

DOI: https://doi.org/10.48009/2_iis_2026_149

Rethinking assessment in the age of generative AI: Process-evidence assessment for information systems courses

Pratibha Menon, *Pennsylvania Western University*, menon@pennwest.edu

Jayakumar Annadatha, *Pennsylvania Western University*, jannadatha@pennwest.edu

Abstract

Generative artificial intelligence (GenAI) has weakened a long-standing assumption in Information Systems (IS) assessment: that a competent submitted artifact reliably demonstrates student understanding. When students can use large language models to generate code, data models, requirements analyses, and written explanations, artifact quality alone becomes insufficient evidence of learning. This paper introduces Process-Evidence Assessment (PEA), a pedagogical framework that reorients IS assessment toward observable evidence of reasoning, verification, iteration, and explanation. PEA integrates assessment validity, dual-process theory, authentic assessment, evaluative judgment, self-regulated learning, and computing education into five components: AI-use stratification, evidence packs, process timelines, reasoning-weighted rubrics, and written mini-defense questions. Formative classroom observations from a partial PEA implementation across three undergraduate IS courses suggest that PEA made student reasoning more visible than artifact quality alone, especially when students verified AI outputs, documented rejected suggestions, explained tradeoffs, and answered submission-specific follow-up questions.

Keywords: Generative-AI, assessment, validity, process-based, information-systems

Introduction

Generative artificial intelligence has changed the evidentiary conditions under which Information Systems (IS) instructors assess student learning. In many IS courses, students are asked to submit products: code, entity-relationship diagrams (ERD), data flow diagrams (DFD), requirements documents, test plans, dashboards, security analyses, and project reports. Historically, a competent product gave instructors reasonable evidence that the student had engaged in the disciplinary reasoning required to produce it. That inference is no longer reliable when a student can use a large language model (LLM) to produce a plausible artifact with limited understanding of the underlying concepts.

The problem is not simply that students may use AI inappropriately. GenAI tools are becoming part of professional computing, analytics, and business workflows. Graduates will be expected to use AI-enabled tools productively while still exercising judgment, verification, and accountability. The central assessment problem is therefore a validity problem: Does the assessment still measure the construct it claims to measure? Dawson, Bearman, Dollinger, and Boud (2024) argue that validity matters more than cheating in the GenAI era because the most serious risk is that assessment may no longer provide defensible evidence of student capability. That framing is consequential for IS in a particular way because the discipline's core artifacts, including code, data models, diagrams, and analyses, are exactly the outputs that current LLMs

generate most fluently, so the gap between a competent-looking artifact and demonstrated reasoning is unusually wide in IS coursework.

Recent studies in Issues in Information Systems (IIS) and the broader computing-education literature reinforce this concern by showing that AI-supported programming requires more than tool access. Menon (2025) found shortfalls in students' AI-supported programming practices, including planning before prompting, verifying AI-generated output, and integrating AI assistance with personally authored code. Similar findings across computing education (Denny et al., 2024; Prather et al., 2023) indicate that fluent AI output can outpace students' ability to evaluate it.

Institutional responses to GenAI have commonly centered on detection or prohibition. Detection tools appeal because they preserve existing assessment forms, but they do not measure IS competence and raise serious fairness concerns: Liang, Yuksekgonul, Mao, Wu, and Zou (2023) found that GPT detectors can misclassify non-native English writing as AI-generated. Detection is also technically fraught, with accuracy that degrades as models evolve and a high burden of proving misuse to an academic-integrity standard (Weber-Wulff et al., 2023). Prohibition may suit narrow foundational tasks but fails as a general strategy, since students will use AI tools after graduation. Both responses police AI use rather than redesigning assessment to measure reasoning.

This paper proposes Process-Evidence Assessment (PEA) as a validity-centered alternative. PEA does not treat AI use as automatically illegitimate. Instead, it asks whether the student can demonstrate the reasoning that disciplinary competence requires. The core premise is straightforward: if AI can generate the artifact, then the artifact alone cannot demonstrate learning. What must be assessed is the student's ability to explain choices, identify errors, test claims, verify outputs, adapt reasoning under changed conditions, and use AI assistance without surrendering judgment.

PEA is not a claim that reflection, portfolios, oral defenses, or process documentation are new. Its contribution is to synthesize these established practices into an IS-specific assessment framework for courses in which students have routine access to GenAI tools. Although process-oriented and authentic assessment practices are well established, the literature offers IS instructors little concrete guidance on how to assemble them into a coherent, scalable assessment design. This paper asks: how can IS educators design assessments that remain valid evidence of disciplinary reasoning when students have routine access to generative AI? In response, the paper contributes the PEA framework and four design principles abstracted from it for broader reuse. This is a pedagogical framework and formative teaching-practice paper, not a controlled empirical study. The remainder of the paper reviews the relevant literature and theory, describes the methodology of a Spring 2026 formative implementation, presents the PEA framework and its five components, reports observations, derives design principles, and discusses implications and limitations.

Literature Review and Theoretical Framing

Generative AI, Computing Education, and the Validity Challenge

GenAI has particular significance for computing and IS education because code-oriented LLMs can generate syntactically plausible source code and explanations from natural-language prompts. Denny et al. (2024) describe GenAI as a major challenge and opportunity for computing education because such tools affect how students learn programming, how instructors design tasks, and how assessment evidence should be interpreted. Prather et al. (2023) show that novice programmers can interact with code-generation tools in ways that are both productive and cognitively challenging; students may receive outputs that appear to satisfy their intentions while still lacking sufficient understanding to evaluate or modify them.

The empirical picture is not simply negative. Kazemitabaar et al. (2023) found that AI code generators can support novice programming performance under certain conditions, which suggests that the educational issue is not whether AI should be banned but how students should be taught and assessed when AI is available. If AI assistance is integrated without requiring verification and explanation, students may outsource the very reasoning that assignments are designed to cultivate. If AI assistance is scaffolded, it can become an object of evaluation and critique.

Finnie-Ansley et al. (2022) showed that OpenAI Codex could solve a substantial share of introductory programming exam problems, demonstrating that AI can produce competent-looking IS artifacts without student reasoning. IS-specific evidence points to the same conclusion. Menon (2023) examined GitHub Copilot assistance for object-oriented programming assignments and found that AI-generated suggestions could support code completion and exploration but did not remove the need for student judgment, verification, and explanation. Menon (2025) surveyed undergraduates in Computer Science and Computer Information Systems courses and found substantial shortfalls in recommended AI-supported programming practices, including planning, output verification, and integration of AI suggestions with personal code. Consistent with broader computing-education findings (Becker et al., 2023), these studies suggest that the assessment risk is not AI use itself, but unexamined AI use that bypasses planning, monitoring, and verification.

Dual-Process Theory and AI-Amplified Cognition

Dual-process theories distinguish rapid, automatic, heuristic processing from slower, controlled, analytically demanding reasoning (Evans & Stanovich, 2013; Kahneman, 2011). In this paper, System 2 and Type 2 are used interchangeably to denote the slow, deliberate, analytical mode of reasoning, in contrast to the fast, automatic processing of System 1 (Type 1). Kahneman's System 1/System 2 terminology is useful for assessment design because it highlights the difference between accepting a plausible answer and deliberately checking whether that answer is correct. Evans and Stanovich (2013) similarly emphasize that Type 2 processing supports hypothetical thinking and draws on working memory, which aligns with the kinds of reasoning IS assignments are intended to develop.

In IS coursework, these slower reasoning behaviors include tracing program logic, debugging, checking edge cases, validating data-flow balance, comparing design alternatives, and explaining tradeoffs. GenAI tools are not human cognitive systems, but their fluent outputs can encourage fast acceptance of plausible code, diagrams, or explanations before students engage in deliberate verification. PEA therefore treats verification, error detection, design justification, and adaptive explanation as assessable evidence of the disciplined reasoning that product artifacts can no longer demonstrate by themselves.

Assessment Validity and the Artifact Inference

Validity theory provides the foundation for PEA. Messick (1995) defines validity as an integrated evaluative judgment about the adequacy and appropriateness of interpretations and actions based on assessment evidence. Kane's (2006) argument-based approach similarly emphasizes that assessment claims require connecting observed performance to intended interpretations. In an IS course, the intended interpretation is rarely that a student can submit a polished artifact. The intended interpretation is that the student can reason through a problem, make justified design decisions, verify correctness, and communicate technical logic.

GenAI weakens the warrant connecting product quality to student understanding. A polished entity-relationship diagram may be evidence of modeling competence, but it may also be evidence that a student accepted an AI-generated model. A working PHP form may indicate understanding of validation and

session handling, but it may also indicate successful prompt-and-paste behavior. Product-only assessment therefore risks construct underrepresentation: it samples the visible output while missing critical reasoning processes. PEA responds by expanding the evidence base so that interpretations of learning rest on multiple indicators: artifact quality, process evidence, verification reasoning, and student explanation.

Authentic Assessment and Evaluative Judgment

Authentic assessment literature also supports the need to evaluate judgment in context. Ajjawi et al. (2024) argue that authenticity in assessment should be understood broadly, not merely as replication of specific workplace tasks. For IS education, authenticity means asking students to engage in the kinds of evaluative and corrective practices that professionals use when working with imperfect information, changing requirements, and tool-generated outputs. In a GenAI environment, authenticity includes the ability to interrogate AI suggestions rather than accepting them at face value.

Evaluative judgment is particularly relevant in a GenAI environment. Bearman, Tai, Dawson, Boud, and Ajjawi (2024) argue that students need to develop the capability to judge the quality of AI outputs and AI-mediated processes. PEA operationalizes that idea in IS assignments by requiring students to document what AI produced, what they accepted or rejected, how they verified the output, and what limitations remained. The aim is not merely to catch misuse. The aim is to make judgment itself a target of instruction and assessment.

AI-Use Transparency and Assessment Policy Stratification

Clear AI-use expectations are essential because ambiguity creates inconsistent student behavior and adversarial classroom dynamics. The Artificial Intelligence Assessment Scale (AIAS) provides one useful model. Perkins, Furze, Roe, and MacVaugh (2024) propose a tiered framework for specifying appropriate levels of GenAI use based on learning outcomes, and Furze, Perkins, Roe, and MacVaugh (2024) report pilot evidence that such frameworks can support more transparent and constructive assessment practice. PEA draws from the same principle but adapts it to IS tasks by connecting AI-use permissions directly to evidence requirements. When AI is permitted or expected, students must provide verification evidence. When AI is prohibited, the reason should be linked to foundational skill development. This approach is consistent with broader GenAI assessment reform guidance that calls for transparent task design, explicit expectations, and assessment practices that remain meaningful when AI tools are available (Lodge et al., 2023).

Self-Regulated Learning and Observable Reasoning Traces

Self-regulated learning (SRL) provides a cognitive basis for PEA's evidence pack. Zimmerman (2000) describes learners as cycling through forethought, performance monitoring, and self-reflection. Winne and Hadwin (1998) similarly conceptualize learning as a process in which learners monitor progress, compare outcomes against standards, and adapt strategies. Winne (2020) argues that trace data can support learning analytics when interpreted through valid construct models rather than treated as superficial behavioral proxies. Related work on open learner models also supports the value of making learning evidence visible to both students and instructors. Bull and Kay (2016) argue that learner-facing representations of learning data can support reflection, interpretation, and learner awareness when the evidence is meaningfully connected to the learning construct.

In IS assignments, reasoning traces may include test matrices, version histories, AI interaction logs, rejected design alternatives, annotated diagram revisions, and short verification memos. These traces are not valuable merely because they document activity. When combined together, validity theory, dual-process theory, authentic assessment, evaluative judgment, and self-regulated learning converge on a single implication for IS assessment: when artifacts no longer warrant inferences about reasoning, assessment

must collect direct evidence of the reasoning itself. No existing framework operationalizes this implication specifically for GenAI-enabled IS courses. PEA is designed to fill that gap.

Methodology

The five PEA components were derived deductively from the theoretical foundations reviewed in the prior section of this paper. Each component operationalizes one or more of the constructs (validity warrants, System 2 reasoning, evaluative judgment, and self-regulated monitoring) that the literature identifies as necessary for demonstrating competence but no longer observable in AI-generated artifacts. This paper uses a design-based, formative evaluation approach. The first phase involved developing the PEA framework from assessment validity theory, dual-process theory, GenAI assessment reform literature, authentic assessment, evaluative judgment, self-regulated learning, and computing education research. The second phase involved a partial classroom implementation across three undergraduate IS courses at a regional comprehensive university during Spring 2026.

The courses were a web programming course using PHP-based dynamic applications, an introductory Java programming course focused on object-oriented design and encapsulation, and a systems analysis and design course covering data flow diagrams and entity-relationship diagrams. The programming courses enrolled fewer than 30 students per section, and the systems analysis and design section enrolled approximately 10 students. Two instructors participated. The primary designer of PEA taught the web programming and Java courses. A second instructor independently adopted PEA-informed evidence packs and written follow-up questions in the systems analysis and design course. Table 1 summarizes how PEA was adapted in each course, the evidence required, and the focus of the written mini-defense.

Table 1. Spring 2026 Partial PEA Implementation

Course	PEA adaptation	Evidence required	Mini-defense focus
Web programming	AI-integrated with verification for applied PHP assignments.	Requirements summary, commented source code, AI interaction log, formal test matrix, verification memo.	Input validation, session behavior, edge-case handling, and security-aware testing.
Introductory Java	AI-integrated with verification for selected object-oriented assignments.	AI log, source code, test cases, break-it experiment, explanation of encapsulation failures.	Object state, constructor logic, access modifiers, and consequences of removed safeguards.
Systems analysis and design	AI-integrated with verification for DFD and ERD assignments.	Thinking Log, annotated drafts, assumptions, balancing checks, error identification, revision rationale.	Data-flow logic, process decomposition, entity/relationship decisions, and modeling errors.

Formative observations were drawn from four sources available to the participating instructors by using the completed evidence packs, reasoning-weighted rubric scores, written mini-defense responses, and the instructors' own teaching notes recorded during grading. These sources were reviewed qualitatively after the semester to identify recurring patterns in how evidence-pack depth related to mini-defense quality and to grade interpretation. The review was interpretive rather than systematic, as the observations were not formally coded, no inter-rater reliability was computed, and no student-level data were statistically analyzed. The patterns reported below should therefore be read as instructor-observed tendencies that motivate future controlled study, not as measured effects.

The Spring 2026 cycle should be understood as formative and naturalistic. It did not use random assignment, a control group, formal pre/post measures, or inter-rater reliability testing. The purpose was to observe implementation patterns and refine the framework, not to establish causal learning effects. The current paper therefore makes limited claims: PEA appeared to make student reasoning more visible and grade interpretation more defensible, but controlled and multi-institutional research is needed to evaluate learning outcomes, reliability, and scalability. This work reports the participating instructors' formative observations of their own teaching practice and does not analyze human-subjects data; it was therefore treated as classroom-practice reflection rather than human-subjects research. The planned controlled study described in the Future Research section has received IRB approval prior to data collection.

Process-Evidence Assessment Framework

Design Logic

PEA is a framework for redesigning IS assignments so that student reasoning becomes visible alongside the submitted artifact. It rests on three design assumptions: GenAI use is inevitable in many applied IS contexts; valid assessment requires evidence of reasoning that cannot be inferred reliably from artifact quality alone; and such evidence should be collected through ordinary academic artifacts rather than intrusive surveillance. PEA includes five components. In Spring 2026, four were implemented: AI-use policy stratification, evidence packs, reasoning-weighted rubrics, and written mini-defenses. The process timeline component was designed but deferred because the first cycle prioritized evidence-pack feasibility and written mini-defense design before adding an additional sequencing artifact. Table 2 lists the components and implementation status.

Table 2. PEA Components and Implementation Status

Component	Purpose	Typical Evidence	Status in Spring 2026 implementation
AI-use policy stratification	Clarify whether AI is prohibited, permitted with disclosure, or integrated with verification.	Assignment policy statement; student AI-use declaration.	Implemented.
Evidence pack	Make reasoning, verification, and revision visible.	AI interaction log; verification memo; test matrix; annotated drafts; rejected outputs.	Implemented, adapted by course.
Process timeline	Reconstruct the sequence of work and major decisions.	Milestone log; commit history; LMS timestamps; version history.	Designed but not formally implemented.
Reasoning-weighted rubric	Grade disciplinary reasoning independently from product polish.	Rubric dimensions for verification, error detection, justification, and adaptation.	Implemented.
Written mini-defense	Check whether students can explain, extend, or troubleshoot their own submission.	Submission-specific follow-up questions answered after submission.	Implemented asynchronously.

Component 1: AI-Use Policy Stratification

PEA begins with explicit assignment-level AI-use policies. Rather than using a single coursewide rule, instructors classify tasks according to learning purpose. A foundational task may be AI-prohibited because

students need to practice a basic mental model without tool support. A scaffolded task may be AI-assisted with disclosure, allowing students to use AI for brainstorming or debugging while requiring documentation. An applied project may be AI-integrated with verification, where effective AI use and critical evaluation are part of the assessment itself.

The policy must state not only what is allowed but also what evidence is required. For example, an AI-integrated database modeling assignment may permit students to ask an LLM for candidate entities and relationships, but students must submit a log showing which suggestions were rejected, how cardinalities were verified against requirements, and what normalization decisions were made independently. The policy therefore shifts the classroom norm from concealment to accountable use.

Component 2: Evidence Pack

The evidence pack is the central assessment mechanism in PEA. It is a structured companion submission that documents the reasoning behind the artifact. The exact format varies by course, but the evidence pack should include four types of evidence: (a) problem interpretation, (b) AI interaction and revision evidence, (c) verification evidence, and (d) reflective judgment about limitations.

In programming courses, evidence may include a test matrix with valid inputs, empty fields, type mismatches, boundary values, injection attempts, expected outcomes, actual outcomes, and pass/fail notes. It may also include a short break-it report in which students deliberately remove validation or encapsulation safeguards and explain the resulting failure. In systems analysis and design, evidence may include a Thinking Log describing assumptions, data-flow balancing, decomposition decisions, entity revisions, and error checks such as black holes, miracles, missing data stores, or unresolved cardinalities.

The evidence pack should be short enough to grade but specific enough to be diagnostic. A 300- to 500-word verification memo can work well when paired with structured evidence such as a table or annotated draft. The memo could ask students to identify at least one AI-generated error or limitation, describe how they verified correctness, and explain one significant design decision.

Component 3: Process Timeline

The process timeline records the sequence of work. It may include dates and times for initial interpretation, AI consultation, major revision, verification, and final preparation. The purpose is not to monitor students continuously. The purpose is to make staged work and revision visible when an assignment is intended to develop iterative reasoning.

This component was not formally implemented in the Spring 2026 cycle, so it should be treated as a planned refinement. In future implementations, the timeline could be collected through low-burden methods such as a short milestone log, Git commit history, LMS draft history, or versioned document comments. The timeline should not become a surveillance instrument; it should be used only when it contributes to the construct being assessed.

Component 4: Reasoning-Weighted Rubric

PEA requires rubrics that score reasoning directly. A product-only rubric may award most points for whether the code runs or the diagram looks correct. A PEA rubric still values artifact quality, but it also allocates substantial weight to verification, explanation, error detection, and adaptation. For example, a systems analysis assignment might allocate 60% to technical diagram quality and 40% to the Thinking Log and verification evidence. A web programming assignment might allocate points to functional correctness, security-aware testing, AI-output evaluation, and explanation of edge-case handling.

Table 3. Sample PEA Rubric Dimensions

Criterion	Strong evidence	Partial evidence	Weak evidence
AI-output evaluation	Identifies specific AI errors, assumptions, or omissions and explains how they were corrected or rejected.	Notes some AI limitations but provides limited analysis of how they affected the final work.	Describes AI use without evaluating output quality or explaining acceptance decisions.
Verification and testing	Uses tests, edge cases, balancing checks, validation steps, or traceability evidence tied to requirements.	Provides basic verification but gives limited rationale for why the checks matter.	Provides little evidence that the artifact was checked against requirements or failure conditions.
Design reasoning	Justifies choices, tradeoffs, constraints, alternatives, and limitations using course concepts.	Explains choices generally but provides limited connection to requirements or concepts.	Lists what was done without explaining why the design is appropriate.
Iteration and revision	Shows meaningful movement from draft to final submission with rationale for revisions.	Shows some revision but limited explanation of what changed or why.	Shows little evidence of iteration, revision, or response to feedback or errors.
Mini-defense readiness	Explains, adapts, or troubleshoots the submitted work when asked submission-specific questions.	Explains familiar parts of the work but struggles when the scenario changes.	Cannot explain key details of the submitted work or relies on generic explanations.

The key shift is that a polished artifact with thin reasoning evidence should not receive the same score as a polished artifact accompanied by substantive verification. Similarly, a technically imperfect artifact may still demonstrate meaningful learning if the evidence pack shows accurate diagnosis, informed revision, and clear understanding of remaining limitations.

Because PEA is intended to be adapted across IS domains, the rubric should define common reasoning behaviors while allowing course-specific evidence formats. Table 3 provides an illustrative rubric structure that can be adapted for programming, database, systems analysis, analytics, or cybersecurity assignments. In programming courses, these criteria may be operationalized through test matrices, break-it reports, AI interaction logs, and code explanations. In systems analysis and design, the same criteria may be operationalized through DFD balancing checks, ERD revision rationales, illegal-flow identification, and requirements-to-model traceability.

Component 5: Written Mini-Defense

The mini-defense asks students to respond to a small number of questions generated after the instructor has reviewed the submission. Questions are specific to the student's work. In programming, a question might ask the student to trace what happens when a particular invalid input is submitted, or to explain how a session variable changes across pages. In systems analysis, a question might ask why a data flow crosses a process boundary, why a relationship is one-to-many rather than many-to-many, or how a missing requirement would alter the model. PEA uses written mini-defenses rather than requiring synchronous oral defenses in every case. Written defenses reduce scheduling burden and are more scalable for regular undergraduate sections. They also create a record that can be graded with a rubric. Oral defenses may still be appropriate for capstones, graduate seminars, or high-stakes integrity concerns, but they are not necessary for every PEA implementation. Table A in the Appendix section provides example written mini-defense questions across common IS course types, illustrating how submission-specific questioning adapts to different domains.

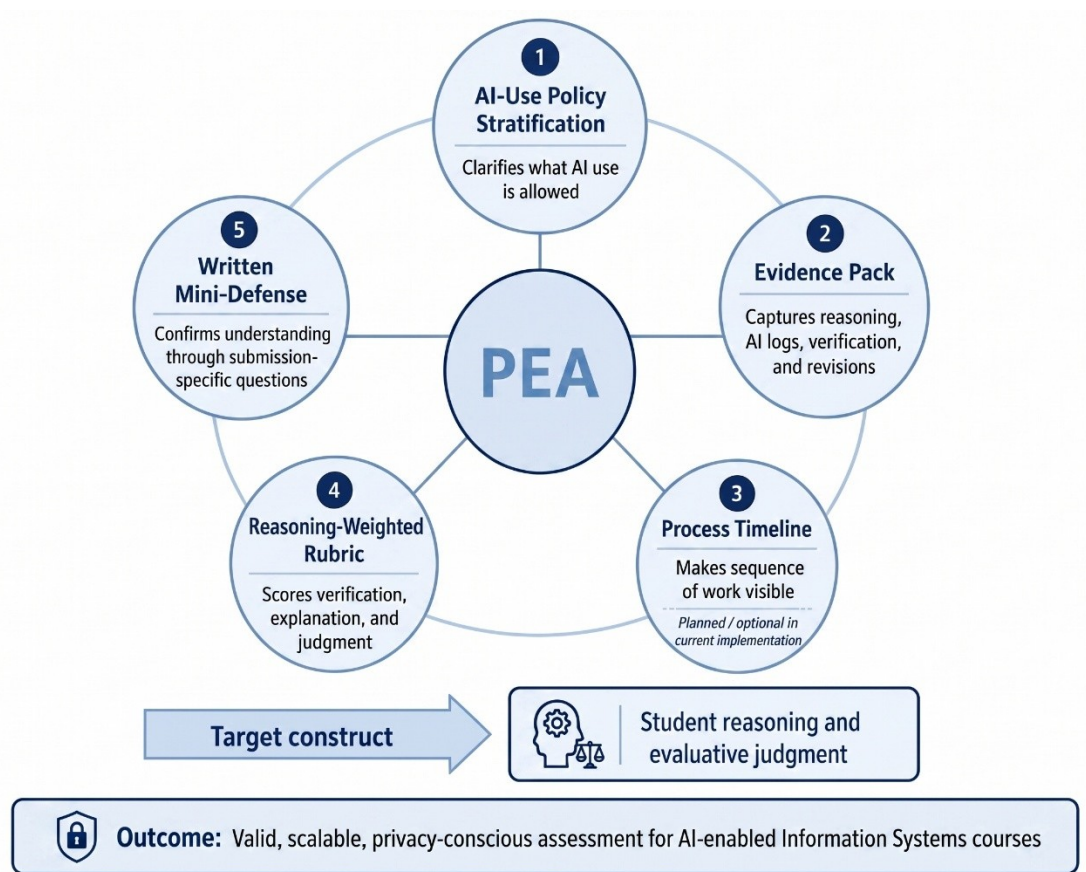


Figure 1. The Process-Evidence Assessment (PEA) Framework

Figure 1 shows how the five components operate as a workflow. Entry to the framework begins when the instructor sets the assignment-level AI-use policy, which establishes what AI use is permitted and what evidence is therefore required. As students work, they assemble the evidence pack and, in future cycles, the process timeline. At submission, the reasoning-weighted rubric is applied jointly to the artifact and the evidence pack. The instructor then issues submission-specific written mini-defense questions. Exit from the framework is the final grade, which reflects the convergence of artifact quality, evidence-pack reasoning, and mini-defense responses rather than artifact quality alone.

Formative Observations

Observation 1: Evidence Depth and Defense Quality Appeared Related

Across the three courses, students who submitted substantive evidence packs tended to provide more precise written mini-defense responses. In web programming, students who completed detailed test matrices could explain why specific boundary values, empty-field cases, type mismatches, or injection attempts were included. Their follow-up responses were more likely to reference concrete code paths and expected behavior rather than generic statements. In Java, students who completed break-it experiments could explain how removing encapsulation guards altered object state or produced invalid behavior.

A similar pattern appeared in systems analysis and design. Students whose Thinking Logs documented assumptions, balancing checks, and revision decisions were better able to answer targeted questions about

diagram logic. When asked why a process had a particular input and output, or why an entity relationship had a specific cardinality, these students tended to provide explanations linked to requirements rather than surface features of the diagram.

Observation 2: Thin Evidence Made Comprehension Gaps More Visible

Students who submitted thin or descriptive evidence packs often struggled when follow-up questions required adaptation. In programming assignments, some students could describe what their code did in broad terms but could not trace how logic changed when a scenario was modified. In systems analysis and design, some students submitted visually complete diagrams but could not identify a missing data flow, explain a balancing issue, or justify a modeling choice when asked. This pattern is important because the artifact alone would not have exposed the gap. A product-only rubric might have awarded high marks to a plausible diagram or functioning program. The PEA evidence pack and mini-defense made it possible to distinguish artifact polish from disciplinary understanding.

Observation 3: Reasoning-Weighted Rubrics Changed Grade Interpretation

The use of reasoning-weighted rubrics changed how grades were interpreted. In systems analysis and design, a rubric that weighted the technical diagram and Thinking Log separately allowed the instructor to recognize students who showed genuine iteration even when their final diagrams had minor technical weaknesses. Conversely, students with polished diagrams but generic Thinking Logs received lower scores than a product-only rubric would likely have produced.

In programming courses, the same principle applied to testing and verification. Functionally similar code submissions were not treated as equivalent when one was accompanied by careful test evidence and the other by minimal explanation. This distinction aligns the grade more closely with the intended construct: not only the ability to produce a working product, but the ability to verify and explain it.

Implementation Challenges

Two implementation challenges were especially visible. First, students needed help distinguishing verification from description. Many students initially wrote summaries of what they built rather than analyses of how they knew it was correct. Annotated exemplars helped. Showing one strong verification memo and one weak descriptive memo clarified expectations more efficiently than repeated verbal instructions.

Second, grading took more time than product-only assessment. In the systems analysis and design course, reading the Thinking Log alongside the diagram roughly doubled grading time for some assignments. In programming courses, reviewing test matrices and break-it reports added time beyond checking whether the code ran. However, the added time produced more diagnostic grading evidence and reduced uncertainty about what students understood. The practical lesson is that PEA should be introduced selectively, beginning with major assignments where validity risk is high and reasoning evidence is most valuable.

Discussion

What PEA Adds to Existing Assessment Practice

PEA should be understood as a synthesis rather than a wholly new assessment genre. It draws from authentic assessment, portfolio logic, reflective writing, oral defense, and process documentation. Its contribution is to integrate these practices into a coherent framework for IS courses in which GenAI can produce the same artifacts that instructors traditionally graded. The distinctive feature is the triangulation of artifact, evidence pack, rubric, and mini-defense around AI-mediated reasoning.

This triangulation matters because each component alone is limited. An AI log can be fabricated or superficial. A verification memo can become performative. A polished artifact can mask weak understanding. A mini-defense can be affected by writing ability or anxiety. Used together, however, these components make it more difficult for students to submit AI-generated work without engaging in the reasoning the course intends to develop. More importantly, they teach students that responsible AI use requires verification, explanation, and accountability.

Practical Feasibility

The claims in this paper are bounded by the formative implementation. The observations come from three undergraduate IS courses at one institution, with small to moderate enrollments. The primary designer of PEA taught two courses, which introduces possible confirmation bias. The implementation did not include a control group, random assignment, pre/post testing, inter-rater reliability, or systematic qualitative coding. The findings should therefore be interpreted as formative evidence that PEA can make reasoning differences more visible, not as evidence that PEA improves learning outcomes. The process timeline was also not formally implemented. Finally, evidence packs, AI logs, verification memos, and even recorded defenses can themselves be AI-generated. PEA mitigates this risk through triangulation and submission-specific questioning, but it does not eliminate dishonesty or guarantee authentic reasoning.

Design Principles for AI-Ready IS Assessment

The following four principles abstract the design logic of PEA into guidance that can be reused beyond the specific components described above. They were derived analytically from the framework's theoretical foundations and refined against the Spring 2026 implementation experience. They are offered as design guidance rather than empirically validated propositions; their validation is part of the future research agenda described below.

Principle 1: Assess reasoning, not artifact quality alone. Artifact quality remains important, but it is insufficient when AI can generate plausible outputs. IS assessments should require students to explain decisions, verify outputs, identify errors, and justify tradeoffs. The exact evidence should match the domain: a test matrix in web programming, a break-it report in object-oriented programming, a balancing log in DFD work, or an annotated normalization rationale in database design.

Principle 2: Make AI expectations explicit and tiered. Students need to know whether AI is prohibited, permitted, or expected. The rule should be linked to the learning objective. If AI is prohibited, the instructor should explain the foundational skill being protected. If AI is integrated, students should be required to document how they evaluated and revised AI outputs.

Principle 3: Triangulate evidence. No single evidence source is enough. A PEA assignment should combine the submitted artifact, process evidence, verification reasoning, and a brief defense. Triangulation improves interpretive confidence because it connects the final product to the reasoning process that produced it.

Principle 4: Design for realistic workload. Assessment reforms fail when they depend on heroic instructor effort. PEA should be implemented through reusable templates, short evidence requirements, written mini-defenses, calibrated rubrics, and selective use on major assignments. The goal is not to document everything students do; the goal is to collect enough targeted evidence to support valid interpretation of learning.

Implications for IS Educators and Programs

For individual instructors, PEA offers a practical way to permit AI use without abandoning assessment integrity. The framework gives students a clear message: AI assistance is acceptable when it is accompanied by evidence of judgment, verification, and understanding. This shifts the classroom conversation from suspicion to professional accountability.

For IS programs, PEA can support curriculum-level assessment reform. Program directors can audit existing assignments by asking four questions: Does the assessment require reasoning evidence? Does it specify AI-use expectations? Does it triangulate evidence beyond the final product? Is it scalable for the enrollment level? This audit can identify high-risk assessments where GenAI most threatens validity. It can also support accreditation conversations by connecting assessment evidence to analytical reasoning, communication, ethical technology use, and professional preparation.

For the workforce pipeline, PEA aligns classroom assessment with AI-augmented professional practice. Employers do not need graduates who merely accept AI outputs. They need graduates who can evaluate AI-generated code, find missing requirements, test edge cases, explain design decisions, and remain accountable for technology-mediated work. PEA treats those behaviors as assessable competencies rather than optional habits.

Limitations and Future Research

Limitations

The claims in this paper are bounded by the formative nature of the implementation. The observations come from three undergraduate IS courses at one institution, with small to moderate enrollments. The primary designer of PEA taught two of the three courses, which introduces possible confirmation bias. The implementation did not include a control group, random assignment, pre/post testing, inter-rater reliability analysis, or systematic qualitative coding. The timeline was deferred because the first cycle prioritized evidence-pack feasibility and written mini-defense design before adding an additional sequencing artifact. The findings therefore should not be interpreted as evidence that PEA improves learning outcomes. They should be interpreted as formative evidence that PEA can make reasoning differences more visible and can guide refinement of assessment design. This concern extends to the evidence pack itself, as verification memos, AI logs, and even recorded defenses can be AI-generated, and polished written reflection is no longer reliable evidence of authentic reasoning on its own. PEA mitigates this through triangulation and submission-specific questioning, but it does not eliminate it.

The paper also does not claim that PEA eliminates academic dishonesty. Students may still submit incomplete logs, fabricate process evidence, or overstate their understanding. PEA reduces reliance on a single artifact by requiring multiple sources of evidence, but it does not create certainty. Its value lies in improving the quality of assessment inference, not in guaranteeing perfect detection of misuse.

Future Research

Future research should test PEA more rigorously. A next step is to develop and validate evidence-pack rubric dimensions, including inter-rater reliability across instructors and teaching assistants. A second step is to examine whether PEA improves students' evaluative judgment of AI outputs over a semester. A third step is to compare PEA and product-only assessment in matched assignments to determine whether PEA produces better evidence of learning, better retention, or stronger transfer to later courses. A fourth step is multi-institutional replication across IS subdomains, institutional types, and enrollment levels.

Automation may also be explored cautiously. Natural language processing could help flag thin evidence packs, identify missing verification elements, or summarize AI-log patterns for instructor review. However, automated scoring should not be deployed until it has been validated against expert human judgment and evaluated for fairness. The goal of PEA is to improve validity and trust, not to replace one unvalidated automation with another.

Addressing AI-generated reflection may require drawing on disciplines such as writing studies and English composition, which have begun to confront the same problem of distinguishing authentic critical thinking from fluent AI-produced text. Designing mini-defense and evidence-pack formats that remain diagnostic when students can generate plausible written reasoning is a priority for the planned controlled study.

Conclusion

GenAI has weakened the assumption that finished artifacts reliably demonstrate student learning. In IS education, this matters because many traditional assignments assess products that AI can now generate or substantially complete. Detection and prohibition may address some misconduct concerns, but they do not solve the deeper validity problem. The assessment task must be redesigned so that the evidence collected matches the competence instructors intend to measure.

Process-Evidence Assessment offers one such redesign. By combining AI-use stratification, evidence packs, process timelines, reasoning-weighted rubrics, and written mini-defenses, PEA shifts attention from artifact appearance to disciplinary reasoning. The Spring 2026 formative implementation suggests that even a partial PEA model can make reasoning differences more visible, especially when students are required to verify AI outputs, document revision decisions, and respond to submission-specific questions. The framework is not yet empirically validated, and more rigorous research is needed. Nevertheless, PEA provides IS instructors with a practical, privacy-conscious, and workforce-relevant pathway for assessing learning in the age of generative AI.

Acknowledgement

The authors used generative AI tools to support drafting, organization, and editing of selected portions of the manuscript. The authors reviewed, revised, and take full responsibility for the final content.

References

- Ajjawi, R., Tai, J., Dollinger, M., Dawson, P., Boud, D., & Bearman, M. (2024). From authentic assessment to authenticity in assessment: Broadening perspectives. *Assessment & Evaluation in Higher Education*, 49(4), 499-510. <https://doi.org/10.1080/02602938.2023.2271193>
- Bearman, M., Tai, J., Dawson, P., Boud, D., & Ajjawi, R. (2024). Developing evaluative judgement for a time of generative artificial intelligence. *Assessment & Evaluation in Higher Education*, 49(6), 893-905. <https://doi.org/10.1080/02602938.2024.2335321>
- Becker, B. A., Denny, P., Finnie-Ansley, J., Luxton-Reilly, A., Prather, J., & Santos, E. A. (2023). Programming is hard — or at least it used to be: Educational opportunities and challenges of AI code generation. In *Proceedings of the 54th ACM Technical Symposium on Computer Science*

- Education (pp. 500-506). Association for Computing Machinery. <https://doi.org/10.1145/3545945.3569759>
- Bull, S., & Kay, J. (2016). SMILI: A framework for interfaces to learning data in open learner models, learning analytics and related fields. *International Journal of Artificial Intelligence in Education*, 26(1), 293-331. <https://doi.org/10.1007/s40593-015-0090-8>
- Dawson, P., Bearman, M., Dollinger, M., & Boud, D. (2024). Validity matters more than cheating. *Assessment & Evaluation in Higher Education*, 49(7), 1005-1016. <https://doi.org/10.1080/02602938.2024.2386662>
- Denny, P., Prather, J., Becker, B. A., Finnie-Ansley, J., Hellas, A., Leinonen, J., Luxton-Reilly, A., Reeves, B. N., Santos, E. A., & Sarsa, S. (2024). Computing education in the era of generative AI. *Communications of the ACM*, 67(2), 56-67. <https://doi.org/10.1145/3624720>
- Evans, J. S. B. T., & Stanovich, K. E. (2013). Dual-process theories of higher cognition: Advancing the debate. *Perspectives on Psychological Science*, 8(3), 223-241. <https://doi.org/10.1177/1745691612460685>
- Finnie-Ansley, J., Denny, P., Becker, B. A., Luxton-Reilly, A., & Prather, J. (2022). The robots are coming: Exploring the implications of OpenAI Codex on introductory programming. In *Proceedings of the 24th Australasian Computing Education Conference* (pp. 10-19).
- Furze, L., Perkins, M., Roe, J., & MacVaugh, J. (2024). The AI Assessment Scale (AIAS) in action: A pilot implementation of GenAI-supported assessment. *Australasian Journal of Educational Technology*, 40(4), 16-31. <https://doi.org/10.14742/ajet.9434>
- Kahneman, D. (2011). *Thinking, fast and slow*. Farrar, Straus and Giroux.
- Kane, M. T. (2006). Validation. In R. L. Brennan (Ed.), *Educational measurement* (4th ed., pp. 17-64). American Council on Education/Praeger.
- Kazemitabaar, M., Chow, J., Ma, C. K. T., Ericson, B. J., Weintrop, D., & Grossman, T. (2023). Studying the effect of AI code generators on supporting novice learners in introductory programming. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Article 172, pp. 1-23). Association for Computing Machinery. <https://doi.org/10.1145/3544548.3580919>
- Liang, W., Yuksekgonul, M., Mao, Y., Wu, E., & Zou, J. (2023). GPT detectors are biased against non-native English writers. *Patterns*, 4(7), Article 100779. <https://doi.org/10.1016/j.patter.2023.100779>.
- Lodge, J. M., Howard, S., Bearman, M., Dawson, P., & Associates. (2023). *Assessment reform for the age of artificial intelligence*. Tertiary Education Quality and Standards Agency. <https://www.teqsa.gov.au/guides-resources/resources/corporate-publications/assessment-reform-age-artificial-intelligence>.
- Menon, P. (2023). Exploring GitHub Copilot assistance for working with classes in a programming course. *Issues in Information Systems*, 24(4), 66–81. https://doi.org/10.48009/4_iis_2023_106

- Menon, P. (2025). Quantifying shortfalls in students' AI-supported programming practices. *Issues in Information Systems*, 26(1), 458–471. https://doi.org/10.48009/1_iis_134
- Messick, S. (1995). Validity of psychological assessment: Validation of inferences from persons' responses and performances as scientific inquiry into score meaning. *American Psychologist*, 50(9), 741-749. <https://doi.org/10.1037/0003-066X.50.9.741>
- Perkins, M., Furze, L., Roe, J., & MacVaugh, J. (2024). The Artificial Intelligence Assessment Scale (AIAS): A framework for ethical integration of generative AI in educational assessment. *Journal of University Teaching and Learning Practice*, 21(6). <https://doi.org/10.53761/q3azde36>
- Prather, J., Reeves, B. N., Denny, P., Becker, B. A., Leinonen, J., Luxton-Reilly, A., Powell, G., Finnie-Ansley, J., & Santos, E. A. (2023). 'It's weird that it knows what I want': Usability and interactions with Copilot for novice programmers. *ACM Transactions on Computer-Human Interaction*, 31(1), Article 4. <https://doi.org/10.1145/3617367>
- Weber-Wulff, D., Anohina-Naumeca, A., Bjelobaba, S., Foltyniek, T., Guerrero-Dib, J., Popoola, O., Sigut, P., & Waddington, L. (2023). Testing of detection tools for AI-generated text. *International Journal for Educational Integrity*, 19, Article 26. <https://doi.org/10.1007/s40979-023-00146-z>
- Winne, P. H. (2020). Construct and consequential validity for learning analytics based on trace data. *Computers in Human Behavior*, 112, Article 106457. <https://doi.org/10.1016/j.chb.2020.106457>
- Winne, P. H., & Hadwin, A. F. (1998). Studying as self-regulated learning. In D. J. Hacker, J. Dunlosky, & A. C. Graesser (Eds.), *Metacognition in educational theory and practice* (pp. 277-304). Lawrence Erlbaum Associates.
- Zimmerman, B. J. (2000). Attaining self-regulation: A social cognitive perspective. In M. Boekaerts, P. R. Pintrich, & M. Zeidner (Eds.), *Handbook of self-regulation* (pp. 13-39). Academic Press.

APPENDIX

Table A. Example Written Mini-Defense Questions by IS Domain

IS Domain	Example written mini-defense question
Programming	Trace what happens when an empty string is submitted to your validation function, and explain why your code responds the way it does.
Systems Analysis	Why does this data flow cross the boundary of Process 2 rather than Process 3, and what would break if it did not?
Database design	Why is this relationship one-to-many rather than many-to-many, and what would change in your model if a customer could have two billing addresses?
Analytics	Why did you choose this method for handling missing values, and how did you check that it did not bias your result?
Cybersecurity	Walk through how your input-sanitization step would, and would not, stop the injection

IS Domain	Example written mini-defense question
	payload listed in your test matrix.