ISSN: 2473-4901
v12 n6773

From Construction to Verification: Redesigning the Systems Analysis and Design Course for the Era of Artificial Intelligence

John R. Drake
drakejo@ecu.edu
East Carolina University
Greenville, NC, 27858, USA

Andrew Bowman
Bowman-a@mssu.edu
Missouri Southern State University
Joplin, MO, 64801, USA

Abstract

Frontier artificial intelligence (AI) models now perform many of the constituent tasks taught in the systems analysis and design (SA&D) course, from synthesizing stakeholder requirements to generating conceptual models, interface prototypes, and production code. Yet the information systems education literature lacks a course-level account of how this capability should reconfigure what and how we teach. The key question for SA&D courses become how to transform assignments so students do not accept flawed results, miss systematic AI errors, and hit a competency ceiling due to a lack of practice. To address this problem, this article proposes a task-based analysis of AI-driven change in analyst work into the SA&D pedagogy discourse. We decompose SA&D into four task domains (requirements elicitation, conceptual modeling, application and interface design, and implementation) and identify, for each domain, which tasks frontier models absorb, which judgments remain with the analyst, and what the corresponding course module must therefore teach. The analysis yields a verification-centric course design grounded in the meaningful learning framework and organized around three principles: students construct before they generate, every AI contribution requires a documented accept-modify-reject decision, and assessment targets judgment rather than artifacts. The design confronts an expertise paradox in which the evaluative skill graduates now need is predicated upon exactly the constructive practice that AI makes unnecessary. Module-level learning objectives, redesigned assignments, assessment strategies, and implementation considerations for Management Information System programs are presented, along with an evaluation agenda for future research.

Keywords: systems analysis and design, generative artificial intelligence, IS pedagogy, meaningful learning, course design, verification skill

ISSN: 2473-4901
v12 n6773

From Construction to Verification: Redesigning the Systems Analysis and Design Course for the Era of Artificial Intelligence

John Drake and Andrew Bowman

1. INTRODUCTION

Frontier artificial intelligence (AI) models, which consist of the most recent and capable models in the world, can now generate the signature deliverables of the systems analysis and design (SA&D) course. A student with access to a current frontier model can submit interview transcripts, operational logs, and process documentation to one of these models and receive in return a structured requirements catalog, a draft entity-relationship diagram, a process model, and a working interactive prototype, each of high quality. For the instructor, the assessment of learning question becomes less clear: what did the student learn, and what competence does the grade certify?

This situation confronts every instructor of SA&D, and it raises the question: when frontier AI can perform many of the tasks a SA&D course exists to teach, what should the course teach instead, and how? Two literatures speak to this question, but they have not been joined. The first is a growing body of IS education research on generative AI in the classroom, which has produced teaching cases, tool-comparison studies, instructor guides, and frameworks for assigning AI to students (Bekkering & Harrington, 2025; Firth & Triche, 2024; Gimpel et al., 2025; Mollick & Mollick, 2023). This work is valuable, but it largely treats AI as a new topic or a new tool inserted into an otherwise unchanged course. The second is the SA&D pedagogy embodied in standard textbooks and in the IS field's task-level research tradition on requirements determination, which decomposed analyst work into elicitation strategies, prompting techniques, cognitive limitations, and stopping decisions (Browne & Rogich, 2001; Dennis et al., 2021; Pitts & Browne, 2004). This tradition offers an exquisitely detailed map of the tasks we teach, but it predates the technology now absorbing them. Early cross-institutional evidence suggests students who use generative AI in SA&D coursework without deliberate scaffolding accept flawed results uncritically, miss systematic AI errors such as misclassified system boundaries, and hit a competency ceiling in identifying gaps in coverage (Elkhodr & Gide, 2025).

To that end, the purpose of this article is to extend a task-based account of technological change into the discourse on SA&D pedagogy. We decompose SA&D into four task domains (requirements elicitation, conceptual modeling, application and user interface design, and implementation) and analyze, within each, which tasks frontier models can perform well now and which judgments should remain human. A consistent pattern emerges: generation migrates to the machine while validation, judgment, and negotiation concentrate in the human role. We call this the verification-centric model of analyst work, and we argue that it should become the organizing principle of the SA&D course. Because frontier models can generate course artifacts quickly and proficiently, learning outcomes of the course should focus more on verifying the artifacts, rather than producing them. This shift demands a redesign of learning objectives, assignments, and assessment. It is not enough to just insert an AI unit into an unchanged syllabus. The redesign is grounded in the meaningful learning framework, which directs pedagogical attention toward cognitive engagement and learning outcomes rather than instructional techniques (Ausubel, 1968; Drake, 2012; Mayer, 2002).

This article is structured as follows. First, we establish the conceptual background, characterizing SA&D as a task system, summarizing the capabilities of frontier models, and deriving the verification-centric model and the expertise paradox it creates for educators. Second, we propose a verification-centric course design, specifying for each of the four task domains the learning objectives, redesigned assignments, and assessment focus that follow from the analysis. Third, we address assessment and academic integrity in an AI-saturated course. Last, we discuss implementation considerations for MIS programs, the limitations of the design, and an agenda for evaluating it.

2. CONCEPTUAL BACKGROUND

2.1 Systems Analysis and Design as a Task System

Systems analysis and design is the process by which organizations determine information requirements, specify system behavior, design technical and interface architectures, and construct working software (Dennis et al., 2021). The SA&D course has always taught this process as a sequence of distinguishable tasks rather than a monolithic activity, regardless of whether focused on traditional waterfall approach or newer agile approach. The requirements determination phase, for example, consists of selecting elicitation strategies suited to the application domain, deploying prompting tactics to surface requirements stakeholders cannot articulate (Browne & Rogich, 2001), and deciding when enough information has been gathered, a judgment analysts make using cognitive stopping rules rather than objective completeness criteria (Pitts & Browne, 2004). This task-level decomposition matters for pedagogy because automation does not usually replace an entire job role, but rather tasks and subtasks within that role.

2.2 The Capabilities of Frontier Models

Frontier models are the most capable general-purpose AI systems available at a given time: large multimodal models capable of long-context reasoning, code generation, tool use, and increasingly autonomous multi-step operation. Four capabilities bear directly on SA&D coursework. First, they synthesize large volumes of unstructured text, ingesting interview transcripts and documentation at a scale beyond student capacity. Second, they translate between representations, rendering a narrative process description as an entity-relationship diagram (ERD), a Unified Modeling Language (UML) specification, or executable code, and back again. Third, they generate working artifacts, including interfaces, schemas, application logic, and test suites, directly from natural language, with controlled studies documenting substantial productivity gains in programming tasks (Peng et al., 2023). And fourth, when operating agentically, they decompose goals into subtasks and iterate toward working software with intermittent human direction.

While these models express increasing capabilities, they do so unevenly across various tasks. Dell'Acqua et al. (2023) characterize this unevenness as a jagged frontier: tasks of seemingly comparable difficulty fall on opposite sides of the boundary of AI competence. People who cannot perceive the unevenness err in both directions, over-relying on AI output inside its zone of failure and under-relying on it inside its zone of strength. Experienced professionals can make these perceptions more easily. However, students will soon be thrust into a world where they will need to make the same perceptions.

For educators, the jagged frontier is the central instructional fact about these tools. Students need to know how to use these tools for their career, but struggle to use them well. They often struggle to verify the output. Most student groups eventually moved beyond passive acceptance of AI output, but none proactively identified gaps that both human and AI analysis had overlooked, and more than half failed to detect AI misclassification of system boundaries (Elkhodr & Gide, 2025). Teaching students to perceive the boundary becomes the critical learning objective for the course.

2.3 The Verification-Centric Model of Analyst Work

Applying the task-based model of technological change, in which occupations are bundles of tasks that technology substitutes for or complements selectively (Autor, 2015), to the four domains of SA&D reveals a consistent pattern. In requirements elicitation, frontier models show strong skills in transcribing, synthesizing, and mining textual sources, but struggle to decide which stakeholders matter, are slow to build the trust with those stakeholders, or cannot bear accountability for the stopping decision. In conceptual modeling, frontier models draft syntactically fluent diagrams, but struggle to confirm if those diagrams match the business reality. In interface design, frontier models collapse the traditional fidelity progression from sketch to prototype, but struggle to judge intent, accessibility, and taste (Norman, 2013). In implementation, frontier models generate code cheaply, but struggle to independently generate the specifications for successful applications. Brooks (1987) distinguished the accidental complexity of construction from the essential complexity of specification. Frontier models, by attacking the accident, make the essence newly visible, while verification of generated code becomes the expensive human task. In essence, generation occurs with frontier models, and validation,

judgment, and negotiation concentrate in the human role. We call this the verification-centric model of analyst work, and it defines the profile toward which the SA&D course must now aim.

2.4 The Expertise Paradox and Meaningful Learning

The verification-centric model creates a design problem that we argue should anchor the entire course: an expertise paradox. The skills needed to verify artifacts, such as detecting when business states are improperly represented in an entity-relationship diagram, spotting a missing exception path in a process model, or catching architectural drift in code, have traditionally been built by constructing the artifacts that frontier models now absorb. Junior analysts learned to evaluate models by drawing them. Junior developers learned to review code by writing it. Student interaction with the artifacts and learning-by-doing not only improves their critical thinking but also improves the student's motivation to evaluate and continue learning the content (Iftikhar et al., 2022). The classroom evidence is consistent with this concern: student evaluative competence plateaus without deliberate scaffolding, and even trained awareness proves fragile, with accessibility considerations appearing in 85% of groups' requirements yet only 10% of their architectural designs (Elkhodr & Gide, 2025).

Elkhodr and Gide create a pedagogical framework with SAGE (Structured AI-Guided Education) which is designed to develop the ability of students to evaluate the outputs of generative AI instead of accepting them as they come. This framework embraces the use of AI instead of restricting it while students will need to conduct metacognitive analysis of AI to better understand the strengths and limitations of the tool.

The meaningful learning framework supplies the pedagogical response. In the Ausubelian tradition, learning is meaningful to the extent that learners actively anchor new material to their existing cognitive structure, whereas rote processing produces knowledge that neither transfers nor supports evaluation (Ausubel, 1968; Mayer, 2002). Extant literature critically examined the active learning tradition in IS education and argued that its emphasis is misplaced: what determines learning is not the instructional technique but the cognitive engagement and outcomes the design produces (Drake, 2012). With such an approach, lectures, cases, and activities can each yield meaningful learning when they are structured around that engagement. However, they can also yield rote learning when cognitive engagement is not emphasized. Applied to SA&D courses, instructors might use AI to enhanced cognitive engagement or might misplace emphasis in using AI, mistaking busy interaction with a powerful tool for engagement. Recent evidence sharpens the point: students who use generative AI in a mastery-oriented approach, constructing and augmenting their own knowledge, achieve higher-order learning outcomes, while students who use it procedurally, accepting output at face value, achieve lower outcomes despite identical tool access (Pallant et al., 2025).

Two design implications follow, and backward design supplies the method for acting on them, directing the instructor to define the desired graduate competence first and derive activities from it (Wiggins & McTighe, 2005). First, because meaningful evaluation is predicated upon an existing cognitive structure to evaluate against, students must build that structure through calibrated construction before generation begins. Second, because outcomes rather than techniques are the proper object of design, assessment must target the quality of judgment rather than the polish of artifacts. This framing extends the requirements determination tradition by showing that the evaluation tasks it treated as secondary to elicitation now constitute the core of what the SA&D course must teach, and it extends the meaningful learning framework by specifying what cognitive engagement looks like when the artifacts themselves are machine-generated.

3. A VERIFICATION-CENTRIC COURSE DESIGN

The design that follows assumes a junior- or senior-level SA&D course in an MIS program, organized around a semester-long team project with a real or realistic client, consistent with the project-based structure most SA&D courses already employ (Dennis et al., 2021). The redesign does not add an AI unit to this structure. Instead, it reorganizes each existing module around three principles assuming usage of AI, each grounded in the meaningful learning framework. First, students should perform a critical action manually before generating the same artifact with an AI. A step which should be accomplished through a bounded portion of the project or unrelated structured exercises before

prompting an AI to perform on the remainder of the project, because evaluative skill is predicated upon the cognitive structure that constructive practice builds (Drake, 2012). Second, every AI contribution requires a decision. Students document whether they accepted, modified, or rejected each substantive AI output and why, converting tool use from procedural processing into the mastery-oriented engagement that produces higher learning outcomes (Elkhodr & Gide, 2025; Pallant et al., 2025). Third, assess judgment, not artifacts: since polished artifacts are now cheap, grades attach to the quality of validation, critique, and decision rationale rather than to the deliverables themselves, an application of the principle that outcomes rather than techniques are the proper object of pedagogical design (Drake, 2012). Table 1 summarizes the design; the four subsections elaborate it.

Course module	Tasks absorbed by frontier models	Learning objective (verification-centric)	Exemplar redesigned assignment
Requirements elicitation	Transcription and synthesis of stakeholder input; mining of logs and documents; conflict and ambiguity detection	Validate machine-synthesized requirements; negotiate surfaced conflicts; justify the stopping decision	Live stakeholder interviews, then a validation memo comparing student-derived and AI-derived catalogs; conflict negotiation role-play
Conceptual modeling	Drafting ER, UML, and process models from narratives; cross-model consistency; notation translation	Detect semantic defects in syntactically fluent generated models; choose abstraction levels	Manual model construction on a bounded scope, then a seeded-defect hunt in AI-generated models with written critique
Application and UI design	High-fidelity interactive prototype generation; layout and component design	Curate among generated alternatives; apply accessibility and ethical criteria; articulate design intent	Generate three prototype alternatives, then a curation memo defending the selection on user, accessibility, and ethical grounds
Implementation and testing	Code generation; test-suite generation; refactoring; agentic construction	Write precise specifications; review generated code for security, drift, and divergence from intent; make accountable acceptance decisions	Specification-to-code exercise, then structured review of the generated codebase and a signed acceptance memo

Table 1. A Verification-Centric Redesign of the Four SA&D Course Modules

3.1 Module 1: Requirements Elicitation

Requirements determination remains the most consequential and failure-prone phase of development, even with agile development (Schön et al., 2017), because errors made here propagate at compounding cost downstream. With a redesigned module, students still conduct live interviews with the project sponsor, because stakeholder selection, trust-building, and reading political and tacit context are residual human tasks that no transcript synthesis replaces. Students first produce a manually derived requirements catalog from their own notes on a bounded slice of the project. Only then do they submit full transcripts, logs, and documentation to a frontier model and receive its far more comprehensive catalog.

A proper assessment of this task might include a validation memo comparing the student generated requirements catalog with the AI generated requirements catalog. In such a memo, students document which machine-identified requirements are spurious, which are salient to the sponsor's actual priorities, and which conflicts the broader synthesis surfaced. The comparison is where meaningful learning occurs, because it forces students to anchor the machine's output against the cognitive structure their own elicitation built. Because AI-assisted elicitation consults more sources at lower marginal cost, it reliably surfaces stakeholder conflicts that manual elicitation might miss, and conflict resolution is a

quintessentially human negotiation task. The module could close with a negotiation role-play in which teams resolve a machine-surfaced conflict between stakeholders or a short essay defending the team's stopping decision in terms of validation confidence rather than acquisition fatigue (Pitts & Browne, 2004). The module thus converts the classic elicitation curriculum from information acquisition to information validation without abandoning the interpersonal skills that define it.

3.2 Module 2: Conceptual Modeling

The jagged frontier runs straight through conceptual modeling. While frontier models are becoming progressively better at generating models, those models are syntactically fluent, yet semantically fragile. A generated entity-relationship diagram may be well-formed in its notation while collapsing two entities the business treats as distinct. Likewise, a process model may use correct notation while omitting an exception path that occurs weekly in the client's operations. Detecting such defects requires exactly the conceptual thinking students no longer exercise if generation replaces construction, highlighting the expertise paradox.

To address this paradox, we recommend the module follows a strict construct-then-critique sequence. Students first build entity-relationship and process models manually for a bounded scope of the project. This construction is graded, not because the artifacts matter but because the model generation builds the anchoring knowledge and capabilities that meaningful evaluation requires. Only then do students use an AI to generate a full model suite from the project narrative. With those generated models, students hunt for defects, including defects the instructor may deliberately seed into a generation prompt. The seeded categories mirror the error patterns documented empirically (Elkhodr & Gide, 2025); misclassified system boundaries, data management errors, missing exception handling, etc. The graded deliverable is the written critique, in which every accepted model element, modification, and rejection carries a rationale. It is important to note that the point of manual construction here is calibrated by the instructor to be enough to build the evaluative capabilities, but not so much that the course pretends generation does not exist.

3.3 Module 3: Application and Interface Design

Traditional interface design follows a fidelity progression in which sketches precede wireframes, wireframes precede mockups, and mockups precede prototypes (Nielsen, 1993). Frontier models now yield a high-fidelity, interactive prototype in minutes, however the lack of support for iterative processes makes it difficult to provide concrete changes (Yuan et al., 2026). For the SA&D course, this fidelity collapse converts requirements validation from specification review into prototype interaction. Student teams can put functional prototypes in front of the sponsor within minutes. Although agile methodologies ceased to separate requirements determination from design, AI generated prototypes make this a continuous conversation.

With such capabilities, decisions on design can encapsulate intent and taste quickly through deliberate curation. Teams can meet with their sponsor with at least three distinct prototype alternatives for the same requirements, then quickly iterate during their meeting to ensure user fit, accessibility compliance, and ethical acceptability (Norman, 2013). The curation framing is an analogy to the museum curator, who creates nothing in the gallery yet determines what the visitor encounters; the analogy illuminates where student value now resides, in selection, arrangement, and intent rather than fabrication. Because classroom evidence shows accessibility awareness collapses mid-lifecycle even among students who considered it during requirements (Elkhodr & Gide, 2025), the module embeds an accessibility checkpoint at every stage rather than teaching it once.

As an example of common issues students may want to check off for errors consider looking at visual and design quality elements like consistency (no lorem ipsum text), readability, that icons and images used are appropriate for the design, accessibility, and ease of use.

3.4 Module 4: Implementation and Testing

AI capabilities have exhibited their strongest use case in software engineering. Generated code completes assignments faster but does not always meet the stated requirements, and students who cannot specify precisely cannot detect the shortfall (Bekkering & Harrington, 2025; Peng et al., 2023). How can a SA&D course module exploit this property pedagogically? One suggestion is for students to

first write a precise natural-language specification for a system component. From that specification, an agentic coding tool then builds exactly what was specified, including any ambiguities introduced by the students. The gap between intent and output becomes the lesson. When code is cheap, students can experiment with specifications to better observe how changes impact output.

Additionally, the course module could invert the traditional programming assignment. Rather than writing code, students review it. Students could conduct a structured walkthrough of the generated codebase, searching for security defects, architectural drift, and silent divergence from the specification. After the walkthrough, students sign an acceptance memo in which the team formally accepts accountability for shipping the component. The signature is deliberately ceremonial. It teaches the professional reality that a human remains accountable for the acceptance decision no matter how much of the construction a machine performed.

4. ASSESSMENT AND ACADEMIC INTEGRITY

Verification-centric assessment follows from the third design principle: when artifacts are cheap, grading artifacts rewards prompt access rather than competence, and it invites exactly the procedural, regurgitative tool use that produces inferior learning outcomes (Pallant et al., 2025). Three instruments carry the grade weight. The first is the AI decision log, a running record in which teams document each substantive AI contribution and the accept-modify-reject decision made about it, with rationale, an approach shown to move students beyond passive acceptance (Elkhodr & Gide, 2025). The second is the set of module memos described above, which grade the delta between what the machine produced and what the team shipped. The third is a brief oral defense at each milestone, in which any team member must explain and justify any decision in the log; the defense verifies that judgment is distributed across the team rather than concentrated in one member or outsourced entirely.

This assessment architecture also resolves the academic integrity question that dominates faculty discussion of generative AI. The course does not prohibit AI use; it requires and audits it. Integrity violations are redefined from using the tool to misrepresenting the judgment, whether through an undocumented AI contribution or an unsupportable rationale in the log. This reframing is consistent with emerging guidance that positions AI as an assigned collaborator whose use is structured rather than policed (Gimpel et al., 2025; Mollick & Mollick, 2023), and it prepares students for workplaces that will expect exactly this documented accountability.

5. IMPLEMENTATION CONSIDERATIONS FOR MIS PROGRAMS

Three practical considerations govern adoption. The first is tool access. The course design requires that every student have access to a current frontier model with sufficient usage limits. Free licenses currently have limits to use that make them difficult to implement into classes and discourage experimental use to understand the limitations of the AI tools. Institutional licensing or departmental accounts prevent the design from privileging students who can afford premium subscriptions, and they give the instructor control over which model version students use, which matters for seeding defects reproducibly.

The second is instructor preparation. The seeded-defect exercises and oral defenses demand that instructors themselves work fluently on both sides of the jagged frontier. The course's generation steps must be rehearsed each semester due to shifts in model capabilities. Practical instructor guidance for this preparation is beginning to appear in the IS education literature (Gimpel et al., 2025). To the extent that a department treats this as a shared asset, maintaining a common bank of seeded-defect prompts, curation rubrics, and acceptance-memo templates, the preparation cost amortizes across sections.

The third is curricular placement. The design aligns naturally with the competency orientation of IS2020, which frames the curriculum in terms of demonstrable competencies rather than topic coverage (Leidig & Salmela, 2022); verification, curation, and specification precision are competencies in exactly this sense, and they map onto the systems analysis and application development competency. Programs should also mind prerequisite sequencing: the construct-before-generate principle presumes students arrive with introductory data modeling and programming exposure, so the redesigned SA&D course belongs after those foundations, not in place of them.

6. DISCUSSION AND LIMITATIONS

This article makes three contributions to IS education. First, it joins two literatures that have developed in isolation: the task-level SA&D research tradition, which supplies the task taxonomy, and the emerging classroom literature on generative AI, which supplies the tool evidence but has lacked a principled account of what the SA&D course should now certify. Second, it converts the verification-centric model of analyst work into an actionable course architecture (construct before generating, document every decision, assess judgment) rather than leaving instructors to improvise module by module. Third, it grounds AI-era course design in the meaningful learning framework and the AI centered SAGE framework rather than in technique novelty, extending an argument made previously about active learning to the present debate about AI in the classroom: in both cases, the pedagogical question is not which technique or tool the course employs but whether the design produces cognitive engagement and demonstrable outcomes (Drake, 2012). This grounding cuts against both prohibition and uncritical adoption, and it names the expertise paradox as the central design constraint that any resolution must address.

Three limitations bound the design. First, it has not yet been evaluated as a whole; the module elements rest on documented classroom evidence (Bekkering & Harrington, 2025; Elkhodr & Gide, 2025; Pallant et al., 2025) and parts have been introduced into the authors' classes, but the integrated course awaits assessment, and we invite instructors to adopt and study it. The natural evaluation design is a pre-post measure of semantic-defect detection, an instrument the field currently lacks and should build. Second, frontier model capabilities are a moving target, and specific assignments will require regular recalibration. The verification-centric architecture, however, depends only on the persistence of human accountability for system outcomes, which we expect to through multiple iterations of improvements. Third, the design assumes a project-based course with sponsor access, and it may transfer imperfectly to large-enrollment or fully asynchronous sections, where the negotiation role-plays and oral defenses require adaptation.

7. CONCLUSION

System Analysts are learning how to delegate SA&D deliverables to frontier models. Students who mimic this behavior are not misbehaving but may be short cutting skills they need later in their careers. The failure belongs to any course that continues to grade construction of artifacts in a world that has automated it. Across requirements elicitation, conceptual modeling, design, and implementation, the pattern is consistent: the best approach is AI tools generate the artifacts, and the human validates, judges, and negotiates. A SA&D course should teach accordingly, requiring enough construction to build the cognitive structure that judgment requires, generating at the speed the tools afford, and grading the judgment rather than the artifact. The verification-centric analyst is, in essence, what MIS programs have always claimed to produce, and meaningful learning has always been the standard by which our course designs should be judged. The tools have finally made both claims testable, one course at a time.

REFERENCES

Ausubel, D. P. (1968). *Educational psychology: A cognitive view.* Holt, Rinehart and Winston.

Autor, D. H. (2015). Why are there still so many jobs? The history and future of workplace automation. *Journal of Economic Perspectives, 29*(3), 3–30. https://doi.org/10.1257/jep.29.3.3

Bekkering, T. E., & Harrington, P. (2025). If you want something specific, ask for it specifically: A comparison of generative AI solutions and textbook solutions in an introductory programming course. *Information Systems Education Journal, 23*(1), 4–22. https://doi.org/10.62273/YQWP1758

Brooks, F. P. (1987). No silver bullet: Essence and accidents of software engineering. *Computer, 20*(4), 10–19. https://doi.org/10.1109/MC.1987.1663532

Browne, G. J., & Rogich, M. B. (2001). An empirical investigation of user requirements elicitation: Comparing the effectiveness of prompting techniques. *Journal of Management Information Systems, 17*(4), 223–249. https://doi.org/10.1080/07421222.2001.11045665

Dell'Acqua, F., McFowland, E., Mollick, E., Lifshitz, H., Kellogg, K. C., Rajendran, S., Krayer, L., Candelon, F., & Lakhani, K. R. (2026). Navigating the Jagged Technological Frontier: Field Experimental Evidence of the Effects of Artificial Intelligence on Knowledge Worker Productivity and Quality. *Organization Science*, *37*(2), 403–423. https://doi.org/10.1287/orsc.2025.21838

Dennis, A., Wixom, B. H., & Roth, R. M. (2021). *Systems analysis and design* (8th ed.). Wiley.

Drake, J. R. (2012). A critical analysis of active learning and an alternative pedagogical framework for introductory information systems courses. *Journal of Information Technology Education: Innovations in Practice, 11*, 39–52. https://doi.org/10.28945/1546

Elkhodr, M. & Gide, E. (2025). *Embedding generative AI into systems analysis and design curriculum: Framework, case study, and cross-campus empirical evidence.* arXiv. https://arxiv.org/abs/2511.17515

Firth, D. & Triche, J. (2024). Generative AI in practice: A teaching case in the Introduction to MIS class. *Information Systems Education Journal, 22*(4), 29–47. https://doi.org/10.62273/LDVL8354

Gimpel, H., Hall, K., Decker, S., Eymann, T., Gutheil, N., Lämmermann, L., Braig, N., Maedche, A., Röglinger, M., Ruiner, C., Schoch, M., Schoop, M., Urbach, N., & Vandirk, S. (2025). Using generative AI in higher education: A guide for instructors. *Journal of Information Systems Education, 36*(3), 237–256. https://doi.org/10.62273/QLLG7172

Iftikhar, S., Guerrero-Roldán, A.-E., & Mor, E. (2022). Practice Promotes Learning: Analyzing Students' Acceptance of a Learning-by-Doing Online Programming Learning Tool. *Applied Sciences*, *12*(24), 12613. https://doi.org/10.3390/app122412613

Leidig, P. M. & Salmela, H. (2022). The ACM/AIS IS2020 competency model for undergraduate programs in information systems: A joint ACM/AIS task force report. *Communications of the Association for Information Systems, 50*. https://doi.org/10.17705/1CAIS.05021

Mayer, R. E. (2002). Rote versus meaningful learning. *Theory Into Practice, 41*(4), 226–232. https://doi.org/10.1016/B978-0-08-052029-2.50007-3

Mollick, E. R. & Mollick, L. (2023). *Assigning AI: Seven approaches for students, with prompts.* SSRN. https://doi.org/10.2139/ssrn.4475995

Nielsen, J. (1993). *Usability engineering.* Academic Press.

Norman, D. A. (2013). *The design of everyday things* (Rev. ed.). Basic Books.

Pallant, J. L., Blijlevens, J., Campbell, A., & Jopp, R. (2026). Mastering knowledge: The impact of generative AI on student learning outcomes. *Studies in Higher Education*, *51*(4), 714–735. https://doi.org/10.1080/03075079.2025.2487570

Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). *The impact of AI on developer productivity: Evidence from GitHub Copilot.* arXiv. https://arxiv.org/abs/2302.06590

Pitts, M. G., & Browne, G. J. (2004). Stopping behavior of systems analysts during information requirements elicitation. *Journal of Management Information Systems, 21*(1), 203–226. https://doi.org/10.1080/07421222.2004.11045795

Schön, E., Thomaschewski, J., & Escalona, M.J. (2017). Agile Requirements Engineering: A systematic literature review. *Computer Standards & Interfaces*, 49(C), 79-91. https://doi.org/10.1016/j.csi.2016.08.011

Wiggins, G., & McTighe, J. (2005). *Understanding by design* (2nd ed.). ASCD.

ISSN: 2473-4901
v12 n6773

Yuan, M., Chen, J., Hu, Y., Feng, S., Xie, M., Mohammadi, G., Xing, Z., & Quigley, A. (2026). Towards Human–AI Synergy in UI Design: Supporting Iterative Generation with LLMs. *ACM Transactions on Computer-Human Interaction*, *33*(1), 1–45. https://doi.org/10.1145/3773035