual clues," in Proceedings of the ACM SIGSOFT International Symposium on Software Testing and Analysis, 2014. Available at: https://dl.acm.org/doi/epdf/10.1145/2610384.2610390
5. B. Li, P. Qi, B. Liu, S. Di, J. Liu, J. Pei, J. Yi, B. Zhou, "Trustworthy AI: From Principles to Practices," ACM Computing Surveys, 2021. Available at: https://dl.acm.org/doi/pdf/10.1145/3555803
6. A. Adadi, M. Berrada, "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)," IEEE Access, vol. 6, pp. 52138-52160, 2018. Available at: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8466590
7. E. Engstrom, P. Runeson, M. Skoglund, "A systematic review on regression test selection techniques," Lund University, 2010. Available at: https://lup.lub.lu.se/search/files/4288476/3738217.pdf
8. M. Beller, N. Spruit, D. Spinellis, A. Zaidman, "On the dichotomy of debugging behavior among programmers," in Proceedings of the 40th International Conference on Software Engineering, ACM, 2018. Available at: https://dl.acm.org/doi/pdf/10.1145/3180155.3180175