

A Framework for Evaluating GenAI Adoption and Use in Software Engineering

Liang Yu, Emil Alégroth, Panagiota Chatzipetrou, Tony Gorschek

Abstract—Generative Artificial Intelligence (GenAI) is increasingly integrated into software products to enable new features and user capabilities, from early exploration to operational deployment. GenAI adoption as a component within a software system introduces quality risks because GenAI outputs are probabilistic, prompt-sensitive, and may drift after release. Organizations, therefore, need to decide what to evaluate, when to evaluate, and who owns quality evaluation activities across software design, development, and operations. ISO/IEC 25059 standard distinguishes between software product quality (e.g., usability) and quality-in-use (e.g., satisfaction) for AI-enabled software, yet it provides limited operational guidance for these evaluation activities. We therefore investigate how industrial software teams adopt and use GenAI models in the software systems they build and operate, and how they evaluate system qualities when deciding to adopt GenAI during development and after deployment. We do not benchmark the underlying GenAI model itself.

In this study, we conducted 19 semi-structured interviews in two software development companies. We triangulated the interviews with archival data (15 internal documents and 184 internal wiki/web pages) to capture GenAI adoption steps, quality concerns, evaluation practices, and role responsibilities.

Our findings describe a three-phase adoption process – Ideation, Development, and Operation – highlighting where quality evaluations occur, which criteria are used, and how evaluation responsibilities are distributed. Based on observed practices and using ISO/IEC 25059 as an organizing lens, we synthesize a process-oriented quality evaluation framework. This framework maps metrics to explicit gatekeeping, validation, and monitoring checkpoints, bridging abstract ISO quality characteristics with engineering workflows.

We applied the framework in a GenAI-enabled software product (SE4AI) use case and reported how it supported structured evaluation activities. We also observed that quality evaluations span legal, security, development, QA, and operations, but ownership is fragmented across phases. We therefore propose a GenAI Quality Lead responsibility (often assignable to an existing senior role) to coordinate criteria, evidence, and traceability across quality evaluation activities.

The results contribute to Software Engineering for AI (SE4AI) by clarifying how teams can measure qualities when building software that adopts and uses GenAI.

Index Terms—Generative Artificial Intelligence, GenAI, AI4SE, SE4AI, Empirical Study, Software Engineering

Liang Yu is with Dept. Software Engineering, Blekinge Institute of Technology, Karlskrona, Sweden. Email: liang.yu@bth.se

Emil Alégroth is with Dept. Software Engineering, Blekinge Institute of Technology, Karlskrona, Sweden. Email: emil.alegroth@bth.se

Panagiota Chatzipetrou is with Örebro University, Örebro, Sweden. Email: panagiota.chatzipetrou@oru.se

Tony Gorschek is with Blekinge Institute of Technology, Karlskrona, Sweden, and Fortiss, Munich, Germany. Email: tony.gorschek@bth.se

Corresponding author: Liang Yu.

I. INTRODUCTION

Generative Artificial Intelligence (GenAI) [1] with Large Language Models (LLMs) [2] has been rapidly adopted in software engineering. This adoption covers two complementary perspectives: Software Engineering for AI (SE4AI), which provides practices for building reliable AI-enabled systems, and AI for Software Engineering (AI4SE), where GenAI assists traditional development tasks [3]. From requirements analysis to code generation [4], GenAI is transforming how software is developed and maintained [5]. Industrial GenAI adoption typically starts with prompt-based experimentation and iterative refinement, gradually maturing into production-ready implementations. This adoption introduces risks for software quality [6].

The main challenge is to define what to evaluate, when, and how — with adoption activities and monitoring — for product quality and quality-in-use. Unlike traditional deterministic software [7], GenAI-based software produces probabilistic outputs that can vary across runs, even with identical inputs [6]. The generated content can be plausible but factually incorrect, biased, or potentially harmful, raising concerns around accuracy, security, and compliance [6], [8]. These characteristics – non-determinism, prompt and data sensitivity, and behavioral drift – limit the applicability of traditional software quality assurance techniques, which assume fixed logic and predictable outputs.

ISO/IEC 25059 focuses on AI-enabled software [9], distinguishing between product quality (e.g., functional suitability, reliability, security) and quality-in-use (e.g., effectiveness, efficiency, freedom from risk) [10]. ISO standards define what quality means, but they do not specify when a team should verify each quality characteristic or what evidence is required to transition from a pilot to production. For instance, teams performed legal, security, and output evaluations, but these evaluations were scattered across roles and were rarely recorded in a way that supports later review. This creates a gap between standards and industrial engineering work: teams need process-oriented guidance that ties quality evaluations to adoption steps and role ownership [11].

In this study, we examine the adoption of GenAI for software products, specifically how teams integrate and operate GenAI techniques within the software systems they develop. We focus on engineering practices and quality evaluation activities across the development and operational lifecycle. A case study [12] was conducted, which combined interviews with an analysis of internal web pages and documents, enabling us to capture both documented processes and practices. Based on the

observed AI4SE usage practices, we derived SE4AI guidance in the form of a quality evaluation framework aligned with ISO/IEC 25059. Such guidance supports recurring decisions: whether a use case can proceed past ideation, whether a GenAI feature is ready for release, and whether drift or failures require rollback or revalidation after deployment. Our framework provides a lightweight structure for these decisions by linking evaluation targets, activities, and accountable roles. Organizations can use it to reduce ad hoc gatekeeping, enhance the traceability of quality decisions, and align internal policies with external standards.

Our primary contribution is a **Quality Evaluation Framework** that synthesizes industrial practice with ISO standards. Specifically, we contribute:

- 1) **Process discovery (The “When”)**: Empirical evidence on the multi-phase adoption process and its phases (Ideation, Development, Operation) used in industry.
- 2) **Quality characterization (The “What”)**: A mapping of practitioner concerns (e.g., legal/security) to ISO/IEC 25059 quality characteristics.
- 3) **Role definition (The “Who”)**: The proposal of a *GenAI Quality Lead* role to coordinate cross-functional evaluation.
- 4) **Framework verification**: A verification of the framework on a real-world use case, demonstrating its feasibility.

The remainder of this study is structured as follows. Section II presents background information and related work on GenAI in software engineering and quality evaluation approaches. Section III describes our research methodology, including research questions, case company details, and data collection and analysis methods. Section IV presents the proposed quality evaluation framework, and Section V illustrates the framework verification through a use case implementation. Section VI addresses threats to validity. Section VII discusses implications, including the potential emergence of new roles such as AI Quality Lead. Section VIII concludes the paper with a summary of contributions and future research directions.

II. BACKGROUND AND RELATED WORK

A. Background

GenAI models [2], such as GPT-4o, have been used in software engineering activities, including code generation [13], documentation [14], and requirements drafting [15]. In industry, the adoption of GenAI is not always tied to a complete “system” from the beginning [16]. It often begins with integrating a model into a development task or piloting an idea with prompts. This adoption view helps teams understand how quality considerations arise along GenAI adoption [11].

ISO/IEC 25059 [17] provides quality measurements on AI-enabled software that integrates AI models or services. It structures quality into two complementary categories:

- **Product quality**: characteristics include functional suitability, performance efficiency, reliability, security, maintainability, and portability [17]. For example, ‘functional suitability’ concerns whether outputs satisfy specified functional requirements for the intended tasks.

- **Quality-in-use**: characteristics cover effectiveness, efficiency, satisfaction, freedom from risk, and context coverage [17]. For example, ‘freedom from risk’ covers legal, ethical, and security concerns.

We distinguish two evaluation aspects in this context. Product quality refers to the quality of the delivered software system that integrates GenAI services, including prompts, orchestration logic, validators, fallback handling, and data handling. Quality-in-use refers to system outcomes and risks in operation, such as task success, user satisfaction, and freedom from risk in the deployment context. When solutions rely on third-party models, teams do not directly inspect the model’s internals (e.g., training data); instead, assess system (AI-integrated) behavior through black-box tests on representative inputs. Teams also complement the evaluation with runtime monitoring of failures and drifts. Transparency in this setting is achieved through traceable configuration and records, such as prompt templates, evaluation datasets, and monitoring logs, together with supplier documentation when available.

B. Related work

1) *Quality evaluation methods for GenAI*: Several methods exist for evaluating the quality of AI-based software. Unlike this work, Zhang et al. [3] introduced a structured quality framework addressing aspects such as data quality, model robustness, and integration maturity. Riccio et al. [9] developed testing strategies targeting correctness and validation challenges in AI-driven systems. Similarly, Breck et al. [18] proposed the ML Test Score, a checklist-based framework for assessing operational readiness and reducing hidden technical debt.

More recently, researchers have begun to explore evaluation techniques tailored to generative tasks such as code generation. Ribeiro et al. [19] proposed CheckList, a behavioral testing model for NLP systems that identifies failures in specific linguistic capabilities, such as negation handling and coreference resolution. Wang et al. [13] reviewed evaluation metrics for code generation, including functional correctness, code readability, and runtime performance. These approaches advance technical evaluation and can be applied in later development and product evaluation. Some of them offer guidance on system-level prototypes. However, their practical applicability, scalability, and suitability for industrial adoption remain uncertain due to limited validation in real-world contexts.

Prior work has provided evaluation methods for AI-enabled systems, including readiness checklists and operational gates [18], testing guidance for AI-enabled software [9], and quality frameworks that encompass data, model, and integration concerns [3]. These contributions clarify what can be evaluated and what evidence can be collected, but they do not describe how such evidence is positioned across adoption phases or how responsibility is assigned in industrial settings. Our study complements them by providing a process-oriented view that links evaluation targets to activities and roles across software development, using ISO/IEC 25059 as a shared quality vocabulary.

2) *Empirical studies of GenAI in software engineering:*

Empirical research has examined how GenAI is applied in real-world software development contexts. Barke et al. [20] and Donvir et al. [11] investigated developer interactions with GenAI-assisted tools, reporting both productivity gains and new categories of errors introduced by model-generated code. Other studies have demonstrated GenAI’s applicability in tasks such as test case generation [6], requirements analysis [21], and documentation [14].

These studies also raise concerns regarding quality and trust. Donvir et al. [11] observed that GenAI can produce subtle, hard-to-detect defects despite speeding up ideation and coding. Such defects are hard to detect because the generated code compiles and can pass limited tests while masking logic errors and missing checks. Non-determinism and prompt sensitivity for open-ended tasks make failures difficult to reproduce and verify. Aleti et al. [6] highlighted the difficulty of verifying correctness and security in GenAI-generated outputs. Park et al. [10] emphasised the importance of prompt engineering and developer awareness for achieving high-quality outputs.

Recent studies have further unpacked the drivers of this adoption. Khojah et al. [22] and Russo et al. [23] highlight that adoption is driven less by personal innovativeness and more by workflow compatibility and trust. Lambiase et al. [24] and Choudhuri et al. [25] discuss how cultural values and cognitive styles can hinder adoption. These studies primarily view adoption through the lens of individual developer productivity and trust (AI4SE). However, an organizational challenge remains: once adopted, how teams verify the quality of the resulting GenAI-enabled products (SE4AI).

3) *Research gap:* The existing literature on GenAI adoption does not provide a usage-oriented view of how quality is evaluated in the industry. While prior works explain *why* and *how* individuals adopt GenAI, they neglect *how* organizations govern the quality of the resulting systems. Our study addresses this gap by documenting the process of GenAI adoption in software products and mapping observed evaluation practices to ISO/IEC 25059.

III. RESEARCH METHODOLOGY

We conducted a case study following the guidelines of Runeson et al. [12]. Archival data and semi-structured interviews were used to collect in-depth insights into how quality is evaluated throughout GenAI adoption in software products. An overview of the research process is shown in Fig. 1.

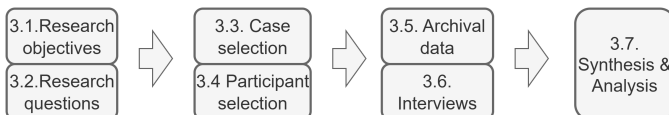


Fig. 1. Research process overview.

A. *Research objectives*

Our study objective is, first, to investigate the technical workflows of individual engineers integrating GenAI into software products; second, to understand how the engineering organization governs and evaluates software quality throughout

this adoption. We focus on capturing the exact processes followed by practitioners. In parallel, we examine how software quality is evaluated throughout the use of GenAI to identify recurring practices or patterns. For example, the practices include which quality aspects are considered important, when and how evaluations are conducted, and who is responsible for carrying them out.

B. *Research questions*

To reach the research objectives, we formulate the following research questions:

- **RQ1:** What processes do individual engineers follow when adopting GenAI in software products?
This question breaks down adoption into phases and practical steps, covering model selection, system integration, and operational use. It provides the baseline for understanding the surrounding activities where quality concerns may emerge.
- **RQ2:** What quality aspects are evaluated across GenAI adoption and use in software products?
This question focuses on the types of quality concerns that practitioners prioritize throughout the adoption and use, such as legal risk, security, or data handling.
- **RQ3:** Who within the engineering organization is responsible for evaluating quality throughout GenAI adoption?
Given the identified evaluation aspects, we examine which roles participate in quality-related work. It connects identified quality concerns to organizational ownership.
- **RQ4:** How does the organization conduct quality evaluation throughout GenAI adoption?
We investigate existing approaches to carry out the evaluations, which reveal how quality is judged in practice.

C. *Case selection*

We selected two software development companies, denoted as Company A and Company B, using purposive sampling [26] from our industrial network. Both belong to the same corporate group but operate in different domains (FinTech and Telecom). We anonymized the companies under non-disclosure agreements. When this study began, few external partners had established processes for using GenAI in software development. The selected companies offered ongoing AI initiatives and access to internal artifacts, enabling a focused in-depth case study. We discuss implications for generalizability in Section VI.

TABLE I
CONTEXT INFORMATION [27] OF THE SELECTED COMPANIES.

Context		Company A	Company B
Product	Business domain	FinTech	Telecom
	Product type	Finance services	Telecom support services
	Maturity	Mature product	Mature product
Organization	Number of employees	>500	>2000
	Corporate group	Same corporate group	Same corporate group
GenAI applications	number of AI initiatives	9	263

Table I presents the contexts of the selected companies. Company A produces a financial system (e.g., transactions, loans, and payments) that has successfully served customers for approximately two decades. Company B is in the telecommunications domain, providing support services for network management. Both companies operate globally and expand rapidly. Our data collection focused on the Swedish and Indian development sites. Both have mature software engineering practices and have established organizational goals to integrate GenAI into their products. At the time of the study, a total of 272 AI initiatives were recorded. Most of the recorded initiatives were ongoing, and some had been completed.

With many parallel AI initiatives, ad hoc practices lead to non-comparable results, duplicated effort, and limited visibility into risk and drift across teams. Shared adoption patterns and processes, such as common checkpoints (e.g., verification of data privacy), a minimal metric start-set (e.g., metrics on output validity/cost), clear role ownership (e.g., product owners), and traceable records (e.g., model versions and prompt templates), enable cross-team monitoring and incident review. This supports consistent decisions and reduces rework in legal and security reviews.

D. Research design overview

To address the research questions, we adopted a two-stage research design that combined an exploratory empirical study with a confirmatory case verification.

- **Stage 1: Framework Construction (RQ1–RQ4).** We conducted an exploratory study at two companies to understand current GenAI adoption practices and quality evaluation needs. We synthesized these findings into a quality evaluation framework.
- **Stage 2: Framework Verification.** We then applied the framework in a specific project (Section V) to verify its feasibility in a real-world setting.

This structure ensures that the framework is grounded in empirical observation (Stage 1) and validated through practical application (Stage 2).

E. Participant selection

Participants for this study were selected using purposive sampling [26] following a multi-step process. Firstly, we identified active GenAI initiatives within each company using internal documentation (see Table I for initiative counts). Secondly, the first author was embedded with development teams in both companies, which provided us with sufficient understanding of the organization. Thirdly, this embedded access gave visibility into AI initiatives (teams, internal documentation, and demo sessions), which we used to identify participants across roles and responsibilities. We prioritized practitioners with hands-on GenAI responsibilities and direct involvement in quality evaluation activities. During interviews, we collected detailed information about participants' specific roles, geographic locations, and deeper insights into their GenAI experiences. As a result, shown in Table II, 19 participants were chosen.

TABLE II
PARTICIPANTS SELECTED FROM COMPANY A AND COMPANY B

IDs	Participant role	No. of participants	Experience (years)	Company
P1–P3	Software developer	3	7 – 20	A (1), B (2)
P4–P5	Software architect	2	11 – 23	A (1), B (1)
P6–P8	Quality assurance engineer	3	8 – 19	A (1), B (2)
P9–P11	Legal engineer	3	9 – 23	A (1), B (2)
P12–P14	Security engineer	3	6 – 17	A (2), B (1)
P15–P17	Operations engineer	3	7 – 15	A (1), B (2)
P18–P19	Product owner	2	13 – 22	A (1), B (1)

F. Archival data

In addition to interviews, we collected archival data [12] (e.g., internal documents) to triangulate our findings and gain deeper insights into GenAI adoption and quality evaluation practices.

We examined a total of 184 internal wiki/web pages, which are counted as unique URLs, not A4 pages. Pages ranged from short information (1–2 paragraphs) to multi-section guides, and the count indicates breadth rather than physical length. These pages span a wide range of topics related to GenAI adoption, including GenAI development portals, legal compliance and risk assessment pages, GenAI experimentation guidelines, system development processes, and tooling documentation (e.g., API consoles and quality dashboards).

TABLE III
LIST OF SELECTED ARCHIVAL DOCUMENTS

ID	Document title / category	Purpose in analysis
Doc-GenAI-Portals	AI development portal	Overview of approved tools and access
Doc-ModelSelection	Model selection guide	Criteria for choosing permissible models
Doc-ExpGuide	GenAI handbook	Process for piloting and PoCs
Doc-TestPlan	Test plan template	Standardized testing fields
Doc-RiskGuardrails	GenAI risk guardrails	Legal/Security constraints checklist
Doc-DevOps	Ops monitoring dashboard	Metrics for runtime deployment

Furthermore, we analyzed 15 internal documents (e.g., PDF, Docx, Excel, slides), as shown in Table III, focused on how GenAI is adopted and evaluated in practice, such as “GenAI risk guardrails” (legal and technical risks), “AI adoption assessment” steps, “Responsible GenAI” checklists, model registration and vendor assessment records, security and privacy reviews, metric definitions and dashboard specifications, and training slide decks. We selected archival artifacts using a two-step approach. Firstly, we collected pages directly linked to the GenAI initiative board and its referenced templates. Secondly, we applied a sampling by following links to referenced technical pages, metric definitions, and review artifacts that were mentioned in interviews or embedded in the governance pages. We excluded artifacts that were not specific to GenAI adoption, lacked actionable procedures, or were duplicates.

We treated archival artifacts as evidence to triangulate interview statements, by confirming the existence of practices (e.g., evaluation activities and dashboards), clarifying role ownership, and extracting concrete evaluation criteria and metrics. Firstly, they provided a high-level view of the GenAI adoption process, including handoffs and ownership that individual practitioners did not always see. Secondly, they enabled a cross-check through comparison with documented procedures and templates. Thirdly, they revealed organizational expecta-

tions for GenAI quality that were not yet consistently aligned with teams' daily work.

Although the archival data strengthened credibility, they also have limits. Access was restricted to internal platforms, so coverage reflects what was available during November 2024 to June 2025. As documents varied in detail and maintenance, we mitigated these risks by cross-checking with interviews.

G. Interviews

We conducted semi-structured interviews [12] to investigate how industrial practitioners adopt GenAI in software products and how they evaluate quality during this process. The interviews enabled an in-depth exploration of practices, decision-making rationales, and cross-role responsibilities not captured in internal documents, complementing the archival analysis.

The interview guide was structured into four sections: (1) Current GenAI adoption status and recent examples of use, (2) Quality evaluation practices across design, development, and operations, (3) Challenges and risks that triggered additional evaluation activities, and (4) Roles and responsibilities, including who performed quality evaluations and who could approve progress. This design was aimed at aligning with RQ1–RQ4 by transitioning from practice descriptions to quality concerns and role ownership. The full guide is available in the replication package.

We conducted four pilot interviews with senior developers and quality managers to calibrate the interview guide. Pilot data were excluded from the main study analysis. The pilot interviews resulted in two changes. First, we separated questions about software 'product quality' from questions about 'quality-in-use', because pilot answers conflated static code properties with runtime user outcomes. Second, we adjusted the ordering so that participants first described a recent GenAI use case, and then discussed quality evaluation and role ownership, which improved recall and reduced abstract answers.

A total of 19 interviews were held between November 2024 and June 2025. Twelve participants attended in person, and seven participated online, due to the geographical distribution of participants. The interviews were conducted by the first author and held in English, using an interview guide consisting of 21 questions. The interview guide is available on the IEEEDataPort [28]. Each interview lasted between 46 and 60 minutes. We used the same interview guide across interviews. We asked core questions consistently and used follow-up probes to clarify specific examples (e.g., what artifact was produced, who approved it, or what metric was checked).

We took several actions to enhance interview validity. Firstly, we covered various roles during the participant selection. To encourage honest feedback during interviews, we informed participants that the study reports results only in anonymized form and that no raw transcripts were shared with the companies. We removed names of teams, tools, and internal systems during transcription and used participant identifiers (P1–P19). We also requested concrete, recent artifacts, such as checklist items, dashboards, and notes, to ground claims in observable practice rather than general opinions. Secondly, we refined the interview guide after pilot

interviews based on emerging themes, then used the finalized guide consistently across all remaining interviews. Thirdly, we conducted member checking with participants through email and follow-up conversations to confirm our interpretations. We have obtained the consent information from all study participants.

However, certain limitations exist. As participants came from two companies within a large corporate group with ongoing AI initiatives, the sample may not reflect settings with minimal tooling, restricted data access, or without in-house legal and security support. Moreover, self-reporting can introduce recall bias. We used internal documents and web pages mentioned or shared by participants to confirm statements and elaborate on details from interview transcripts. However, we acknowledge that these sources may reflect similar organizational perspectives rather than providing fully independent triangulation.

H. Data synthesis and analysis

We employed thematic analysis [29] to identify patterns and themes in the collected data. We selected thematic analysis because our goal is to derive a view that links technical work (e.g., prompts, metrics, monitoring) with organizational work (e.g., approvals, ownership, governance). This aligns with qualitative analysis, where artifacts are treated as traces of work and coordination rather than isolated documents. We therefore coded interviews and archival artifacts together, recording both the technical content and its context, including role, phase, and decision point. This analysis followed a four-step approach, supported by iterative collaboration and joint interpretation sessions.

- 1) *Initial coding*: A starting set of codes was generated from interview transcripts and archival artifacts. A mixed coding strategy was used. We applied inductive coding to derive adoption phases, steps, and evaluations from practice descriptions. We applied deductive coding when organizing quality concerns, using ISO/IEC 25059 as a naming and grouping index. For interviews, a meaning unit is a participant statement that describes an action, decision, artifact, metric, or role of responsibility. For archival data, a noted unit is a document fragment that defines a rule, checklist item, gate criterion, metric definition, template field, or role ownership statement. Codes were generated based on their relevance to themes that connect to our research questions. For example, codes reflect GenAI adoption (e.g., "prompt piloting"), evaluation (e.g., "risks" and "metrics"), and role responsibilities (e.g., "product owner" and "security"). We coded excerpts when they reported concrete work (e.g., actions, artifacts, or metrics), rather than general opinions without a specific example.
- 2) *Refinement and disambiguation*: The initial codes were iteratively reviewed to improve granularity and avoid semantic overlap. Disagreements were resolved by revisiting the raw excerpt, aligning on the intended meaning, and either refining code definitions or splitting merged codes. For example, "correctness" was split into "output

TABLE IV
EXAMPLES OF CODING PROGRESSION (RAW DATA → CODE → THEME)

Raw data excerpt (Quote/Doc)	Initial code	Sub-theme	Theme (ISO/Process)
“We did not even get to try the model, legal shut it down because of terms.”[P4]	Legal Blocking	Gatekeeping	Ideation Phase / Legal Risk
“Doc-RiskGuardrails: Do not use for personal data unless contract explicitly allows.”	Compliance Rule	Data Constraint	Ideation Phase / Security Risk
“We look at whether it gives the right XML strict syntax is required.” [P1]	Syntax Check	Output Validity	Development Phase / Reliability
“Users complain if it takes >120s to generate they just close the tab.” [P16]	Latency Pain	User Experience	Operations Phase / Performance Efficiency

TABLE V
TRACEABILITY OF MAIN FINDINGS TO DATA SOURCES

RQ	Main finding	Primary evidence source	Traceability description
RQ1	Three-phase adoption process (Ideation, Development, Operation)	Triangulation	Derived from both interview accounts of workflows [P1]–[P19] and process documents ([Doc-ExpGuide], [Doc-GenAI-Portals]).
RQ2	Identification of quality aspects across phases and mapping to ISO/IEC 25059	Triangulation	Identified via interview complaints/challenges and cross-verified with internal checklists ([Doc-RiskGuardrails], [Doc-TestPlan]).
RQ3	Distributed responsibility for quality evaluation	Triangulation	Role assignments extracted from archival data and confirmed by interviewees across phases.
	Need for and proposal of a centralized “GenAI Quality Lead” role	Interviews & Synthesis	Emerged as a synthesized interpretation from interviewees citing vague ownership (e.g., [P2], [P9], [P14]), not present in current archival policies.
RQ4	Process-oriented quality evaluation framework	Interpretive Synthesis	A higher-level synthesis grouping the processes (RQ1), quality concerns (RQ2), and roles (RQ3) into a unified conceptual model.

accuracy” and “code validity” to capture domain-specific distinctions.

- 3) *Theme construction*: We organized codes along two categories: process area (adoption phase and step) and quality dimension (ISO/IEC 25059 characteristics). We formed themes when repeated evidence appeared in both categories. Edge cases were resolved through team discussion to avoid overlap.
- 4) *Mapping to research questions*: Themes were organized according to their relevance to each research question. For example, codes concerning risk evaluations were connected to RQ2, while those related to developer responsibilities were connected to RQ3.

Table IV illustrates our coding progression with four examples from raw excerpts to initial codes and themes. Each row starts from a quote, either an interview excerpt (tagged as [P#]) or an archival fragment (tagged as Doc-*). The “Initial code” column shows the first label assigned to that quote. The “Sub-theme” column groups related initial codes that reflect a shared evaluation practice or concern. The “Theme (ISO/Process)” column links the sub-theme to an adoption phase and, when relevant, the ISO/IEC 25059 characteristic used to organize quality concerns. Together, the rows illustrate both inductive coding for process elements and deductive grouping for quality characteristics.

The first author performed the initial coding and maintained the codebook. The second and third authors reviewed the evolving codes and themes in repeated discussion rounds using shared excerpts and artifact fragments. A shared coding protocol was used to align definitions, coding rationale, and

theme development. Preliminary interpretations were verified through feedback sessions with selected participants to check alignment with their experiences.

I. Steps to derive a quality evaluation framework

Through integrating findings from RQ1 (Process), RQ2 (Quality Aspects), and RQ3 (Roles) into a unified view, we synthesized a quality evaluation framework (Section IV.D). The derivation followed three steps: First, we mapped the identified quality aspects (from RQ2) to the corresponding adoption steps (from RQ1) where they were most frequently observed. For example, “legal risk” was mapped to the “Ideation” phase as a gatekeeping evaluation. Second, we assigned the identified responsible roles (from RQ3) to quality evaluation activities based on interview consensus. Third, we categorized the evaluations into three types: Gatekeeping (pre-development), Validation (development), and Monitoring (operations), forming the framework’s core logic. This synthesis integrates the “when” (process), “what” (quality), and “who” (role) into a single actionable structure.

IV. RESULTS

We present study findings to answer our research questions in this section. **Evidence attribution**: Interview evidence is tagged with participant identifiers [P1]–[P19]. Archival evidence is tagged with document identifiers [Doc-*]. If a claim is supported by both, we include both tags.

Table V provides a summary mapping of these main findings to their empirical origins, indicating whether the insight stems primarily from interviews, archival data, triangulation of both, or higher-level interpretive synthesis.

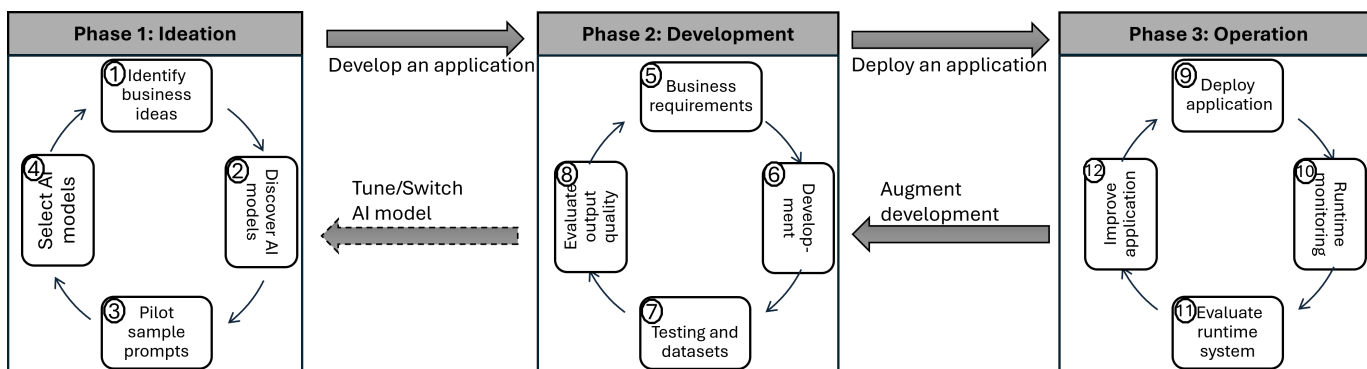


Fig. 2. Phases and steps followed to adopt GenAI.

A. Results of RQ1 – What processes do individual engineers follow when adopting GenAI in software products?

Fig. 2 illustrates the identified phases of GenAI adoption and use in software products. The process is organized into three main phases: *Ideation*, *Development*, and *Operation*. These phases reflect a combination of exploratory use, iterative design, and deployment-oriented practices.

1) *Phase 1: Ideation*: This phase is initiated by identifying high-level business opportunities (Step 1). The ideas are typically sourced by product teams, often without strong input on technical feasibility at this stage [P3, P8, P17]. This reflects a trend in which business-driven interests precede technical alignment [P5, P7, Doc-GenAI-Portals].

Once candidate ideas are selected, engineers explore potential GenAI solutions by surveying available models (Step 2). At this stage, engineers screen for early fit and risk rather than formal accuracy. They verify task fit (outputs align with the intended task and domain), data and policy fit (inputs comply with privacy and security rules), the legal acceptability of the license, basic output quality on a few representative prompts, and integration feasibility (including API access, hosting, and deployment path) [P5, P13, Doc-ModelSelection].

Piloting sample prompts (Step 3) serves as a lightweight feasibility check. It enables teams to evaluate whether GenAI output aligns with expectations before making further investments. In the studied companies, this step was often used to discard ideas quickly, reflecting a low-cost experimentation practice [P12, Doc-ExpGuide].

If the output appears promising, teams proceed to select a specific AI model (Step 4). Selection is commonly based on empirical prompt testing, vendor trust, and ease of integration, rather than formal quality guarantees [P5, P19, Doc-ExpGuide]. This also implies that model selection is frequently revisited as the project progresses.

2) *Phase 2: Development*: In the development phase, the focus shifts from ideation to system integration. Teams refine business requirements (Step 5) to accommodate GenAI capabilities and constraints, often reducing the granularity of traditional specifications [P15, P18]. In contrast to conventional software, requirements are shaped around acceptable variation in model output, making this step inherently iterative.

The development step (Step 6) involves incorporating the selected GenAI model into an application or service. This

may include prompt engineering, workflow orchestration, or API-level integration. Participants noted that development often advances without complete clarity on evaluation criteria, reinforcing the need for later adjustments [P4, P9].

Testing and dataset preparation (Step 7) vary considerably in formality. Some teams curate specific test cases to validate expected behaviors, while others rely on production-like data to simulate real-world input [Doc-TestPlan]. The level of rigor here depends on the perceived risk and target user.

Output evaluation (Step 8) is where the prototype is reviewed for its usefulness, correctness, and consistency. Unlike deterministic software, evaluation involves human-in-the-loop judgment and often lacks predefined acceptance thresholds [P1, P5, P10]. This introduces challenges in comparing iterations or tracking progress over time.

3) *Phase 3: Operation*: After deployment (Step 9), systems enter the operations phase, where GenAI becomes part of production environments. This transition often marks a shift in responsibility from development teams to maintenance and operations teams [P6, P8, P14, Doc-DevOps].

Runtime monitoring (Step 10) is implemented to track anomalies, drift, or failure cases. However, monitoring practices remain immature; few teams employ systematic mechanisms for runtime evaluation beyond basic usage statistics [P2, P16, P18].

Evaluation of the runtime system (Step 11) focuses on stability, user satisfaction, and reliability in production contexts. This step is typically reactive—issues are addressed after surfacing, rather than through predictive assessments [P15, P17, Doc-DevOps].

Improvement cycles (Step 12) are driven by operational feedback and usage data. These iterations may involve switching models, re-engineering prompts, or redesigning workflows [P1, P7, P16, Doc-ModelSelection]. However, the improvement process is rarely governed by formal change management, reflecting the still-exploratory nature of GenAI operations.

4) *Observations*: We observed that GenAI adoption in industry follows a typical pattern of iterative experimentation, model probing, and incremental discovery, similar to how organizations adopt other new technologies [P2, P12, P19]. This reflects a shift from deterministic system design toward

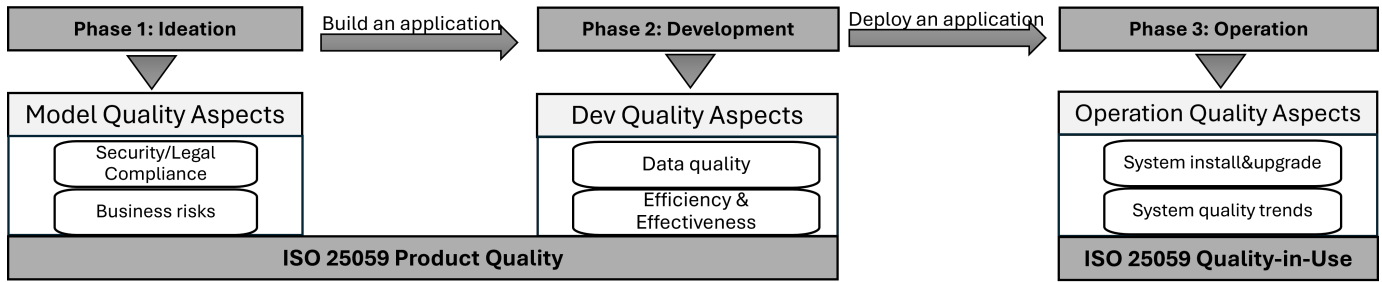


Fig. 3. Primary quality aspects considered throughout GenAI adoption and use, mapped to ISO/IEC 25059.

empirical convergence, where systems evolve through usage feedback and feasibility trials.

Moreover, we learned that quality evaluation occurs in most phases, but practices are often informal and subjective [P4, P8, P18] (e.g., accepting outputs without predefined thresholds for accuracy, latency, or safety). Practices such as prompt piloting, runtime observation, and prototype judgment serve as implicit gates for decision-making. This raises concerns about traceability and accountability in GenAI-based systems.

For practitioners, the identified three phases can serve as a reference to position their own efforts. It also offers researchers an example of how GenAI adoption unfolded in two industrial settings.

B. Results of RQ2 – What quality aspects are evaluated across GenAI adoption and use in software products?

We report the quality concerns that emerged from our empirical data. Fig. 3 summarizes the quality aspects identified by participants across the three phases of GenAI adoption and use. At the end of each phase, we present “Proposed Actions” that serve as the foundation for the evaluation framework proposed in this paper.

Quality aspects in Phases 1 and 2 relate to *Product Quality*, and Phase 3 focuses on operation, which falls under the *Quality-in-Use* characteristics as defined in ISO/IEC 25059. These aspects reflect where practitioners currently focus their attention, shaped by their roles and organizational constraints [P3, P7, P13, P16, Doc-GenAI-Portals].

1) Model Quality Aspects (Phase 1) — ISO 25059 Product Quality:

a) *Security and legal compliance*: Participants emphasized the importance of understanding legal implications early, particularly regarding intellectual property, data privacy, and third-party model licensing [P9, P10, P11, P12, P13, P14]. This concern is not only about selecting ‘safe’ models but also about whether the use case is legally viable [P10, Doc-RiskGuardrails]. The motivation stems from the uncertainty around AI model behavior and ownership of training data, which introduces legal risks. Practitioners noted that without early legal validation, entire business ideas may later be blocked or delayed [P9, P11, P12, P14, Doc-ExpGuide].

b) *Business risks*: Feasibility was often framed in business rather than technical terms, for example, unclear value propositions, unpredictable behaviour in user-facing contexts,

or prohibitive costs [Doc-ExpGuide, Doc-GenAI-Portals, Doc-DevOps]. Participants noted that even technically functional GenAI models might conflict with user trust or supportability goals [P2, P3, P5, P7, P16]. As a result, early-phase quality evaluation often prioritizes business viability alongside compliance.

c) *Proposed Actions*: At this phase, quality evaluation focuses on compliance and purpose. Organizations are advised to involve legal, compliance, and product strategy teams early to assess feasibility before prototyping.

2) Development Quality Aspects (Phase 2) — ISO 25059 Product Quality:

a) *Data quality*: Data quality emerged as a central concern. Participants were uncertain whether test datasets and prompts accurately reflected real-world adoption, especially when outputs were highly sensitive to subtle input changes. In some teams, overfitting prompts to hand-crafted cases reduced generalization after deployment [P1, P2, P4, P15, Doc-ExpGuide, Doc-TestPlan].

b) *Efficiency and Effectiveness*: Effectiveness was interpreted contextually: some defined it as “acceptable output for task completion”, others included time-to-answer or resource efficiency (e.g., latency, token cost). Formal benchmarks were rarely used, according to participants. Instead, teams used quick iterations with subjective judgment [P2, P3, P5, P6, P7, P18, Doc-ExpGuide, Doc-GenAI-Portals]. While agile, this informality makes scaling and repeatability difficult.

c) *Proposed Actions*: Informal assessments should be complemented with measurable criteria, for example, output consistency, processing time, and task success rates. Data quality should be revisited during development, not assumed from earlier checks.

3) Operation quality aspects (Phase 3) — ISO 25059 Quality-in-Use:

a) *System install and upgrade*: Participants reported deployment friction for GenAI components due to version mismatches, hosting constraints, and unclear dependencies [P1, P3, P4, P5, P7, P18, P19]. Such issues are more disruptive for GenAI than traditional systems, given their dynamic dependencies and licensing restrictions [Doc-RiskGuardrails, Doc-TestPlan, Doc-DevOps].

b) *System quality trends*: Monitoring quality trends in production was valued but underdeveloped [P3, P5, P15, P17, Doc-DevOps]. Most participants lacked formal systems in place to track model drift, unexpected failures, or changing

behavior across environments. Instead, most relied on user complaints or ticket reports from support teams. This limits feedback loops and hinders root-cause analysis [P6, P7, P8, P17, Doc-DevOps].

c) *Proposed Actions*: Practitioners should treat GenAI-based components as evolving artifacts. Continuous monitoring, logging, and sampling should be used to detect unexpected changes, supported by feedback channels from users and operations teams.

4) *Observations*: While the identified aspects align with ISO/IEC 25059, several characteristics remain under-addressed:

- **Functional adaptability** — system responsiveness to dynamic environments or evolving data.
- **Robustness** — stable performance under bias, adversarial inputs, or novel conditions.
- **Transparency** — clear documentation of AI models' logic, data provenance, and decision rationale.
- **Societal and ethical risk mitigation** — adherence to fairness, privacy, and accountability principles.

During the interviews, these quality characteristics were rarely mentioned unless specifically asked about. For instance, *functional adaptability* – an important reason for choosing GenAI in the first place – was not systematically evaluated. *Robustness* and *transparency* were typically implicit or informal (e.g., “seems to work well”) rather than captured through structured criteria or logging. Without explicit evaluation, systems may degrade in unobserved ways, particularly after deployment [P15, Doc-DevOps, Doc-TestPlan].

Engineering teams should integrate selected ISO/IEC 25059 attributes into lightweight quality evaluation checklists or design review templates. The following aspects are actionable:

- **Transparency**: Document prompt structures, data origins, and model versioning. Use structured logging to maintain prompt output behavior over time.
- **Robustness**: Evaluate and guard against structured-output (e.g., JSON/XML) failures. Use schema validators (e.g., JSON Schema, XSD), constrained decoding or typed function calling, and strict parsers. Add automatic repair, enforce timeouts, and use retries with backoff and circuit breakers. Define fallbacks for known failure cases (e.g., cached answers, rule-based generation, or human-in-the-loop).
- **Societal and ethical risk mitigation**: Align GenAI adoption with Responsible AI frameworks [30]. Include non-engineering stakeholders early, such as legal, security, compliance officers, domain experts, and representative users.

These steps do not require full ISO compliance but can raise quality awareness and connect daily GenAI practice with standardized guidance.

C. *Results of RQ3 – Who within the engineering organization is responsible for evaluating quality throughout GenAI adoption?*

GenAI integration into software products creates new quality challenges, with responsibility for addressing these challenges remaining fragmented across roles. Fig. 4 summarizes

the roles identified in each phase of GenAI adoption—ranging from business ideation to runtime operations—and the specific quality concerns associated with them.

1) *Responsible roles*: In the **Ideation** phase, responsibility is distributed among business owners, legal and licensing teams, and security units. Business owners assess the viability of use cases and business risks, while security and legal experts validate compliance with regulations and intellectual property policies when selecting models [P5, P18, P19, Doc-ModelSelection]. As a participant stated: “We had to get the legal team involved early because we weren’t sure if the use case was even allowed with the data we had [P9].”

In the **Development** phase, developers and architects lead integration, while quality engineers and analysts support testing and validation. Yet evaluation practices often remain informal or rely on domain intuition [P2, P6, P7, P8, Doc-GenAI-Portals]. Several participants reported: “It is mostly trial and test prompts, tweak outputs, and we go with what feels right [P16].”

Some teams assign evaluation duties to developers by default, and in the absence of structured criteria, they rely on domain expertise and human judgment. Several participants [P4, P5, P18] noted: “There is no fixed checklist. We just look at whether the response makes sense or is dangerous.”

In the **Operation** phase, roles such as infrastructure engineers and support analysts become involved. However, runtime feedback mechanisms are often weakly connected to earlier evaluation decisions [P15, P16, P17, Doc-DevOps]. Participants [P7, P15, P18] stated that “We rely on support tickets to learn when something’s off. There’s no automated alerting tied to model output yet.”

2) *The need for a dedicated role in ownership*: Across all phases, we found limited unified ownership for GenAI quality evaluation. Although responsibilities are scattered among domain experts, the lack of coordination and traceability reduces the accountability and repeatability of evaluation activities. For example, a participant stated: “Everyone is CERTAIN that SOMEONE is checking the quality, but no one knows who actually does it [P1].” This ambiguity is not unique to GenAI. Conventional projects may also suffer from diffuse quality ownership in siloed organizations [P7, P8, Doc-TestPlan]. GenAI magnifies the problem because quality decisions span multiple areas, including legal, security, compliance, and software development.

A recurring theme in interviews was the absence of a designated role that integrates technical, legal, and ethical concerns across the GenAI adoption and use [P2, P9, P11, P14, Doc-RiskGuardrails]. As participants [P4, P5, P6, P10, P19] repeated that “It would help to have someone who knows what to evaluate, not just technically, but in terms of risk or impact too.” Based on this finding, we outline a recommended *GenAI Quality Lead* role in the Discussions section, alongside alternative organizational solutions.

Responsibilities of the quality lead include:

- Facilitate early-phase risk identification (e.g., legal, security, bias).
- Establish and adapt quality criteria with legal, security, and domain stakeholders.

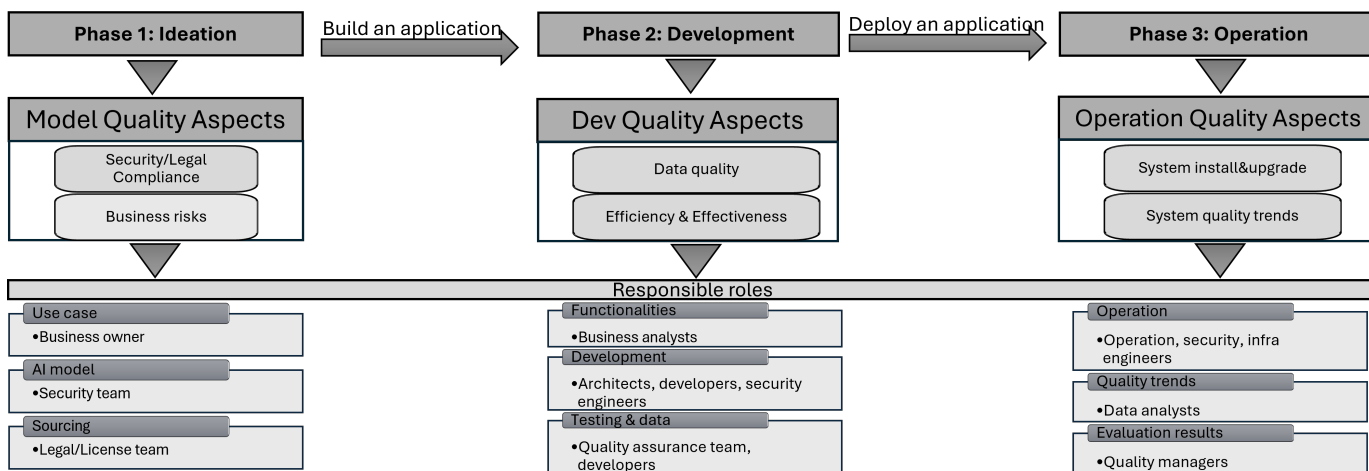


Fig. 4. Responsible roles for quality evaluation throughout GenAI adoption.

- Defining evaluation criteria with product and engineering teams.
- Drive evaluation practices across development and operations (e.g., output testing, runtime monitoring).
- Translating quality standards (e.g., ISO/IEC 25059) into project-specific practices.
- Serve as a contact point for review boards, audits, and compliance alignment.

This role builds upon principles of SE4AI, which emphasizes quality assurance throughout the AI engineering lifecycle. By formalizing this role, companies can better integrate emerging standards (e.g., ISO/IEC 25059) and internal Responsible AI guidelines into their everyday practices. Importantly, the role is not a replacement for domain-specific experts, but rather a coordinating role that enhances quality, accountability, and traceability. It helps move beyond ad hoc evaluations by embedding structured thinking into the day-to-day engineering of GenAI-based solutions [P2, P5, P18, P19, Doc-TestPlan, Doc-ExpGuide].

D. Results of RQ4 – How does the organization conduct quality evaluation throughout GenAI adoption?

Based on our findings from RQ1 to RQ3, we synthesized a process-oriented framework that integrates GenAI adoption phases with quality evaluation practices and role responsibilities. This framework illustrates HOW quality is evaluated during GenAI adoption in phases and indicates WHO is accountable.

Fig. 5 consolidates our findings into a process-oriented view of GenAI adoption, with quality evaluations integrated into each phase and corresponding roles.

Each phase in the figure contains steps (white boxes), among which specific actions related to quality evaluation are highlighted. The ★ marks indicate points where evaluation is **directly enabled**, for example, assessing legal risks, verifying outputs, or analyzing system qualities. The ♦ marks identify **supporting evaluation activities**, for example, model prompting or installability that shape or constrain quality assurance

decisions. Together, they form a network of quality signals – some proactive, others reactive.

Below each phase, the gray panels describe evaluation practices, which range from compliance vetting to metric-based evaluations. At the bottom, the roles responsible are connected, revealing a distributed accountability structure where legal, engineering, and quality teams are all involved at different stages.

1) *Quality evaluation practices in GenAI adoption:* Based on collected data, three interconnected evaluations emerge: Gatekeeping – evaluation before development, Validation – testing during development, and Monitoring – evaluation during operation.

a) 1. *Gatekeeping (before development):* Evaluations in the Ideation phase are often conducted to determine whether a GenAI model or idea can be pursued [P6, P7, P18, Doc-ExpGuide, Doc-RiskGuardrails]. These gatekeeping controls are typically compliance-driven, involving legal and security teams.

The checklist in Table VI was referenced by multiple participants as a tool to assess risk exposure and vendor accountability before prototyping. Participants from legal teams [P9, P10, P11] mentioned: “We block use cases if the supplier can’t commit to data privacy or copyright terms.” These evaluations serve as safeguards, especially in large organizations where regulatory pressure is high.

b) 2. *Validation (during development):* In the development phase, evaluation activities shift from permission to execution. Here, teams examine if an AI model behaves appropriately in their specific settings. This includes assessing data quality, response consistency, efficiency, and effectiveness in relation to internal policies. We found that evaluation practices often lack mature tooling and rely on custom metrics [P1, P6, P7, P8, Doc-TestPlan, Doc-GenAI-Portals]. For example, a senior developer shared that “We often create our specific key performance indicators in our implementation, aiming to give a signal on hallucinations [P3].”

Table VII outlines a cross-functional practice that was used in multiple settings to structure quality assessments during

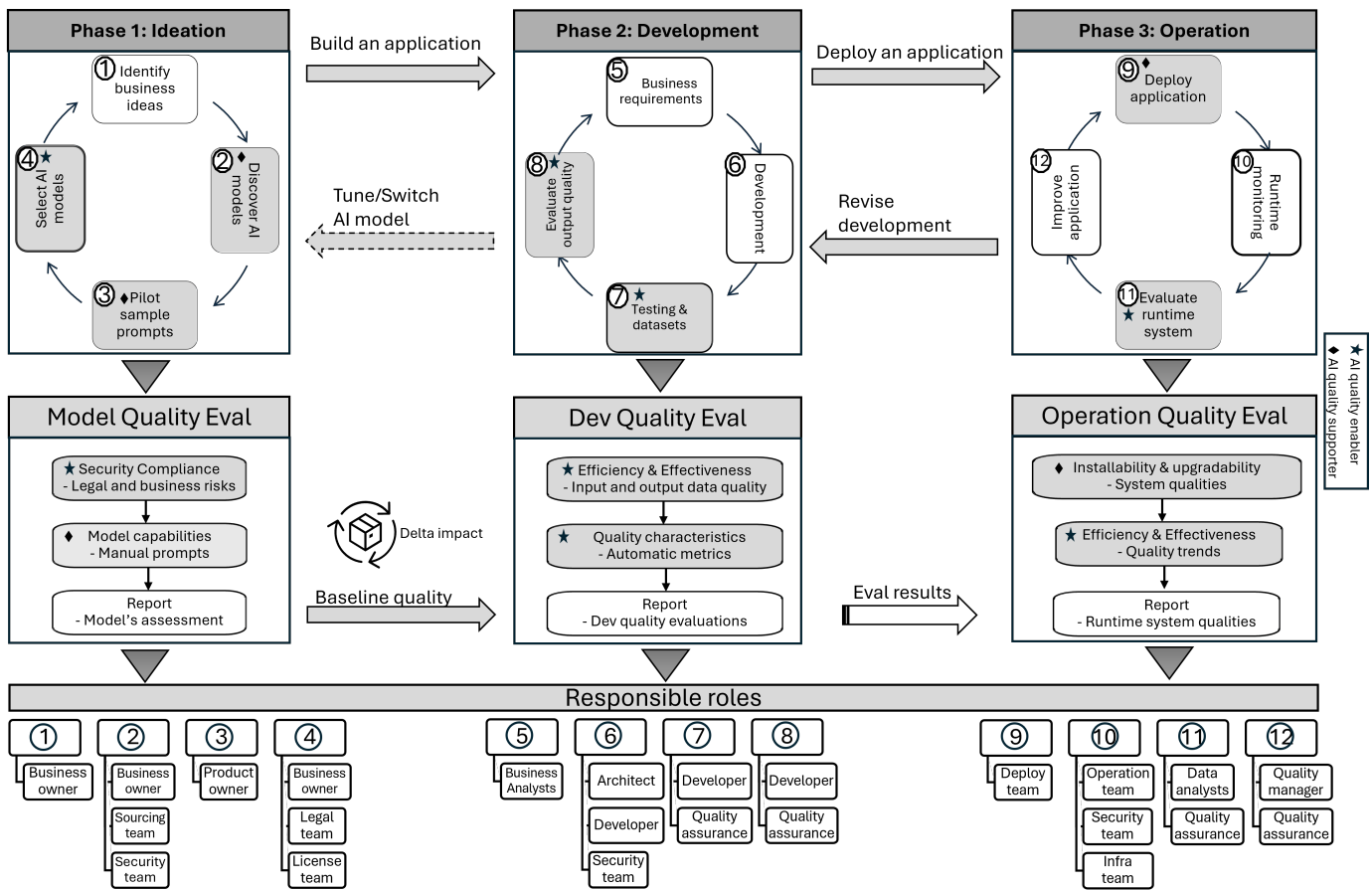


Fig. 5. Overview of quality evaluation framework during GenAI adoption in software products.

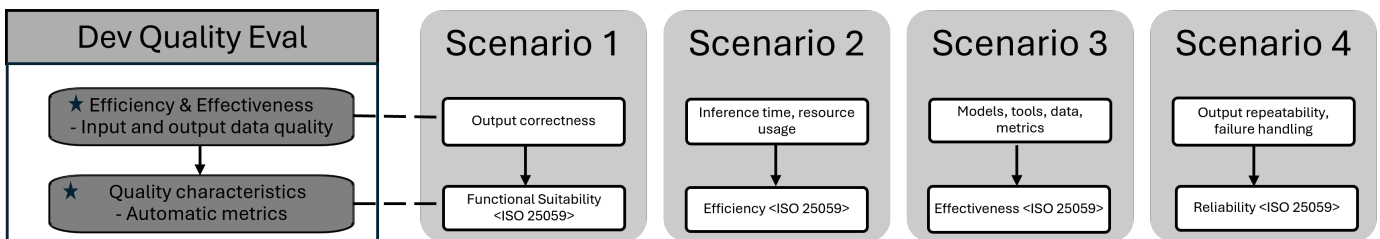


Fig. 6. Relation between observed Dev Quality Evaluation practices and ISO quality characteristics.

implementation. These assessments are rarely isolated. They are tightly coupled to development iterations.

To better understand the relationship between the observed practices and software quality, we mapped the evaluation activities to ISO/IEC 25059 characteristics in Fig. 6.

As shown in Fig. 6, observed practices (e.g., prompt testing, the use of tools, and metrics) map implicitly to ISO quality characteristics. Specifically, they reflect:

- **Functional Suitability.** It includes the correctness and appropriateness of output.
- **Efficiency.** It covers measures for inference time duration and resource usage.
- **Effectiveness.** It covers task success and utility in the target context, assessed through targeted tests and usage evaluation.
- **Reliability.** It refers to output repeatability and failure

handling.

These mappings indicate *what* to evaluate; they do not prescribe *how*. In our cases, metrics were selected per context, and some remained unsettled. Future tooling and policy definitions may benefit from translating these ISO sub-characteristics into practical evaluations and observable metrics.

c) 3. **Monitoring (during operation):** Evaluation does not end at deployment. Once a system is deployed successfully, teams perform runtime assessments based on logs, live quality trends, and user feedback [P15, P17, Doc-DevOps]. This mode focuses on identifying performance degradation, systemic errors, or violations of usage policies. As shared by an operations engineer, “*It is required that we detect issues not when they happen, but when someone flags them downstream [P16].*” These practices are increasingly important in GenAI, where behavior can drift over time. Importantly, monitoring

TABLE VI
LEGAL RISK ASSESSMENT CHECKLIST AND OWNERS

Legal risk type	Evaluation(s)	Responsible
Intellectual property infringement	Who is liable for infringement of copyrights in training data, e.g., the supplier or user?	Legal and Security teams
Training data	What data can be used to train/tune AI models?	Legal team
License	What Open Source Software has been used to develop the model(s), and which license applies?	Licensing team
Inputs	How do the AI suppliers treat users' confidential information, e.g., publicize user inputs or user inputs used to train AI models?	Legal and Security teams.
Outputs	Who is legally liable for the output of AI models? The supplier may disclaim all liabilities for use of their models	Legal team
Data privacy	Who is responsible for any illegal processing of personal data during the adoption of AI?	Legal and Security teams
	Can the AI model provider guarantee that the user data is not used to train the model, or in any other way processed by the provider of the system, or other third party for other purposes?	
	For any use cases of AI leading to recommendations, conclusions, or predictions related to individuals, can data protection for individuals be mitigated from the legal aspect?	

TABLE VII
QUALITY EVALUATIONS FOR GENAI APPLICATION IMPLEMENTATION

Development tasks	Evaluation(s)	Responsible
Information Security	How are objectives on confidentiality, integrity, and availability of information managed?	Security team
Data governance	How to handle data quality issues, unauthorized access, and improper use of data?	Security team
Contractual Frameworks	What are the terms of use for both internal AI adoption and for delivery towards the customer?	Legal team
Intellectual Property Rights	Avoid leakage of confidential information.	Licensing team
	No licensing out of patents.	
	Strict adherence to open source policies.	
Product Security	No use if no full control over AI tools/services.	Architect team
	A small and simple proof-of-concept is required.	
	AI is a support tool, not a decision maker.	
	Not be solely used for sensitive functionalities where incorrect outcomes cannot be tolerated.	
Data privacy	Complying with data protection principles/laws.	Legal team
	Processing user data only according to the customer contracts.	
	Maintaining records of processing use cases is a regulatory requirement.	
	No violations of individuals' rights and freedoms.	
Ethical and responsible outcomes	For in-house-built models, the training data should be explained and evaluated for bias.	Product owner Development team Quality team Management team
	For sourced models or tools, vendors must provide information about their bias analysis.	
	The ethical and responsible consequences of model error should be evaluated for the use case	
	Human-in-the-loop controls should be implemented.	
	Analyze all AI use cases in terms of their compliance with regulations. Ban those prohibited by law.	
Quality of outcomes	Classify AI use cases as limited- or high-risk under the AI Act.	Product owner Development team Quality assurance Management team
	PoC use cases to determine the seriousness of erroneous responses.	
	Avoid use cases where the consequences are serious or no opportunity to check the response before action is taken.	
	Implement controls on the model training (data quality).	
	Inform the user about the possibility of an error and the appropriate caution to take.	
	Use application designs that allow source checking with source references.	
Provide explainability tooling to help users understand how answers were determined.		
	Implement monitoring, response and support mechanisms for dealing with errors detected during production use.	

efforts feed back into development and legal reassessments, completing the lifecycle.

The integrated view in Fig. 5 reveals a decentralized architecture of quality assurance in GenAI adoption. Evaluation is not owned by a single role or function. It is distributed, contextual, and embedded in day-to-day decisions. Tables VI and VII reflect how companies attempt to formalize what would otherwise remain tacit.

However, several challenges remain: metrics are inconsistent, practices vary across teams, and accountability is shared but unclear. Some participants described this as a lack of a coherent QA strategy tailored to GenAI, “*We have a security master in each dev team. But who is the quality champion for GenAI? [P7]*”

This observation motivates the discussion of a new role in

Section VII-B, and underscores the practical value of treating quality evaluation not as an afterthought, but as an ongoing design activity.

2) *Model reuse*: The quality evaluation framework presented in Fig. 5 offers a reusable structure for organizations seeking to evaluate GenAI-enabled software.

By clarifying when and how evaluations are conducted, and by whom, the overview can serve as a reference for organizations lacking established quality practices for GenAI. It also helps bridge informal routines and ISO-inspired evaluation dimensions, as illustrated in Fig. 6. Therefore, the framework is a synthesized process reflecting actual engineering practice.

To further examine its practical utility, we conducted a case implementation in one of the studied companies. The aim was to apply the evaluation structure to a real GenAI adoption and

assess its feasibility. The next section presents this framework verification case.

V. VERIFICATION OF THE FRAMEWORK

The goal of this verification is to demonstrate how the framework can be instantiated in a real-world project by translating the abstract quality characteristics into project-specific evaluations, metrics, and role assignments. We focus on feasibility and traceability by verifying whether teams can perform the necessary evaluations using available artifacts (test suites, validators, dashboards, and logs), and whether the resulting evidence supports the adoption of GenAI and post-deployment monitoring. The verification case was conducted in Company B.

A. Case description

The case was a GenAI-assisted application for generating XML code used in Unstructured Supplementary Service Data (USSD) menus. USSD is a widely used communication protocol in mobile services that enables interaction through short codes, eliminating the need for internet access. The target users of this tool were merchants who needed to create dynamic menus for product browsing and purchasing without manually writing XML code. The tool enabled these users to provide natural language descriptions of their menu logic, which the system converted into valid, executable XML files for integration into mobile financial platforms.

This project was selected as a verification case for two reasons. First, it represents a realistic and commercially relevant adoption of GenAI, involving code generation in a critical domain. Second, it required cross-role collaboration between developers, quality assurance engineers, legal reviewers, and product owners, reflecting the multi-stakeholder structure of GenAI adoption.

B. Operationalization of the Framework

The development process was structured around the phases and evaluation steps outlined in the quality evaluation framework. To apply the framework, the team first translated the abstract ISO/IEC 25059 quality characteristics into concrete project metrics (Table VIII), such as semantic alignment score, XML validation pass rate, average generation latency, and compliance sign-off. This step was critical to move from “quality” as a vague concept to measurable acceptance criteria.

TABLE VIII
OPERATIONALIZATION OF QUALITY CRITERIA FOR USSD CASE

Quality characteristic	Project-specific definition	Metric / Acceptance criteria
Functional Suitability	Does the generated XML match described menu logic?	Semantic alignment score > 4.5/5 on test set
Reliability	Is the generated XML valid and executable?	XML validation pass rate > 92%
Performance Efficiency	Is the generation fast enough for real-time editing?	Average generation latency < 45 seconds
Security Risk	Does the model generate prohibited content?	Compliance sign-off

With the project-specific quality criteria definitions in Table VIII, the framework verification process was executed through three phases, as shown in Fig. 7.

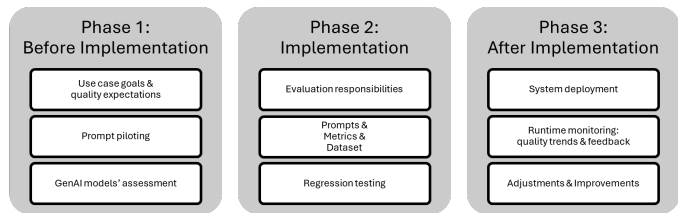


Fig. 7. Mapping of the USSD project activities to the framework's quality gates.

1) *Phase 1: Ideation (Before Implementation)*: The *Ideation Checkpoint* was conducted by the Product Owner and Legal Team. They evaluated *Security Risk* by reviewing the data flow. Since the tool would process merchant inputs but no end-user personal data, the risk was classified as low, provided no training occurred on customer data. Model selection (GPT-4o vs Llama 3) was evaluated based on the *Functional Suitability* of initial prototypes.

2) *Phase 2: Development (During Implementation)*: During development, the *Development Checkpoint* focused on *Reliability* and *Performance Efficiency*. Developers integrated a validator that automatically checked every generated XML against the schema. The QA team ran a regression suite of 50 complex menu descriptions. Metrics were tracked daily: initial pass rates were low (60%) but improved to 92% after prompt engineering refined the XML structure instructions.

3) *Phase 3: Operation (After Implementation)*: Post-deployment, the *Operation Checkpoint* involved continuous monitoring via Grafana. The Operations team tracked *Performance Efficiency* (latency) and user feedback (implied *Functional Suitability*). An alert threshold was set for latency spikes (> 45s), which would trigger a fallback to a smaller, faster model.

C. Verification outcomes

The proposed framework was applied to all three phases of the USSD XML Generator case. It enabled the team to address quality concerns with defined responsibilities and metrics.

In the early phase, the model assessment helped filter out unsuitable options and anticipate risks. Legal and security engineers used the checklist to validate compliance concerns before deployment. One engineer noted, “The checklist turned vague worries into actionable points, such as licensing, datasets, inputs, and outputs.” During development, quality metrics were embedded into the implementation process. Accuracy and latency were measured continuously using in-house scripts and Prometheus monitoring [31]. A quality assurance engineer remarked, “The quality characteristics gave us criteria early, so we knew what to test and why.” Operational monitoring added a feedback loop post-deployment. Monitoring dashboards showed runtime trends that directly informed model prompt adjustments. “Seeing the correctness rate of generated outputs improve week by week gave us real confidence,” said an operations engineer.

Despite its practical value, the framework had limitations during the case implementation. First, the success of its application relied on the willingness and capacity of team

members to take ownership of additional evaluation tasks. Second, although the framework included evaluation practices with ISO alignment, the evaluation results still depended on the quality of test data and prompt engineering, where guidance was limited. Third, the monitoring approach, while useful, required specific tooling and metrics setup that may not generalize to other organizations.

We position the framework as a meta-level structure with a stable core (checkpoints, role duties, and a minimal metric set) and configurable parts (attributes, datasets, and safeguards). Teams can instantiate it per feature or domain using shared templates and steps, rather than building a new method or bringing new tools each time. In the USSD case, the core metrics were reused; prompts and test data were tailored.

VI. VALIDITY THREATS

We follow Runeson et al.'s guideline to address validity threats: construct, internal, external, and conclusion validity [32].

a) Construct validity: To identify quality evaluation practices in GenAI adoption, we conducted 19 interviews with practitioners who had hands-on experience with GenAI-enabled software products. Participants were selected from multiple roles, including development, testing, architecture, legal, and management, to ensure diverse perspectives. Although each participant had experience with parts of the entire process, their combined accounts provided a holistic view across its constituent steps. However, participants may have used inconsistent terminology or lacked shared definitions of “quality,” “evaluation,” or “GenAI adoption.” To mitigate this, we clarified questions during interviews (e.g., “What did your team do to ensure the model was suitable?”), followed by clarifying questions and validation of summaries during transcription checks.

b) Internal validity: Our thematic analysis followed a staged coding process, including initial open coding, coding for pattern identification, and iterative refinement. Multiple researchers reviewed the coding structure and figure derivation to minimize personal bias. However, participant statements may still include misinterpretations. We mitigated this through three group sessions to review collected artifacts (e.g., internal documents and metrics). For the verification case, the same research team collaborated with practitioners during implementation, which may introduce bias toward confirming our proposed framework. To mitigate this, study participants interpreted evaluation metrics and improvement steps.

c) External validity: Because participants from two selected companies work within the same corporate group, their accounts may reflect shared governance norms and vocabulary, so complete independence of perspectives cannot be guaranteed. We mitigated this by sampling across roles, sites, and functions and by triangulating interview accounts with archival artifacts, but we still interpret the findings as reflecting this organizational setting and discuss transfer limits accordingly.

The framework was derived in a setting with development teams, in-house legal and security support, and cross-site collaboration. Organizations that lack these prerequisites, such

as smaller teams without dedicated risk or operations functions, may need to adapt roles, activities, and metrics before adoption. Therefore, we present the framework as process-oriented guidance rather than a universal prescription, and we distinguish elements from configurable ones to aid tailoring. The verification case demonstrates feasibility in the studied context, but applying the framework elsewhere will require adaptations to local constraints.

d) Conclusion validity: We documented research procedures, coding steps, and analytical decisions to support transparency. The first three researchers analyzed the data and resolved differences to ensure reliability. Participants' feedback was used to validate our interpretations of key themes.

We acknowledge potential threats to conclusion validity. For example, if the data concerning legal evaluation practices were incomplete or misinterpreted, an alternative conclusion might be that legal responsibilities are decentralized rather than coordinated through a central unit. However, our collected data showed that legal evaluation was explicitly addressed by multiple participants. Furthermore, our multi-step verification, which includes peer checking, participant feedback, and triangulation, reduces the risk of such misinterpretations.

Additionally, we reported both positive outcomes and challenges during the verification of the proposed framework, which aims to mitigate confirmation bias. Future studies are needed to understand the framework's long-term effects in other industrial settings.

VII. DISCUSSIONS

A. Legal risk and security compliance come first?

In our cases, preliminary legal and security checkpoints blocked or delayed pilots. The reasons included supplier terms that allowed training on user inputs, unclear ownership of generated code, and data-residency constraints that conflicted with company policy (Table VI). While many engineering activities are iterative and exploratory, legal and compliance violations can trigger irreversible consequences, such as breach of contract, regulatory sanctions, or reputational damage. This risk profile justifies the placement of legal and security evaluation as the first quality gates in our framework.

The necessity of prioritizing these concerns became evident in our interviews. As one participant noted: “We did not even get to try the model, legal shut it down because of the terms of service.” Unlike performance issues or usability flaws, legal violations typically cannot be patched silently once a product reaches customers. This means that the cost of deferred evaluation is not technical debt, but rather organizational risk. Treating these issues as upstream concerns helps ensure that software built on GenAI is legally viable and security-resilient before other quality aspects are even meaningful.

Additionally, the placement of these assessments early in the workflow promotes shared accountability across teams. Rather than delegating compliance evaluation to external gatekeepers, integrating them into quality evaluation encourages engineers to consider these aspects as part of their technical responsibility. This shift from reactive compliance to proactive

governance requires additional resources to address risk and compliance evaluation.

B. Introducing a GenAI quality lead role

Our findings show that quality evaluation responsibilities in GenAI-enabled software products are distributed across several functional roles, including legal teams, developers, security engineers, analysts, and quality managers. While each role addresses a specific concern—such as compliance, technical performance, or operational monitoring—no designated function oversees quality holistically across the entire lifecycle.

This absence of centralized responsibility leads to fragmentation in evaluation practices and limited traceability of decisions. Practices such as legal risk assessment or runtime evaluation are conducted in isolation, without a unifying perspective that ensures GenAI-specific quality concerns are systematically addressed.

To mitigate this gap, we propose the establishment of a dedicated role: the *GenAI Quality Lead (GQL)*. The GQL is envisioned as a cross-functional role that ensures quality considerations are proactively embedded throughout the GenAI adoption lifecycle—from model selection and data preparation to deployment and continuous operation.

Specifically, the GenAI Quality Lead would coordinate the following responsibilities:

- Collaborate with legal and security teams during model sourcing to address compliance and licensing risks.
- Work with developers and analysts to define evaluation criteria aligned with business and technical requirements.
- Ensure structured assessment of output quality and runtime behavior, drawing on relevant metrics and feedback loops.
- Facilitate alignment with external standards, such as ISO/IEC 25059, by translating quality characteristics into actionable practices.

This role draws conceptual support from the SE4AI (Software Engineering for AI) perspective, which emphasizes the need for dedicated quality assurance practices tailored to AI-based systems. The GenAI Quality Lead does not replace existing domain experts, but rather functions as a coordinating actor that integrates diverse quality concerns into a consistent and traceable process.

Establishing such a role may enhance an organization's capability to evaluate GenAI applications using measurable metrics. By providing transparent quality metrics, this role addresses the “trust” and “risk perception” barriers identified in recent adoption studies [25] [24].

This role also helps bridge the gap between “Product Quality” (e.g., accuracy) and “Quality-in-Use” (e.g., user satisfaction). Organizations often focus on static product metrics, yet adoption failure often stems from poor quality-in-use. The GQL ensures that operational feedback loops explicitly measure these dynamic aspects, thereby preventing the trust gap that arises when technically suitable models fail in practice. While proposing a GenAI Quality Lead explicitly addresses the coordination gaps we observed, organizations may explore alternative solutions depending on their structure. For instance,

rather than creating a dedicated new role, companies could establish cross-functional AI Review Boards to periodically evaluate GenAI projects at critical phase gates. Alternatively, organizations might integrate GenAI-specific evaluation criteria into the existing duties of traditional Quality Managers, aligning GenAI quality assurance with established Quality Management Systems (QMS) and ISO standards. The essential requirement is that coordination of GenAI quality becomes a deliberate, centralized responsibility, regardless of the specific organizational form.

C. Practical Implications

Organizations can use the framework to transition from ad-hoc prompting to a structured quality assurance process. We recommend a three-step instantiation approach:

- 1) **Map existing gates:** Identify where current quality evaluations (e.g., legal/security) occur and overlay the framework's processes. For GenAI adoption, the ideation gate is critical to filter high-risk use cases early.
- 2) **Define the GQL responsibilities:** Even without a dedicated new hire, assign the “GenAI Quality Lead” duties to a senior developer or architect. This ensures clear ownership of model selection and prompt versioning.
- 3) **Operationalize metrics:** Replace vague goals like “be helpful” with measurable proxies, such as “user edit rate” (for code generation) or “latency < 15s” (for chat interfaces), as demonstrated in our verification case.

This structured approach reduces accumulated evaluation issues when teams experiment without defined success criteria.

Researchers outline that *trust* [25] [24] and *workflow compatibility* [23] [22] are primary drivers of GenAI adoption. Our findings extend this by showing that *organizational trust* is fragile without structured evaluation. The framework provides the missing “verification layer” that transforms individual developer trust (AI4SE) into organizational confidence (SE4AI).

D. Why ISO quality characteristics matter

Across the organizations we studied, quality was interpreted through different aspects: efficiency by developers, explainability by managers, and compliance by legal teams. This fragmentation created challenges in aligning expectations, coordinating evaluations, and making trade-offs explicit. What was missing was a common foundation to reason about quality without flattening its complexity.

In our study context, the companies did not adopt ISO/IEC 25059 as a formal compliance standard. They still performed quality work through internal policies and routines, such as security and privacy reviews, supplier and licensing evaluations, QA testing, and operational monitoring. However, these activities were not recorded as traceable evidence. We therefore use ISO/IEC 25059 as a shared quality vocabulary to name and group observed quality concerns and to highlight which quality-in-use aspects are seldom evaluated in practice.

In the framework verification case, ISO characteristics helped participants translate vague concerns into actionable targets. For instance, instead of “the model should be fast

and safe,” teams articulated expectations as “inference time < 45 seconds (efficiency)” and “no confidential leakage from enterprise information (security).” This shift enabled traceable decisions and role-specific accountability throughout the life-cycle.

Moreover, ISO’s separation of characteristics and sub-characteristics (e.g., from reliability to robustness) allowed practitioners to tailor depth without discarding structure. This flexibility appears important in real-world development, where not all quality attributes are equally relevant. The standard helps teams organize what they have already done, while exposing what might be missed in terms of software quality.

We acknowledge that ISO 25059 is still under refinement and does not address all GenAI-specific characteristics. However, our evidence suggests that it provides a shared model grounded in both academic rigor and industrial recognition.

a) *Relation to the EU AI Act:* While ISO/IEC quality standards help structure technical quality, they do not by themselves address regulatory duties under the EU AI Act (e.g., risk classification, documentation, conformity assessment, post-market monitoring). When this study started, the operational guidance for the Act was still evolving in the studied companies, and the usage of software development was considered unclear. Therefore, teams mapped existing ISO-based quality characteristics to internal policies for GenAI adoption and usage, conducted legal and privacy reviews, and prepared for later alignment with evolving standards and regulatory requirements, not limited to ISO/IEC 25059.

Future studies should investigate how GenAI quality evolves over time and how teams update evaluation criteria accordingly. Moreover, the GenAI Quality Lead role needs further exploration, including required competencies and interactions with existing quality assurance activities.

VIII. CONCLUSIONS

The use of GenAI technologies in industrial software products introduces new challenges in software quality assurance. Through a case study in two industrial organizations, we investigated the process of GenAI adoption in practice, what quality aspects matter, who is responsible for evaluation, and how these activities are carried out across the lifecycle of GenAI-enabled software.

In the study, we contribute a framework: a process-oriented and role-aware view of GenAI quality evaluation. The framework is built based on observed practices and practitioners’ feedback. This framework clarifies when and how quality should be assessed between legal, technical, and operational responsibilities.

By visualizing where evaluation takes place and what characteristics are considered, we help practitioners gain a deeper understanding of GenAI-related quality. The introduction of the GenAI Quality Lead role addresses a missing coordination function, supporting traceability, consistency, and compliance in multi-role environments. The real-world implementation further confirms the framework’s applicability and exposes opportunities for refinement.

Looking forward, GenAI adoption can accelerate across sectors and development contexts. It raises the necessity for

quality evaluation frameworks that are not only theoretically grounded but also implementable in practice. Our study provides a framework for practitioners to follow and offers a lens for rethinking software quality in the era of GenAI.

ACKNOWLEDGEMENT

We acknowledge support from the KKS Foundation through the S.E.R.T. Research Profile Project at Blekinge Institute of Technology.

DATA AVAILABILITY

The datasets supporting the conclusions of this article are available in the IEEEDataPoint [28].

REFERENCES

- [1] S. Feuerriegel, J. Hartmann, C. Janiesch, and P. Zschech, “Generative ai,” *Business & Information Systems Engineering*, vol. 66, no. 1, pp. 111–126, 2024.
- [2] Z. Zheng, K. Ning, Q. Zhong, J. Chen, W. Chen, L. Guo, W. Wang, and Y. Wang, “Towards an understanding of large language models in software engineering tasks,” *Empirical Software Engineering*, vol. 30, no. 2, p. 50, 2025.
- [3] J. Zhang, M. Harman, L. Ma, and Y. Liu, “Machine learning testing: Survey, landscapes and horizons,” *IEEE Transactions on Software Engineering*, vol. 46, no. 10, pp. 1070–1093, 2020.
- [4] L. Yu, “Paradigm shift on coding productivity using genai,” *arXiv preprint arXiv:2504.18404*, 2025.
- [5] P. d. O. Santos, A. C. Figueiredo, P. Nuno Moura, B. Diirr, A. C. Alvim, and R. P. D. Santos, “Impacts of the usage of generative artificial intelligence on software development process,” in *Proceedings of the 20th Brazilian Symposium on Information Systems*, 2024, pp. 1–9.
- [6] A. Aleti, “Software testing of generative ai systems: Challenges and opportunities,” in *2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE)*. IEEE, 2023, pp. 4–14.
- [7] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, “Hidden technical debt in machine learning systems,” *Advances in Neural Information Processing Systems*, vol. 28, pp. 2503–2511, 2015.
- [8] C. T. Ungureanu and A. E. Amironesi, “Legal issues concerning generative ai technologies,” *Eastern Journal of European Studies*, vol. 45, 2023.
- [9] V. Riccio, G. Jahangirova, A. Stocco, N. Humbatova, M. Weiss, and P. Tonella, “Testing machine learning based systems: A systematic mapping,” in *Proceedings of the 14th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, ser. ESEM ’20. ACM, 2020, pp. 1–12.
- [10] J. Park and S. Choo, “Generative ai prompt engineering for educators: Practical strategies,” *Journal of Special Education Technology*, p. 01626434241298954, 2024.
- [11] A. Donvir, S. Panyam, G. Paliwal, and P. Gujar, “The role of generative ai tools in application development: A comprehensive review of current technologies and practices,” in *2024 International Conference on Engineering Management of Communication and Technology (EMCTECH)*. IEEE, 2024, pp. 1–9.
- [12] P. Runeson and M. Höst, “Guidelines for conducting and reporting case study research in software engineering,” *Empirical software engineering*, vol. 14, no. 2, p. 131, 2009.
- [13] J. Wang and Y. Chen, “A review on code generation with llms: Application and evaluation,” in *2023 IEEE International Conference on Medical Artificial Intelligence (MedAI)*. IEEE, 2023, pp. 284–289.
- [14] M. Tufano, C. Watson, G. Bavota, M. Di Penta, M. White, and D. Poshyvanyk, “Using deep learning to generate complete log statements,” in *Proceedings of the 44th International Conference on Software Engineering*, ser. ICSE ’22. ACM, 2022, pp. 883–895.
- [15] J. Li, T. Tang, W. X. Zhao, J.-Y. Nie, and J.-R. Wen, “Pre-trained language models for text generation: A survey,” *ACM Computing Surveys*, vol. 56, no. 9, pp. 1–39, 2024.

- [16] M. Simaremare and H. Edison, "The state of generative ai adoption from software practitioners' perspective: An empirical study," in *2024 50th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 2024, pp. 106–113.
- [17] International Organization for Standardization, "Iso/iec 25059:2023 software engineering — systems and software quality requirements and evaluation (square) — quality model for ai systems," *International Organization for Standardization*, 2023.
- [18] E. Breck, S. Cai, E. Nielsen, M. Salib, and D. Sculley, "The ml test score: A rubric for ml production readiness and technical debt reduction," *IEEE Big Data*, pp. 1123–1132, 2017.
- [19] M. T. Ribeiro, T. Wu, C. Guestrin, and S. Singh, "Beyond accuracy: Behavioral testing of nlp models with checklist," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 4902–4912.
- [20] S. Barke, M. B. James, and N. Polikarpova, "Grounded copilot: How programmers interact with code-generating models," in *Proceedings of the 44th ACM/IEEE International Conference on Software Engineering*, ser. ICSE '22. IEEE, 2022, pp. 319–331.
- [21] U.-e. Habiba, M. Haug, J. Bogner, and S. Wagner, "How mature is requirements engineering for ai-based systems? a systematic mapping study on practices, challenges, and future research directions," *Requirements Engineering*, vol. 29, no. 4, pp. 567–600, 2024.
- [22] R. Khojah, M. Mohamad, P. Leitner, and F. G. de Oliveira Neto, "Beyond code generation: An observational study of chatgpt usage in software engineering practice," *Proc. ACM Softw. Eng.*, vol. 1, no. FSE, Jul. 2024. [Online]. Available: <https://doi.org/10.1145/3660788>
- [23] D. Russo, "Navigating the complexity of generative ai adoption in software engineering," *ACM Trans. Softw. Eng. Methodol.*, vol. 33, no. 5, Jun. 2024. [Online]. Available: <https://doi.org/10.1145/3652154>
- [24] S. Lambiase, G. Catolino, F. Palomba, F. Ferrucci, and D. Russo, "Investigating the role of cultural values in adopting large language models for software engineering," *ACM Trans. Softw. Eng. Methodol.*, vol. 35, no. 1, Dec. 2025. [Online]. Available: <https://doi.org/10.1145/3725529>
- [25] R. Choudhuri, B. Trinkenreich, R. Pandita, E. Kalliamvakou, I. Steinmacher, M. Gerosa, C. A. Sanchez, and A. Sarma, "What needs attention? prioritizing drivers of developers' trust and adoption of generative ai," *arXiv preprint arXiv:2505.17418*, 2025.
- [26] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, A. Wesslén *et al.*, *Experimentation in software engineering*. Springer, 2012, vol. 236.
- [27] K. Petersen and C. Wohlin, "Context in industrial software engineering research," in *2009 3rd International Symposium on Empirical Software Engineering and Measurement*. IEEE, 2009, pp. 401–404.
- [28] L. Yu, "A process model for evaluating genai adoption and use in software development," 2025. [Online]. Available: <https://dx.doi.org/10.21227/zhbk-fc92>
- [29] D. S. Cruzes and T. Dyba, "Recommended steps for thematic synthesis in software engineering," in *2011 International Symposium on Empirical Software Engineering and Measurement*. IEEE, 2011, pp. 275–284.
- [30] K. Kenthapadi, H. Lakkaraju, and N. Rajani, "Generative ai meets responsible ai: Practical challenges and opportunities," in *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*, 2023, pp. 5805–5806.
- [31] Prometheus, "Prometheus: Monitoring for your systems and services," <https://prometheus.io>, 2025, accessed 2025-08-01.
- [32] P. Runeson, M. Höst, A. Rainer, and B. Regnell, *Case Study Research in Software Engineering: Guidelines and Examples*. Hoboken, NJ: Wiley, 2012.