Engineering Quality in the Age of AI: A Lifecycle Framework for Reliable, Interpretable and Verifiable AI Systems

Rajeew Vishvakarma
Project Manager, Infosys / American Express (*Contract*)

***Abstract*—Traditional software assurance assumes that system behaviour is defined by explicit logic and relatively stable requirements. In AI-enabled systems, these assumptions weaken because quality depends on data representativeness, probabilistic outputs, post-deployment drift, limited interpretability, and context-sensitive performance. At the same time, AI is increasingly used inside testing, defect analysis, triage, and release workflows, so both AI-enabled products and AI-supported quality processes require assurance. This paper proposes a unified, lifecycle-oriented framework for engineering software quality in the age of AI. The framework synthesises software-quality models, AI governance guidance, and machine-learning systems research into six interrelated dimensions covering models, data, verification and validation, operations, governance and traceability, and AI-supported quality processes. These dimensions are operationalised through a seven-step assurance method spanning context definition, data assessment, model evaluation, layered validation, monitoring, evidence governance, and validation of internal AI tools. An illustrative retail-banking fraud-monitoring scenario shows how reliability, robustness, interpretability, fairness, drift resilience, and accountability can be assessed within a single workflow. The paper remains conceptual rather than experimental, but it also integrates empirical findings from prior studies to ground key assurance priorities. The contribution is a practical assurance model that translates trustworthy-AI principles into implementable software-engineering practice and identifies a focused agenda for future empirical validation.**

***Keywords*—software quality; AI-enabled systems; trustworthy AI; verification and validation; data quality; monitoring; governance; interpretability; fairness; MLOps**

I. Introduction

Software quality has traditionally been discussed in terms of characteristics such as functional suitability, reliability, maintainability, performance efficiency, security, and usability [9]. These characteristics remain important, but the rise of AI-enabled systems changes the conditions under which they are achieved and assessed. In conventional software, behaviour is largely specified through explicit logic and controlled changes to code. In AI-enabled systems, by contrast, behaviour is co-determined by data, model architecture, training choices, deployment context, and environmental change. Quality can therefore no longer be treated as a purely pre-release property of deterministic artefacts.

This shift matters because AI components often generate scores, classifications, recommendations, or decisions that affect users, operations, and organisational risk. A model can perform well in development yet fail under distribution shift, subgroup imbalance, missing data, or adversarially unusual inputs. It may also remain too opaque for debugging, human review, or governance. NIST's Artificial Intelligence Risk Management Framework (AI RMF 1.0) emphasises that AI risks differ from traditional software risks because data can change significantly over time and because deployed functionality can be difficult to understand in context [16]. ISO/IEC 42001 and ISO/IEC 23894 similarly frame AI as a lifecycle governance problem rather than a one-off modelling task [10], [8].

A second change is equally important. AI is no longer only the object of assurance; it is increasingly used as an instrument of assurance. Engineering teams now employ AI to generate tests, predict likely failures, prioritise regression suites, triage defects, cluster incidents, and support release decisions. This creates a dual quality challenge: organisations must assure AI-enabled products themselves, and they must also assure AI-supported quality processes whose recommendations can influence software reliability and operational exposure.

Existing standards and studies provide valuable building blocks, but they are often fragmented across product quality, data governance, model evaluation, operational monitoring, and organisational control. Practitioners therefore still lack a single engineering view that connects these elements across the software lifecycle. This paper addresses that gap by proposing a lifecycle-oriented framework that integrates technical, operational, and governance evidence into one assurance model.

The paper makes four main contributions. First, it reframes software quality for AI-enabled systems by unifying two perspectives that are often treated separately: the quality of AI-enabled systems and the quality of AI-supported engineering processes. Second, it defines six interrelated quality dimensions that connect data, models, testing, operations, governance, and internal AI tools. Third, it operationalises those dimensions through a seven-step assurance method that links pre-release evaluation with post-deployment monitoring and traceability. Fourth, it demonstrates the framework through an industrially realistic fraud-monitoring scenario and derives practical guidelines for implementation.

The remainder of the paper is structured as follows. Section 2 explains why AI changes the software-quality problem. Section 3 reviews related work and the standards landscape. Section 4 identifies the research gap and design principles. Section 5 presents the integrated lifecycle framework. Section 6 details the assurance method. Section 7 illustrates the framework in a retail-banking fraud-monitoring setting. Section 8 discusses implications, limitations, and future empirical validation. Section 9 concludes.

II. Background: Why AI Changes the Software-Quality Problem

Traditional software engineering assumes that quality can be judged against functional and non-functional requirements that are relatively stable, observable, and testable through deterministic execution. These assumptions weaken when behaviour is partly learned from data. In AI-enabled systems, performance depends not only on code correctness but also on data quality, feature pipelines, training procedures, and the relationship between operational conditions and development-time evaluation data.

AI reshapes software quality in at least four ways. First, behaviour is highly data-dependent: incomplete, stale, unrepresentative, or poorly labelled data can degrade outcomes even when code remains unchanged. Second, outputs are often probabilistic rather than exact, which makes calibration, uncertainty handling, and threshold management central quality concerns. Third, behaviour can deteriorate after release because of data drift, concept drift, environmental change, or changes in human and organisational behaviour. Evidence from in-the-wild distribution-shift studies illustrates that models performing well on development distributions can degrade materially in deployment settings; in one WILDS subpopulation-shift example, accuracy falls from 55.3% to 32.8% [12]. Fourth, many AI components remain difficult to interpret at the level required for debugging, human challenge, or governance, especially when downstream action depends on a decision threshold or composite pipeline.

These changes create two complementary quality agendas. The first concerns the quality of AI-enabled systems themselves, including reliability, robustness, interpretability, fairness, drift resilience, and accountability. The second concerns AI supporting software quality processes, including AI-assisted test generation, failure prediction, anomaly detection, triage support, and release-risk assessment. Figure 1 summarises this two-sided view. Treating only one side as a quality problem is insufficient because internal AI tools can reshape what gets tested, what gets investigated, and what ships to production.

The practical implication is that quality in AI-enabled software must be established through a broader evidentiary base than headline accuracy alone. Teams need evidence about data integrity, scenario coverage, subgroup behaviour, explanation usefulness, monitoring controls, approval records, and the trustworthiness of any AI used in the quality workflow itself. This motivates a framework that is lifecycle-oriented, multi-dimensional, and explicit about governance and traceability.

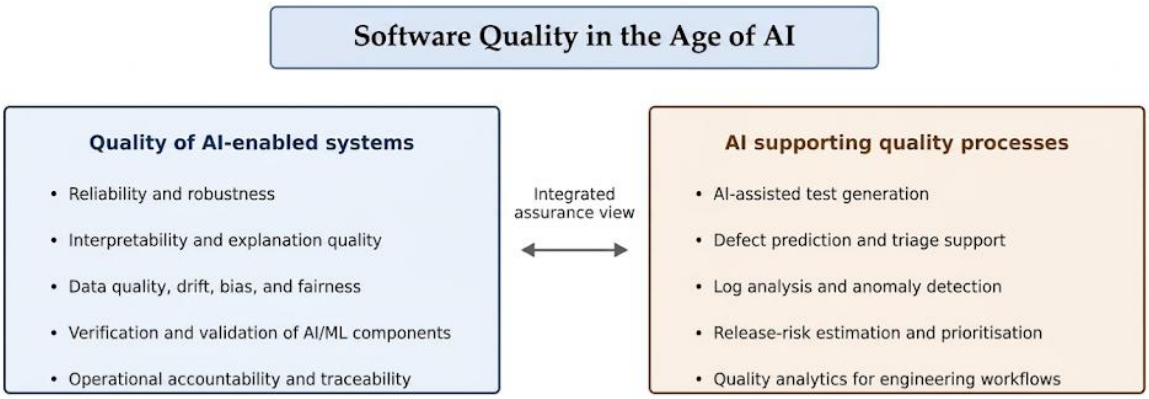

Figure 1. Two complementary perspectives on software quality in the age of AI: assuring AI-enabled systems and assuring AI-supported quality processes.

III. Related Work

Prior literature relevant to this paper can be organised into six themes. The first theme concerns the software-engineering realities of machine-learning systems. Sculley et al. [24] described hidden technical debt in ML systems, while Breck et al. [2] translated production readiness into a practical rubric of tests, monitoring expectations, and release gates. Amershi et al. [1] showed that organisations building AI-based products face coordination and workflow challenges that differ materially from conventional software projects.

The second theme addresses data quality and dataset governance as production concerns. Polyzotis et al. [19] argued that data validation should be treated as a first-class component of ML pipelines, and Gebru et al. [6] proposed datasheets for datasets to improve transparency, reproducibility, and accountability. NIST SP 1270 extends this line by emphasising that harmful bias in AI should be approached as a socio-technical risk that emerges across the lifecycle rather than as a purely statistical defect to be detected at one stage only [23].

The third theme concerns interpretability and documentation of model behaviour. Ribeiro et al. [20] introduced LIME for local, human-interpretable explanations, while Lundberg and Lee [14] proposed SHAP as a unifying feature-attribution framework. Mitchell et al. [15] complemented explanation techniques with model cards, which document intended uses, evaluation conditions, and group-specific performance considerations. Together, these studies show that explanation quality must be considered alongside performance quality.

The fourth theme concerns testing and validation of AI/ML components. DeepXplore popularised automated white-box testing ideas for deep learning systems and demonstrated that test adequacy for AI cannot be reduced to traditional unit or integration coverage alone [18]. Galhotra et al. [5] showed that discrimination can be framed as a software-testing problem, and Saleiro et al. [22] operationalised bias and fairness audits for practical workflows. Holstein et al. [7] complemented tooling-oriented fairness work with an empirical study of practitioner needs across industry teams. In NLP, behavioural-testing work such as CheckList showed that strong held-out accuracy can hide systematic failures that only surface under capability-oriented or perturbation-based testing (Ribeiro et al., 2020). These findings reinforce the need for layered verification and validation rather than single-metric acceptance.

The fifth theme concerns deployment fragility and operational monitoring. D'Amour et al. [3] demonstrated that many ML pipelines are underspecified, meaning that models with similar test-set performance can behave very differently in deployment. Klaise et al. [11] argued for monitoring that extends beyond predictive performance to include data drift, outlier detection, and explainability support. Kumar et al. [13] provided a taxonomy of machine-learning failure modes that further illustrates why ML failures differ from traditional software failures and why post-release monitoring is part of quality engineering rather than an optional operational add-on.

The sixth theme concerns governance standards and policy guidance. ISO/IEC 25010 provides a product-quality reference model, ISO/IEC 23894 provides AI-specific risk-management guidance, and ISO/IEC 42001 specifies requirements for an AI management system [9], [8], [10]. NIST AI RMF 1.0 structures trustworthy-AI work around the functions Govern, Map, Measure, and Manage [16]. The OECD AI Principles and the EU AI Act reinforce the importance of transparency, accountability, human oversight, and traceability at organisational level [17], [4].

Taken together, these streams provide strong foundations. However, they remain dispersed across technical methods, operational controls, and governance expectations. The central problem for practitioners is therefore not the absence of any individual technique, but the absence of an integrated engineering model that connects them coherently across the lifecycle.

Literature-derived empirical signals

Although the present paper does not report a new experiment, published empirical studies already show why AI quality must be treated as a multi-evidence engineering problem. Amershi et al. [1] combined 14 interviews with 551 survey responses from Microsoft engineers and reported statistically significant positive correlations between workflow maturity and perceived activity effectiveness (Spearman 0.4982-0.7627). DeepXplore generated one erroneous test input per second on average across 15 deep-learning models, achieved 34.4% and 33.2% higher neuron coverage than random and adversarial baselines, and improved accuracy by up to 3% after retraining on generated tests [18]. WILDS further showed that natural distribution shifts can materially reduce performance in real-world settings [12].

Human-centred and behavioural-testing studies point in the same direction. Holstein et al. [7] studied 35 practitioners across 25 ML product teams from 10 companies and surveyed 267 industry practitioners, showing that fairness needs are deeply embedded in organisational practice rather than confined to model selection alone. Ribeiro et al. [21] reported that practitioners using CheckList created twice as many tests and found almost three times as many bugs as a comparison group. Table 1 synthesises representative empirical signals and shows how they motivate the framework proposed here. These studies do not validate the full integrated framework directly, but they provide concrete evidence for the priority assigned to data quality, layered validation, operational monitoring, governance, and human review.

TABLE 1. LITERATURE-DERIVED EMPIRICAL SIGNALS MOTIVATING THE PROPOSED ASSURANCE FRAMEWORK.

Study	Empirical setting	Selected quantitative signal	Implication for the framework
Amershi et al. (2019)	Microsoft study of ML engineering practice	14 interviews; 551 survey responses; workflow-maturity and effectiveness correlations of 0.4982-0.7627	Quality engineering for AI is also a workflow and coordination problem, not only a model problem.
Pei et al. (2017)	Coverage-guided testing of 15 DL models on 5 datasets	One erroneous input per second on average; 34.4% and 33.2% higher neuron coverage than random and adversarial baselines; up to 3% accuracy gain after retraining	Layered V&V and challenge testing can reveal faults missed by conventional testing alone.
Koh et al. (2021)	WILDS benchmark of 10 real-world distribution-shift datasets	Example subpopulation-shift accuracy drop from 55.3% to 32.8%	Operational quality must include drift resilience, monitoring, and re-evaluation under realistic shifts.
Ribeiro et al. (2020)	User studies of behavioural testing with CheckList	Practitioners created 2x as many tests and found almost 3x as many bugs	Aggregate accuracy is insufficient; capability-oriented tests materially improve defect discovery.
Holstein et al. (2019)	Fairness practices across industry ML teams	35 practitioners across 25 teams from 10 companies; survey of 267 practitioners	Fairness and subgroup analysis are socio-technical assurance tasks that require process support, not only metrics.

IV. RESEARCH GAP AND DESIGN PRINCIPLES

The research gap addressed in this paper is integrative rather than merely incremental. Existing work offers valuable techniques for robustness testing, fairness analysis, interpretability, data validation, MLOps, and AI governance. Yet software teams do not experience these as separate problems. They must decide whether an AI-enabled system is ready to release, safe to operate, diagnosable in production, and governable when failures occur. They must also decide whether AI tools used within testing, triage, or release support should themselves be trusted.

Current standards help, but they do not by themselves provide a concrete engineering workflow for connecting model behaviour, data quality, validation evidence, monitoring, and governance artefacts. ISO/IEC 25010 gives a quality vocabulary. ISO/IEC 23894 and ISO/IEC 42001 strengthen lifecycle risk-management and management-system thinking. NIST AI RMF provides a practical risk-governance structure. However, software teams still need an operational synthesis that translates these high-level anchors into day-to-day assurance actions.

The framework proposed here was designed according to three principles. First, it must bridge technical and organisational controls. AI quality failures often emerge from interactions among data, thresholds, process decisions, and governance gaps rather than from model architecture alone. Second, it must connect pre-release validation with post-deployment monitoring, because quality in AI-enabled systems is dynamic. Third, it must treat AI-supported quality processes as part of the quality system rather than as neutral automation. This last point is important because internal AI tools can materially influence test depth, defect visibility, triage priorities, and release outcomes.

The framework is intentionally conceptual rather than an empirical benchmark study. Its value lies in providing a structured, lifecycle-oriented model that can support design reviews, release gates, operational controls, audit preparation, and future empirical validation. To make the model usable, the paper converts these principles into six quality dimensions and a seven-step assurance method.

V. Proposed Integrated Lifecycle Framework

The framework is organised around six interrelated dimensions of quality. These dimensions are not separate checklists; they are mutually reinforcing lenses through which release readiness and operational quality should be assessed. Figure 2 shows how they cut across the lifecycle from requirements and risk analysis to deployment and improvement. Table 2 summarises the six dimensions and their assurance objectives.

Model quality

Model quality covers predictive performance, reliability, robustness, calibration, interpretability, and stability. Performance remains necessary, but it is insufficient on its own. A model may appear accurate in development while still being brittle under perturbation, poorly calibrated for decision thresholds, or too opaque for effective review and debugging. Model quality therefore includes both statistical adequacy and operational intelligibility.

Data quality

Data quality covers completeness, consistency, freshness, representativeness, label quality, lineage, and bias exposure. Because AI behaviour is data-dependent, weak data quality can undermine overall system quality even when code quality remains high. This dimension also includes dataset and feature documentation, sampling assumptions, and monitoring for data changes after deployment.

Verification and validation quality

Verification and validation quality concerns the adequacy of evidence used to judge whether the system is fit for purpose. In AI-enabled systems, this must go beyond conventional functional testing to include challenge scenarios, subgroup analysis, behavioural tests, stress conditions, regression checks for evolving models, and end-to-end pipeline validation. The key question is not simply whether the model works on average, but whether the evidence is strong enough for the intended use and risk profile.

Operational quality

Operational quality covers drift detection, monitoring, incident response, threshold governance, retraining controls, rollback readiness, and production stability. AI quality is dynamic; it must be sustained after release rather than assumed from development-time performance. This dimension is therefore essential for systems that continue to learn, adapt, or operate in changing environments.

Governance and traceability

Governance and traceability cover documentation, lineage, reproducibility, approval records, change control, decision logs, and accountability structures. These elements make quality claims inspectable and contestable. They are also what allow organisations to reconstruct why a model behaved in a certain way, what evidence supported release, and what changed when incidents occur.

AI-supported quality processes

AI-supported quality processes cover AI used in testing, defect management, log analytics, triage, and release decision support. This dimension recognises that internal AI tools can improve quality but can also weaken it if they are inaccurate, biased, unstable, or insufficiently monitored. Assurance is therefore required not only for AI-enabled customer-facing systems but also for AI embedded inside the quality workflow itself.

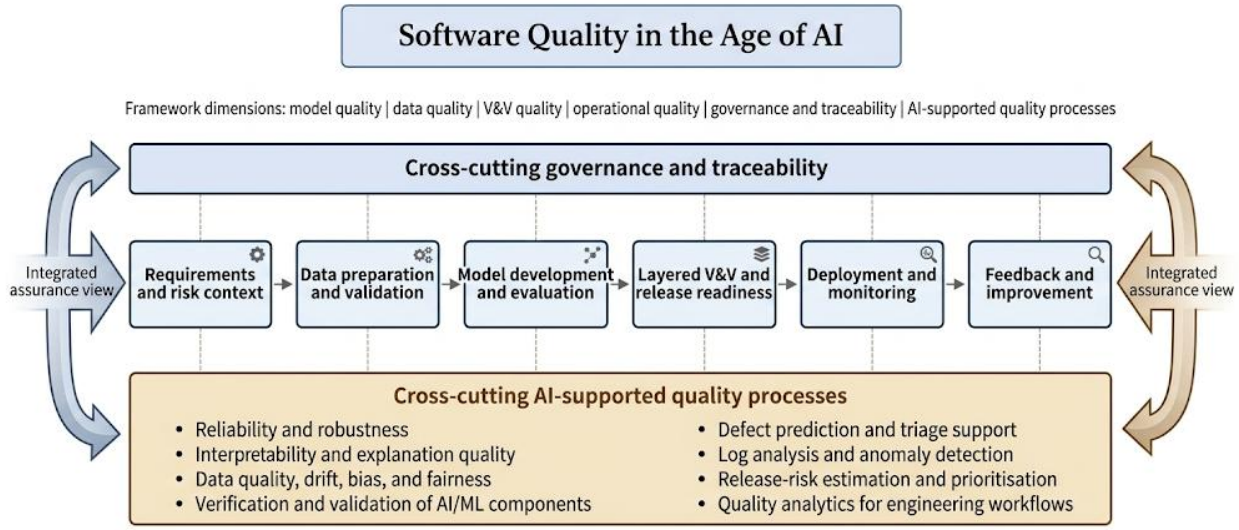

Figure 2. Lifecycle-oriented assurance model showing the sequential lifecycle stages and the cross-cutting roles of governance, traceability, and AI-supported quality processes.

TABLE 2. SIX FRAMEWORK DIMENSIONS AND THEIR ASSURANCE OBJECTIVES.

Dimension	Main focus	Assurance objective
Model quality	Performance, robustness, calibration, interpretability, stability	Ensure the AI component behaves acceptably and predictably in its intended context.
Data quality	Completeness, freshness, representativeness, label quality, bias exposure	Ensure inputs can support reliable and fair system behaviour.
Verification and validation quality	Testing depth, scenario coverage, subgroup analysis, regression evidence	Ensure the system is fit for purpose and that evidence is decision-ready.
Operational quality	Drift detection, monitoring, incident response, rollback and retraining controls	Sustain acceptable behaviour after deployment.
Governance and traceability	Lineage, documentation, approvals, reproducibility, change control	Support accountability, reproducibility, and inspectability of quality claims.
AI-supported quality processes	AI in test generation, triage, defect analytics, release-risk support	Ensure internal AI tools improve rather than weaken software quality.

VI. LIFECYCLE-ORIENTED ASSURANCE METHOD

The six dimensions become actionable through a seven-step assurance method. The seven steps were not chosen arbitrarily. They emerged from grouping the minimum distinct activities needed to operationalise the six dimensions across the lifecycle without collapsing governance, monitoring, or AI-in-QA concerns into generic evaluation. The first four steps address pre-release quality evidence, the next two address operational and governance continuity, and the final step addresses internal AI tools that influence software quality decisions.

Define system context and risk profile

Define the intended use, affected stakeholders, operational criticality, likely failure consequences, and regulatory or contractual sensitivities. The goal is to set proportional assurance expectations. A model supporting a low-risk internal recommendation service does not require the same evidence package as a model influencing fraud alerts, clinical decisions, or eligibility outcomes.

Assess data quality and bias exposure

Assess completeness, freshness, representativeness, feature integrity, class balance, label quality, and potential bias mechanisms. Evidence should include provenance, documentation, and explicit checks for sensitive or affected subgroups where appropriate. Practical indicators may include missingness rates, label-agreement checks, class-balance ratios, subgroup error gaps, or fairness indicators such as false-positive-rate parity and equalised-odds deltas. This step establishes whether the inputs can credibly support model claims.

Evaluate model quality

Evaluate predictive performance together with calibration, robustness under perturbation, explanation usefulness, and stability across relevant slices of the data. The aim is to test whether the model is not only accurate enough but also dependable, interpretable, and decision-ready for the intended context.

Perform layered verification and validation

Combine benchmark testing with behavioural testing, challenge scenarios, subgroup analysis, regression testing for model updates, and integration testing for the full pipeline. Layered V&V is required because different failure modes surface under different forms of evidence. A strong aggregate metric cannot substitute for targeted challenge design.

Establish operational monitoring and response

Define what will be monitored after deployment, including data drift, concept drift, score instability, anomalous inputs, service degradation, alert-volume shifts, and incident triggers. Monitoring may combine population stability indices, calibration drift, subgroup-performance dashboards, latency and service-level indicators, and human-review queues. Monitoring should be tied to thresholds, escalation paths, and response rules such as rollback, retraining review, or threshold adjustment.

Govern evidence and traceability

Maintain versioned records of datasets, features, models, thresholds, evaluation reports, approvals, deployment decisions, monitoring outputs, and incident responses. This step converts quality claims into inspectable evidence and enables later reproduction, audit, or challenge.

Validate AI-supported quality processes

Benchmark AI tools used for test prioritisation, triage, defect prediction, or release-risk support against simpler baselines and monitor them over time. Their outputs should remain reviewable by humans where they materially affect consequential engineering decisions. This step closes a gap that is often ignored in practice.

TABLE 3 PROVIDES AN EXAMPLE MAPPING FROM EACH DIMENSION TO INDICATIVE METRICS, EVIDENCE SOURCES, AND POSSIBLE ACTIONS. THE PURPOSE IS NOT TO PRESCRIBE A UNIVERSAL METRIC SET, BUT TO SHOW HOW THE FRAMEWORK CAN BE TRANSLATED INTO OPERATIONAL ARTEFACTS.

TABLE 3. EXAMPLE METRICS, EVIDENCE SOURCES, AND INDICATIVE ACTIONS BY FRAMEWORK DIMENSION.

Dimension	Indicative metrics	Typical evidence	Indicative actions if outside tolerance
Model quality	F1 or AUROC; calibration error; perturbation sensitivity; explanation stability	Evaluation reports; model cards; slice analyses	Recalibrate thresholds; refine model; add robustness tests; restrict use case.
Data quality	Missingness; feature freshness; label disagreement; class imbalance; subgroup coverage	Profiling reports; lineage logs; datasheets	Repair pipeline; relabel samples; rebalance data; block release if critical gaps remain.
Verification and validation quality	Scenario coverage; subgroup deltas; challenge-case failure rate; regression pass rate	Test suites; behavioural tests; integration reports	Add scenarios; expand challenge cases; hold release until coverage is adequate.
Operational quality	Population-stability index; score-distribution shift; alert-volume spikes; rollback rate	Monitoring dashboards; incident logs	Escalate incident review; enable fallback mode; retraining or threshold review.
Governance and traceability	Lineage completeness; approval coverage; reproducibility success rate	Change records; approval logs; audit artefacts	Complete missing evidence; freeze deployment until traceability gaps are closed.
AI-supported quality processes	Failure-detection recall; ranking precision; recommendation stability; override rate	Release analytics; defect outcomes; reviewer feedback	Rebaseline against simpler methods; tighten human review; retrain or retire tool.

VII. ILLUSTRATIVE APPLICATION: RETAIL-BANKING FRAUD MONITORING

To demonstrate how the framework can be used in practice, consider a retail-banking fraud-monitoring platform that scores payment transactions for investigation. The system operates in a continuous-delivery environment and includes an AI-supported regression-prioritisation service that selects the most informative tests after model and rules changes. This setting is suitable for illustration because it combines risk-sensitive predictions, concept drift, operational thresholds, and internal AI support tools.

Under model quality, the fraud-scoring model is assessed not only for precision, recall, and false-positive burden, but also for calibration near decision thresholds, robustness to missing or noisy signals, and the usability of explanation outputs for investigators. Under data quality, the team reviews transaction-feature freshness, delayed labels, class imbalance, merchant and channel coverage, and subgroup behaviour across customer segments.

Under verification and validation quality, the team supplements benchmark evaluation with seasonal stress scenarios, behavioural tests for unusual device and location combinations, subgroup analysis, regression checks after threshold changes, and end-to-end pipeline tests covering feature extraction, model scoring, routing, and case creation. Under operational quality, monitoring tracks feature drift, score-distribution shift, alert-volume spikes, investigator backlog, false-positive surges, and rollback events.

Under governance and traceability, the platform retains versioned records of datasets, feature contracts, model artefacts, thresholds, validation reports, release approvals, explanation configurations, and monitoring evidence. Under AI-supported quality processes, the regression-prioritisation model is itself evaluated for failure-detection recall, ranking precision, and stability over time against simpler heuristics. Table 4 illustrates the resulting evidence package and decision questions by dimension.

The scenario shows why quality failures rarely arise from one source alone. A release issue may stem from stale data, threshold changes, fragile explanations, insufficient challenge scenarios, weak monitoring, or over-reliance on an internal AI tool. The value of the framework lies in making these interactions visible and in forcing teams to maintain a coherent evidence trail rather than relying on isolated metrics or after-the-fact explanations.

TABLE 4. ILLUSTRATIVE EVIDENCE PACKAGE FOR THE FRAUD-MONITORING SCENARIO.

Framework dimension	Illustrative assurance question	Example evidence in the fraud-monitoring scenario
Model quality	Is the scoring model reliable near the investigation threshold?	Precision-recall curves, calibration plots, explanation review by investigators, perturbation tests on missing features.
Data quality	Are labels, features, and customer segments adequately represented?	Feature freshness checks, delayed-label analysis, class balance reports, subgroup coverage review.
Verification and validation quality	Have realistic failure scenarios been tested?	Seasonal fraud-surge scenarios, behavioural tests for unusual device-location combinations, regression and integration results.
Operational quality	Will degradation be detected and acted on quickly?	Drift dashboards, alert-volume monitoring, investigator backlog thresholds, rollback runbook.
Governance and traceability	Can the release decision be reconstructed later?	Versioned datasets, feature contracts, model artefacts, validation reports, threshold approvals, monitoring evidence.
AI-supported quality processes	Does the regression-prioritisation service improve failure detection without creating blind spots?	Ranking precision, failure-detection recall versus heuristics, override logs, reviewer feedback.

VIII. DISCUSSION

The proposed framework suggests that software quality in AI-enabled systems should be treated as a multi-evidence engineering discipline. Performance metrics remain important, but they do not capture the full quality picture. Quality claims for AI-enabled systems require evidence about data integrity, validation depth, monitoring, governance, and the trustworthiness of AI embedded within the quality workflow itself.

The framework also provides a practical bridge between software-quality thinking and AI-governance thinking. ISO/IEC 25010 contributes a product-quality lens; ISO/IEC 23894 and ISO/IEC 42001 contribute lifecycle risk management and organisational control; NIST AI RMF contributes a practical governance structure; NIST SP 1270 sharpens the treatment of harmful bias as a socio-technical concern; and the EU AI Act reinforces the value of logging, traceability, and human oversight in higher-risk contexts [23], [4]. The framework does not replace these anchors. Rather, it provides an engineering synthesis that teams can apply during design reviews, release readiness, and operational assurance.

Several practical guidelines follow. Organisations should define AI quality requirements during system design rather than after model training. Data quality should be treated as a first-class quality dimension. Validation should extend beyond aggregate accuracy to include robustness, subgroup behaviour, explanation usefulness, and behavioural challenge scenarios. Monitoring, retraining controls, and rollback readiness should be part of release planning. End-to-end traceability should be maintained for data, features, models, thresholds, and approvals. Finally, AI tools used in testing or release management should be validated and monitored rather than assumed to be harmless because they are internal.

The framework should also be applied proportionately. A full seven-step evidence package may be unnecessary for low-risk internal tools and burdensome for small teams if interpreted as a fixed checklist. In practice, organisations should scale the depth of assurance by materiality and risk: core controls around data quality, layered validation, monitoring, and traceability should be universal, while the intensity of fairness review, explanation analysis, and AI-in-QA validation can be expanded for higher-impact systems. This risk-proportionate use helps avoid checklist fatigue while retaining accountability.

Limitations and future empirical validation

This paper is a conceptual and synthesis-oriented contribution rather than a large empirical study. The literature-derived empirical evidence summarised in Table 1 strengthens the motivation for the framework, but it does not substitute for direct validation of the integrated model as a whole. The framework still requires empirical testing across sectors such as finance, healthcare, public services, and consumer software. Future work should assess whether using the framework improves release readiness, drift detection, auditability, or incident response when compared with narrower assurance approaches.

A focused empirical follow-on would be feasible. For example, a study could apply the framework to a public fraud or tabular risk dataset, introduce controlled data-drift and threshold-change scenarios, and measure whether the proposed evidence package improves detection of degradation, reproducibility of decisions, or operational response time. A second line of work could compare AI-assisted quality tools with simpler baselines to determine when they improve quality and when they introduce new forms of risk. Such studies would strengthen the framework's external validity and sharpen its sector-specific adaptations.

TABLE 5. ALIGNMENT OF THE PROPOSED FRAMEWORK WITH SELECTED STANDARDS AND GUIDANCE.

Guidance	Primary relevance to the framework
ISO/IEC 25010:2023	Provides a product-quality model and vocabulary for expressing quality characteristics and trade-offs.
ISO/IEC 23894:2023	Frames AI-specific risk management across the lifecycle and supports proportional controls.
ISO/IEC 42001:2023	Adds management-system discipline, governance responsibilities, traceability, and continual improvement.
NIST AI RMF 1.0	Structures AI risk work around Govern, Map, Measure, and Manage functions.
NIST SP 1270	Strengthens the treatment of harmful bias as a socio-technical lifecycle risk rather than only a statistical defect.
OECD AI Principles	Provides high-level guidance on trustworthy, human-centred, and accountable AI.
EU AI Act (2024)	Reinforces logging, traceability, risk management, and human oversight expectations in higher-risk contexts.

IX. CONCLUSION

AI changes software quality in two connected ways: it introduces new risks into software-intensive systems, and it provides new tools for improving software-quality work. Existing standards and research offer important foundations, but practitioners still need an integrated engineering model that connects data, models, validation, operations, governance, and AI-supported quality processes across the lifecycle.

This paper has proposed such a model. By organising assurance around six dimensions and operationalising them through a seven-step method, the framework provides a structured way to move from high-level trustworthy-AI principles to implementable software-engineering practice. Its main value is not in prescribing one universal metric set, but in showing how multiple forms of evidence can be connected to support release readiness, operational resilience, traceability, and accountable use of AI in both products and engineering workflows. As AI becomes more deeply embedded in software systems and delivery processes, lifecycle-oriented quality engineering will become not an optional enhancement, but a core capability.

X. ACKNOWLEDGMENT

The author would like to thank colleagues and reviewers whose feedback helped improve the clarity, structure, and practical relevance of this manuscript. The author also acknowledges the broader research and standards community in software engineering, AI governance, and machine learning systems for providing the foundational ideas that informed this work.

XI. DECLARATIONS

Funding. No specific funding was received for this study.
Competing interests. The author declares no competing interests.
Data availability. No datasets were generated or analysed during the current study.

XII. REFERENCES

[1] Amershi, S., Begel, A., Bird, C., DeLine, R., Gall, H., Kamar, E., Nagappan, N., Nushi, B. and Zimmermann, T. (2019), "Software engineering for machine learning: a case study", 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), IEEE, pp. 291-300.

[2] Breck, E., Cai, S., Nielsen, E., Salib, M. and Sculley, D. (2017), "The ML test score: a rubric for ML production readiness and technical debt reduction", Google Research.

[3] D'Amour, A., Heller, K., Moldovan, D., Adlam, B., Alipanahi, B., Beutel, A., Chen, C., Deaton, J., Eisenstein, J., Hoffman, M.D. et al. (2022), "Underspecification presents challenges for credibility in modern machine learning", Journal of Machine Learning Research, Vol. 23, pp. 1-86.

[4] European Union (2024), "Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act)", Official Journal of the European Union, OJ L 2024/1689, 12 July.

[5] Galhotra, S., Brun, Y. and Meliou, A. (2017), "Fairness testing: testing software for discrimination", Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering, ACM, pp. 498-510.

[6] Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J.W., Wallach, H., Daumé III, H. and Crawford, K. (2021), "Datasheets for datasets", Communications of the ACM, Vol. 64 No. 12, pp. 86-92.

[7] Holstein, K., Wortman Vaughan, J., Daume III, H., Dudik, M. and Wallach, H. (2019), "Improving fairness in machine learning systems: what do industry practitioners need?", Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, ACM, pp. 1-16.

[8] International Organization for Standardization (2023a), ISO/IEC 23894:2023 Information Technology - Artificial Intelligence - Guidance on Risk Management, ISO, Geneva.

[9] International Organization for Standardization (2023b), ISO/IEC 25010:2023 Systems and Software Engineering - Systems and Software Quality Requirements and Evaluation (SQuaRE) - Product Quality Model, ISO, Geneva.

[10] International Organization for Standardization (2023c), ISO/IEC 42001:2023 Information Technology - Artificial Intelligence - Management System, ISO, Geneva.

[11] Klaise, J., Van Looveren, A., Cox, C., Vacanti, G. and Coca, A. (2021), "Monitoring and explainability of models in production", arXiv preprint arXiv:2007.06299.

[12] Koh, P.W., Sagawa, S., Marklund, H., Xie, S.M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R.L., Gao, I. et al. (2021), "WILDS: a benchmark of in-the-wild distribution shifts", Proceedings of the 38th International Conference on Machine Learning, PMLR, pp. 5637-5664.

[13] Kumar, R.S.S., O'Brien, D., Albert, K., Viljoen, S. and Snover, J. (2019), "Failure modes in machine learning systems", arXiv preprint arXiv:1911.11034.

[14] Lundberg, S.M. and Lee, S.-I. (2017), "A unified approach to interpreting model predictions", Advances in Neural Information Processing Systems 30.

[15] Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I.D. and Gebru, T. (2019), "Model cards for model reporting", Proceedings of the Conference on Fairness, Accountability, and Transparency, ACM, pp. 220-229.

[16] National Institute of Standards and Technology (2023), Artificial Intelligence Risk Management Framework (AI RMF 1.0), NIST AI 100-1, U.S. Department of Commerce, Gaithersburg, MD.

[17] Organisation for Economic Co-operation and Development (2019), OECD AI Principles, OECD, Paris.

[18] Pei, K., Cao, Y., Yang, J. and Jana, S. (2017), "DeepXplore: automated whitebox testing of deep learning systems", Proceedings of the 26th Symposium on Operating Systems Principles, ACM, pp. 1-18.

[19] Polyzotis, N., Roy, S., Whang, S. and Zinkevich, M. (2019), "Data validation for machine learning", Proceedings of Machine Learning and Systems, Vol. 1, pp. 334-347.

[20] Ribeiro, M.T., Singh, S. and Guestrin, C. (2016), "'Why should I trust you?": explaining the predictions of any classifier', Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, pp. 1135-1144.

[21] Ribeiro, M.T., Wu, T., Guestrin, C. and Singh, S. (2020), "Beyond accuracy: behavioural testing of NLP models with CheckList", Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, pp. 4902-4912.

[22] Saleiro, P., Kuester, B., Stevens, A., Anisfeld, A., Hinkson, L., London, J. and Ghani, R. (2018), "Aequitas: a bias and fairness audit toolkit", arXiv preprint arXiv:1811.05577.

[23] Schwartz, R., Vassilev, A., Greene, K., Perine, L., Burt, A. and Hall, P. (2022), Towards a Standard for Identifying and Managing Bias in Artificial Intelligence, NIST Special Publication 1270, U.S. Department of Commerce, Gaithersburg, MD.

[24] Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F. and Dennison, D. (2015), "Hidden technical debt in machine learning systems", Advances in Neural Information Processing Systems 28, pp. 2503-2511.