Trends and Reviews of Technological Engineering in Education
Vol. 2, No. 3 (2026) | ISSN 3071-2211 | Open Access

RESEARCH ARTICLE

A Digital Twin–Enabled Human-in-the-Loop Orchestration Protocol for AI Co-Teaching and Assessment in Engineering Capstone Courses

Songyi Zhuo[1], Xiaomei Yuan[2, *]
[1] City Graduate School, City University Malaysia, Petaling Jaya, Malaysia 46100
[2] Business School, Xi'an Innovation College of Yan'an University, Xi'an City, China 710100
[*] Correspondence: yuanmay@126.com

Received 18 March 2026 · Accepted 22 July 2026 · Published 30 September 2026

Abstract

Engineering capstone teams increasingly use instrumented prototypes, simulations, building information models, and AI assistants, but these resources are seldom coordinated as one inspectable learning workflow. This article specifies a technical design for a course project twin framework and a human-in-the-loop orchestration protocol for AI co-teaching and assessment in engineering capstones. The framework links an engineering-system representation, project-process events, and learning-evidence records while keeping their purposes distinct. It reserves "digital twin" for a bounded subsystem with a declared synchronization relationship: M0 is an instructional model, M1 a one-way shadow, and M2 a controlled synchronized subsystem. AI co-teaching is defined here as instructor-designed task allocation in which AI retrieves, compares, or drafts bounded support; learners test and explain engineering claims; and instructors retain authority over safety, assessment, and milestone decisions. The specification includes a service architecture, typed record contracts, a nine-state workflow with exception paths, an outcome-to-evidence map, governance controls, and a reproducible synthetic bridge-monitoring case. Proposed acceptance criteria distinguish deterministic safety and provenance gates from human-rated feedback quality and future learning outcomes. This is a design specification; no implementation, classroom deployment, or measured outcome is claimed.

Keywords: Engineering Capstone Education; Digital Twin; Human-AI Collaboration; Learning Orchestration; Formative Assessment; Engineering Design; Evidence Provenance

How to cite: Songyi Zhuo & Xiaomei Yuan (2026). A Digital Twin–Enabled Human-in-the-Loop Orchestration Protocol for AI Co-Teaching and Assessment in Engineering Capstone Courses. TRTEE, 2(3), 22–56.
https://doi.org/10.63646/SMLK8377

1. Introduction

Engineering capstone courses ask students to integrate disciplinary analysis, design iteration, implementation, verification, communication, and team coordination. A team may need to interpret an open-ended need, convert it into testable requirements, create a model, choose an architecture, build or simulate a prototype, inspect results, and defend a decision under constraints. The final artifact is only one part of that work. Students must also learn to explain why a model is fit for a purpose, identify where it is uncertain, and revise a design when evidence disagrees with an assumption. Research on engineering design education has long emphasized the iterative and information-rich character of this work [1, 2], while project-based learning research shows that authentic activity needs explicit scaffolding to connect artifact production with learning [3, 4]. Capstone design pedagogy also emphasizes project definition, integration, and the relationship between active work and intended learning [5, 6, 7, 8, 9, 10].

The technical environment of capstone projects is changing. Students may work with sensor streams, CAD or BIM models, simulations, version-controlled code, issue trackers, laboratory equipment, or remote industrial data. Digital-twin approaches can connect a system representation with selected data and operations over time. In education, published work has explored BIM-based engineering capstones, laboratory digital learning environments, and digital-twin learning systems, but the examples leave an instructional design question: how should a course connect project state, bounded AI assistance, individual learning evidence, and accountable human decisions in one traceable workflow [11, 12, 13]? Manufacturing classifications such as Kritzinger et al. [14] inform the model–shadow–twin distinction used here; they are not treated as education cases.

Generative AI adds another layer. It can help explain a model, propose test cases, compare stated alternatives, summarize a project record, or draft rubric-linked formative feedback. These capabilities may help instructors support multiple teams, but they also create specific design risks. A model may hallucinate a sensor interpretation, mistake stale data for a current state, summarize away a dissenting team member’s concern, or convert a tentative engineering observation into a confident recommendation. AI-generated text can appear authoritative even when the underlying twin is incomplete or outside its validated range. Work on AI in education and human-AI collaboration therefore points toward meaningful human control, clear task boundaries, and role-aware interaction rather than generic automation [15, 16, 17, 18, 19, 20, 21, 22, 23]. Engineering education research on student use, academic integrity, and human-teacher collaboration further supports defining permitted assistance and preserving learner authorship [24, 25, 26].

This article specifies a course project twin framework and an orchestration protocol for engineering capstones. “Framework” names the linked course records and services; it does not make every project representation a digital twin. An M0 component is an instructional model, an M1 component is a one-way shadow, and the term digital twin is reserved for a bounded M2 subsystem with a documented synchronization relationship. The framework has three linked but distinguishable representations: engineering-system state, project-process records, and learning evidence. It does not claim that activity traces reveal cognition or that a course workspace is a twin of a learner. The orchestration service routes bounded AI capabilities and human review: AI can prepare an explanation or candidate feedback,

© 2026 TRTEE. Open Access under CC BY 4.0

deterministic tools perform specified calculations, students remain responsible for engineering choices, and instructors retain authority for safety and assessment.

The contribution is an implementation-oriented design specification with five elements. It defines an instructional model/shadow/twin maturity contract; links engineering artifacts, project events, and learning evidence through typed records; specifies a state protocol for bounded assistance, review, learner response, and milestone decisions; maps learning outcomes to observable evidence and role-specific actions; and supplies a synthetic bridge case with proposed verification criteria. The paper presents an analytical design synthesis rather than a systematic review, implementation report, or empirical evaluation.

2. Design Problem and Scope

2.1 The coordination gap in digitally enabled capstones

In many courses, the design model lives in one application, simulation results in another, meeting notes in a third, and rubric judgments in a learning-management system. Even when each tool is individually useful, a team or instructor may not be able to answer basic provenance questions quickly: Which model revision produced this plot? Which requirements were active when the simulation ran? Was this feedback based on a current artifact? Did a student revise the design after the instructor's comment? What evidence supports an individual claim of competence? The challenge is not simply collecting more data. It is preserving the relationships among technical state, learning activity, and accountable decisions.

The system boundary of an engineering project may itself be only partly observable. A bridge-monitoring project might use a digital structural model and a synthetic sensor stream, but neither object captures every property of a real bridge. A model may omit material degradation, boundary-condition variation, environmental influences, or sensor faults. A meaningful teaching design must make those omissions visible. Digital-twin engineering distinguishes models and shadows from twins in terms of their connection and synchronization with an observable physical element; the educational system should retain that distinction rather than use "twin" as a label for any 3D visualization [14, 27, 28].

The coordination problem has a pedagogical dimension as well. Team products are shared, but learning is not necessarily shared equally. One team member may build the simulation while another interprets evidence, and a third may coordinate stakeholder requirements. Click counts, message volume, or time online do not establish contribution quality or individual understanding. Collaborative learning research treats planning, monitoring, and adaptation as socially shared processes, but an AI service should not infer these internal processes from telemetry without validation and consent [29, 22]. The design therefore records explicit decisions and submitted evidence and avoids psychological labels or automated individual rankings.

2.2 Engineering twin versus learner record

The phrase "learning digital twin" can refer to different things: a digital model of an engineering object used for instruction, a digital representation of a laboratory, or a profile that predicts an individual learner's state. This article uses course project twin for the first two project-facing purposes and deliberately does not create a student twin. A student's reflection, explanation, or submitted work may be evidence in the course, but the system does

© 2026 TRTEE. Open Access under CC BY 4.0

not maintain a continuously updated proxy of the student's cognition, motivation, or ability. This boundary is both technically prudent and educationally important: observable project artifacts can support a bounded interpretation, while latent psychological states cannot be read directly from an event log.

The project twin also does not replace a physical system. Its engineering-system layer may connect to a physical prototype, a lab rig, a simulation, or an openly documented dataset. The interface shows the source type and synchronization status. If measurements are simulated, they are labeled simulated. If a connection is delayed or one-way, the interface says so. If no live physical object exists, the system calls the artifact a model or a project representation according to the maturity rules in Section 3. This vocabulary prevents the system from implying validation or fidelity that the course has not established.

2.3 Scope and exclusions

The proposal targets a semester-length or year-long engineering capstone in which teams build, model, analyze, or operate a technical system. It assumes a course owner can define milestones, roles, learning outcomes, rubrics, permitted tools, and review authority. It does not prescribe one learning-management system, one digital-twin vendor, one AI model, or one discipline-specific simulator. It does not claim compliance with accreditation requirements. It does not automate approval of engineering safety, professional certification, grades, student discipline, or progression decisions. Such decisions remain with the responsible human role and the institution's established process.

The article also does not claim that a digital twin is required for every capstone. A twin adds value only when the project needs a versioned connection between a system representation and evidence over time. For a low-complexity design exercise, a conventional model and a well-structured evidence record may be sufficient. The architecture is modular: a course can use the event and provenance protocol without connecting a live device, or can use a simulator and artifact registry before introducing AI. This staged adoption keeps the educational objective primary and avoids treating technology deployment as its own learning outcome.

3. Design Foundations and Requirements

3.1 Digital-twin distinctions for instructional use

Digital-twin literature does not use a single universal definition, but common accounts distinguish a digital model from a one-way digital shadow and a synchronized digital twin. In a model, the representation is updated manually and no automatic data link is assumed. In a shadow, information flows from a physical entity to a digital representation. In a twin, a defined synchronization relationship connects the physical and digital entities, potentially with controlled feedback [14, 28, 30]. This article applies that distinction strictly: M0 denotes an instructional model and is not called a digital twin; M1 denotes a one-way instructional shadow; and only a bounded M2 subsystem with a declared synchronization contract is called a digital twin. For each level, the course also states what learners may inspect, what can change, and which actions remain outside the system boundary.

Technical accounts also emphasize that a twin is a system of models, data connections, and services rather than a single rendered object. Its usefulness depends on the boundary of the observable entity, the structure of the data, synchronization rules, and the function of the

© 2026 TRTEE. Open Access under CC BY 4.0

model in a decision workflow [31, 27, 32, 33]. The present design narrows these general ideas to a course project and makes the instructional purpose and learner-facing limits explicit. A digital representation can support design exploration without being a validated operational twin, and model fidelity should be discussed in relation to the question being taught rather than treated as a visual-quality metric [34, 35].

The maturity levels are intentionally conservative. M0, instructional model, is a versioned engineering representation with documented assumptions and no automatic data synchronization; it is not a digital twin. M1, instructional shadow, receives time-stamped observations from a declared physical source, with no control command

returning to equipment. M2, controlled instructional twin, is reserved for a named subsystem whose state variables synchronize under explicit timing and validity rules; any reversible command is issued only through a separate authorized control service inside a human-approved safety envelope. A course may call only that subsystem a digital twin, not the whole asset. ISO 23247 provides a manufacturing framework that can inform technical interfaces, but it does not certify educational validity or replace the course-specific scope and safety policy [36, 37].

3.2 Instructional orchestration and human-AI roles

Orchestration means coordinating learners, resources, activities, and intervention across a changing class context. The course twin provides a shared, versioned view of selected project state; it does not dictate an instructional response. An orchestration event can arise from a student request, a milestone rule, an explicit missing-evidence condition, or a model-generated alert. Each event records its origin. The teacher can choose to respond, delegate a bounded task to an approved AI service, or leave the workflow unchanged. Teacher-centered co-orchestration work underlines that shared control must be designed to keep teacher judgment usable and visible [38, 39].

Hybrid intelligence emphasizes complementary human and AI strengths [79], while the machines-as-teammates agenda examines potential benefits and harms [83]. These motivate role design, not claims of validated educational performance. Here, co-teaching divides work among the instructor, approved AI capabilities, and learners. AI may retrieve approved material, compare documented alternatives, flag candidate evidence gaps, or draft questions. Instructors set tasks, review consequential outputs, and own course decisions. Learners inspect evidence, test or reject suggestions, and explain their reasoning. AI is neither an equal nor an autonomous instructor: it cannot set requirements, certify safety, grade, or declare learning outcomes achieved. Each capability has an allowed role, input scope, output schema, and prohibited-action list.

3.3 Assessment and feedback as evidence use

Formative assessment and feedback research supports making goals, criteria, evidence, and next actions explicit [40, 41, 42, 78]. Feedback must be interpreted and used in its learning context [80]; a polished paragraph or score is not sufficient. Feedback literacy requires learners to understand, evaluate, and act on feedback [43, 44, 45]. The proposed response path therefore lets learners request clarification, test a claim, explain disagreement, or document a revision.

Rubrics can make criteria inspectable, but a rubric label is not evidence by itself. The system should distinguish the criterion, artifact feature, assessor interpretation, and final judgment. It should also keep team-level and individual-level evidence separate. A team simulation

© 2026 TRTEE. Open Access under CC BY 4.0

report can show that the group compared scenarios; an oral explanation or individual design memo may be needed to assess a student's own reasoning. This separation supports evaluative judgment and formative self-assessment instead of assuming that a successful team artifact proves each learner has achieved the outcome [46, 47, 48].

Feedback and learner responses form a sequence, not a score. A capability scaffolds reasoning while learners retain consequential choices and explanations [49, 50]. Cyclical self-regulation models connect planning, monitoring, and reflection [82]; they inform the response path without demonstrating gains here. Each feedback action links to a criterion, source evidence, a learner response, and a human review rule.

3.4 Design derivation and traceability

The design was derived through an analytical synthesis of the digital-twin distinctions and architecture sources cited in Sections 2–3, research on project-based engineering learning and feedback, and human–AI interaction and teacher co-orchestration work. The procedure had four steps: (1) set the system boundary and exclude a continuously inferred learner model; (2) translate pedagogical and technical needs into requirements R1–R8; (3) assign each requirement to a service, record contract, role permission, and workflow state; and (4) walk the contracts through a synthetic case and define tests that could falsify the design. This is not a systematic search, quality appraisal, expert consensus exercise, or validation study.

The resulting traceability chain is design premise → requirement → architecture or data contract → state transition → verification case. Table 1 records the requirement-level contract and the evidence a future prototype would need to produce. The manuscript demonstrates internal specification and a synthetic walkthrough only; it does not claim that the proposed tests have been run.

3.5 System requirements

Table 1 translates the design foundations into requirements. R1–R2 define representation scope and maturity; R3–R4 define provenance and bounded AI; R5–R6 preserve human authority and distinct evidence types; R7–R8 define minimization and safe fallback. The final column describes proposed acceptance evidence for a future implementation, not completed test results.

© 2026 TRTEE. Open Access under CC BY 4.0

Table 1. Design Requirements for the Course Project Twin and Orchestration Protocol

ID	Requirement	Required system behavior	Proposed acceptance evidence
R1	Representation scope	Declare physical or simulated entity boundary, variables, source type, assumptions, and exclusions.	Twin card shows boundary, source, and limitations to learners.
R2	Maturity honesty	Label each component M0 model, M1 one-way shadow, or M2 controlled synchronized subsystem; show direction and latency.	Contract tests reject unsupported maturity labels and control paths.
R3	Version provenance	Bind observations, simulations, AI responses, evidence links, and judgments to exact object and policy versions.	Lineage traversal resolves required references; stale links are blocked.
R4	Bounded AI	Allow only named capabilities with role, input, output, retention, citation, and refusal rules.	Out-of-scope, unauthorized, and injection fixtures are refused or quarantined.
R5	Human authority	Reserve safety, milestone, grade, and appeal decisions for named human roles.	No AI or service account can write a consequential decision state.
R6	Evidence separation	Keep team product, team process, and individual learning records typed and distinct.	Rubric and ledger tests reject unsupported evidence-type substitution.
R7	Minimization	Resolve only task-authorized data; declare provider, access class, and separate retention rules.	Access, provider-use, expiry, and deletion tests pass for seeded cases.
R8	Safe fallback	Expose stale, missing, invalid, and unavailable states; route to a human or safe alternative.	Fault injection blocks dependent transitions without fabricated output.

4. Course Project Twin Framework: Data Model and Maturity

4.1 Three linked representations

The course project twin framework contains three representations that link through stable identifiers but remain independently inspectable. The engineering-system representation stores the system boundary, component and interface model, requirements, parameter sets, measurements, simulation outputs, test records, and known limitations. Each item carries source type, version, unit, time or scenario context, and provenance. A model variable named “beam displacement” is not comparable across artifacts unless its units, coordinate frame, sensor or solver source, and version are known. At M0 this is an instructional model; the framework does not relabel it a digital twin. The registry rejects or flags values whose required contract is incomplete.

The project-process representation stores declared work events: requirement decisions, model revisions, test plans, review requests, team agreements, and milestone submissions. It is not a raw surveillance stream. The event may point to a commit, a design-file revision, a signed decision record, or a meeting action selected by the team. It identifies the actor only

© 2026 TRTEE. Open Access under CC BY 4.0

where necessary for course workflow and displays the relevant access label. It does not infer effort, attention, or competence from passive telemetry. Team processes such as planning and monitoring are supported by opportunities to state goals and review progress rather than by hidden behavioral scoring.

The learning-evidence representation maps course outcomes and rubric criteria to inspectable project evidence. An evidence link states which artifact or explanation can support which claim, the relevant artifact version, who produced or reviewed it, and what limitation applies. A link can be proposed, accepted by an educator, challenged, or superseded. It is not a binary truth claim that an outcome has been mastered. Assessment records additionally store the assessor, rubric version, criterion-level judgment, rationale, learner response, and review state. The design permits multiple evidence sources and preserves disagreement rather than averaging away differences.

4.2 Twin maturity contract

Every project instance declares one maturity level for each system component. A course may include a live sensor shadow for one subsystem and an offline model for another. The maturity record includes entity boundary, variables, update direction, update frequency, data delay, model-validity range, control permissions, and known exclusions. A change in any of these values creates a new version. A stale observation is visibly stale; a model extrapolation outside the defined range is flagged; and any command path is disabled unless a human-approved safety envelope is active.

The M0/M1/M2 labels are workflow metadata, not certification. M1 requires an identified physical observation source, declared timing and validity rules, and no command path. M2 requires a bounded physical subsystem, synchronization contract, command bounds, authorization, emergency stop, sandbox status, and a human owner. Missing requirements block the affected connection. Any fallback maturity must be explicitly declared in a new version and shown to users; failed controls cannot be bypassed by silent downgrading. A model confidence score cannot promote maturity. These are proposed contract checks.

4.3 Canonical records and event envelope

Every record uses a common immutable event envelope: event_id (UUID), schema_version, course_id, project_id, team_id, actor_id, actor_role, event_type, event_time (UTC), source_system, object_ref, object_version, policy_version, and access_class. reason_code, parent_event_id, and evidence_refs are optional. Record-specific payloads are separately typed and validated. A TwinObservation requires observation_id, component_id, variable_id, value, unit, event_time, ingest_time, source_ref, and quality_status; calibration_ref is required when the source requires calibration. A ModelRun requires run_id, model_version, input_refs, parameter_hash, solver and version, execution_environment, output_ref, and run_status. A LearningEvidenceLink uses one canonical record name throughout this manuscript and requires link_id, outcome_id, criterion_id, artifact_ref, artifact_version, evidence_span, claim, limitation, status, and author_id; reviewer_id is optional until review. The schema records links and provenance, not a claim that an outcome is mastered.

A minimal LearningEvidenceLink payload illustrates the contract; identifiers and labels below are synthetic and are not learner data:

{"record_type":"LearningEvidenceLink","schema_version":"1.0",
"link_id":"00000000-0000-4000-8000-000000000014","outcome_id":"CO-3",

© 2026 TRTEE. Open Access under CC BY 4.0

"criterion_id":"R-3B","artifact_ref":"model/B-04","artifact_version":"4",
"evidence_span":"scenario/perturbation","claim":"B-04 supports this comparison only",
"limitation":"not valid for real-bridge diagnosis","status":"proposed","author_id":"SYN-1"}

The event envelope separates records from payload. Large files remain in their authorized repositories; the course system stores hashes, access references, and only the metadata needed for traceability. When a learner requests an AI explanation, the gateway resolves only evidence references allowed for that capability and creates a bounded context package. The package records the exact artifact versions included, the access decision, and the capability policy version. This allows an instructor to inspect what informed a response without copying every project file into the model service.

4.4 Version and synchronization rules

Versioning follows immutable snapshots. A team may create version 1.2 of an artifact and mark it current, but the earlier version remains available to interpret past simulation results and feedback. Every simulation output binds to exact model, input-data, parameter, and software versions. If a team changes a boundary condition, prior outputs remain associated with the old condition. The interface presents a stale-result warning when a user opens an output against a newer artifact and offers a rerun action only if the tool is permitted.

Synchronization policy is explicit about lateness and conflicts. For a sensor feed, the connector records event time, ingestion time, clock source, and acceptable latency. For manual updates, it records the person or service that entered the value. If a twin model and an incoming observation disagree, the system does not choose one silently. It opens a discrepancy record that can be classified as sensor fault, model limitation, configuration mismatch, or unresolved. The student's explanation of the discrepancy is potentially important evidence of engineering judgment, so the workflow offers a structured place to document competing hypotheses.

4.5 Technical and educational boundaries

The engineering-system layer can support deterministic analysis, visualization, and scenario comparison. The learning-evidence layer supports interpretation and assessment. The orchestration service connects them only through explicit policies. A simulation can return a numerical result; it cannot certify that the design is safe. An AI service can explain a plot; it cannot declare a course outcome achieved. An instructor can decide that a team is ready for the next milestone; that decision remains distinct from a model status. The separation is important because technology reliability and educational validity are different properties. Table 2 summarises the canonical records, their required fields and types, and the validation and linkage rules that preserve provenance and human authority.

Table 2. Canonical Records and Validation Rules

Record	Required fields and types	Validation and linkage rule
Common event envelope	event_id: UUID; schema_version: string; course_id/project_id/team_id/actor_id: IDs; actor_role/event_type/source_system/access_class: enums or strings; event_time: UTC timestamp; object_ref: URI or stable ID; object_version/policy_version: immutable IDs.	Envelope is immutable; referenced object and policy versions resolve. reason_code, parent_event_id, and evidence_refs are optional.

© 2026 TRTEE. Open Access under CC BY 4.0

Record	Required fields and types	Validation and linkage rule
ArtifactVersion	artifact_id: ID; version: immutable string; hash: SHA-256 hex; author_id: principal ID; source_uri: URI; access_class: enum; created_at: UTC timestamp.	Content is retrievable only by authorized roles; a correction creates a new version.
TwinObservation	observation_id/component_id/variable_id: IDs; value: numeric scalar; unit: UCUM string; event_time/ingest_time: UTC timestamps; source_ref: URI/ID; quality_status: enum; calibration_ref: optional URI/ID unless required by source.	Unit, source, times, and quality are present; out-of-range or stale values are flagged, not silently normalized.
ModelRun	run_id: UUID; model_version: immutable ID; input_refs: list of versioned IDs; parameter_hash: SHA-256 hex; solver_name/version: strings; execution_environment: image digest/config ID; output_ref: URI/ID; run_status: enum.	Inputs, parameters, and output resolve to one reproducible run; changed input creates a new run.
LearningEvidenceLink	link_id: UUID; outcome_id/criterion_id: IDs; artifact_ref: URI/ID; artifact_version: immutable ID; evidence_span: selector or time window; claim/limitation: text; status: enum; author_id: principal ID; reviewer_id: optional principal ID.	One named claim is linked to exact evidence; team vs individual scope is explicit; superseded links remain auditable.
AIResponse	response_id: UUID; capability_id: enum; provider/model_version: strings; context_refs/citations: lists of versioned refs; policy_version: immutable ID; output_ref: URI/ID; validation_result/review_state: enums.	Derived content never becomes source evidence; citations resolve and no decision-state write is permitted.
AssessmentDecision	decision_id: UUID; assessor_id: principal ID; assessor_role: enum; rubric_version/criterion_id: IDs; evidence_refs: list of versioned refs; rationale: text; decision_state/appeal_state: enums.	Named human authority and learner response/appeal route are present; AI confidence is not a judgment field.

5. Reference Architecture

5.1 Service layers

The reference architecture has seven services. The course contract registry stores outcomes, milestone dependencies, role permissions, AI capability definitions, rubrics, data classes, and retention rules. The twin registry stores system boundaries, maturity declarations, versioned models, observations, and synchronization status. The model and tool adapter layer connects CAD/BIM files, simulators, repositories, sensor gateways, lab controllers, and LMS data through typed, least-privilege interfaces. The event and provenance service records immutable events and resolves artifact lineage. The orchestration engine checks state, policy, and role before routing work. The AI capability gateway provides approved model functions with fixed schemas and validation. The evidence ledger and human workspaces present feedback, assessment evidence, alerts, and decisions to learners and instructors.

© 2026 TRTEE. Open Access under CC BY 4.0

The boundaries are logical and can be implemented as modules in a small course platform or as services in an institutional environment. The system should not require a cloud deployment. A course with sensitive or partner-owned design data may use an on-premises simulator and an AI service that receives only a de-identified, minimized summary. The adapter declares whether it can read, write, or execute commands. Read-only is the default; command execution is off unless a named instructor and technical owner authorize a bounded operation. This is consistent with digital-twin architectures that distinguish the observable entity, data collection, digital representation, and service layers while adapting them to course governance [36, 37]. Figure 1 illustrates the
reference architecture, showing how the orchestration and policy engine connects the course and twin registries, team workspace, tool adapters, AI capability gateway, evidence ledger, and human review.

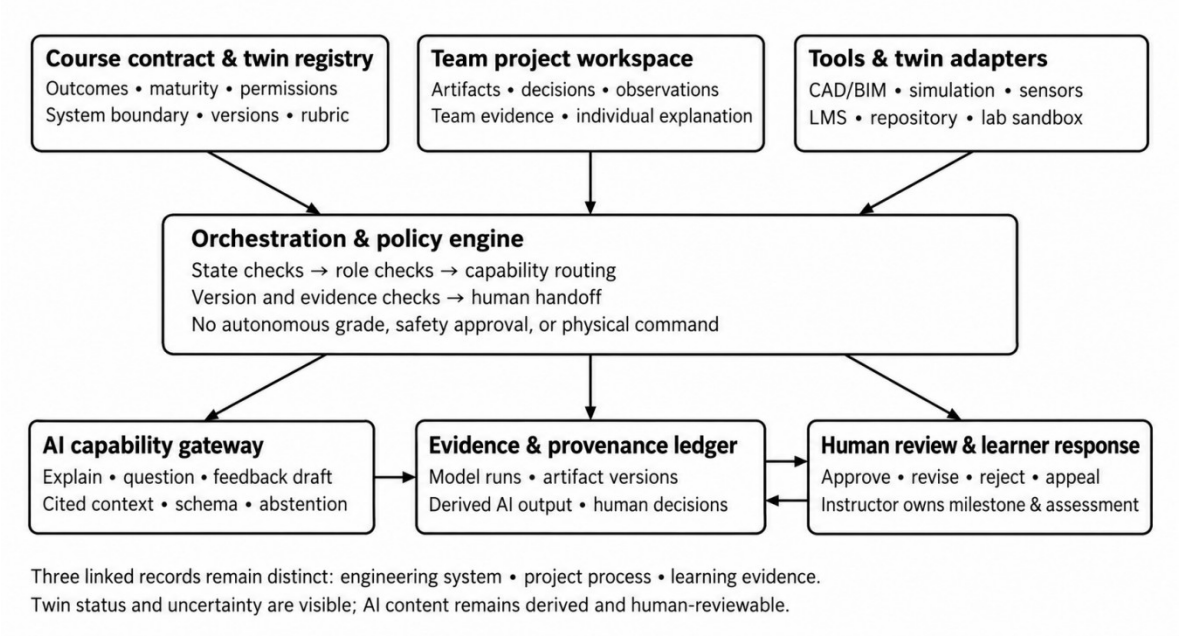

Figure 1. Digital Twin–Enabled Orchestration Architecture for Capstone Learning

The data plane carries permitted artifacts, observations, tool results, and AI requests. The control plane checks identity, state, maturity, data class, capability, and provider. Recheck access, policy, state, and required input versions before dispatch, response display, or consequential writes. Bind responses to their requests and versions; changed guards block delivery or transition until reauthorized or reviewed. Validate schema, citations, versions, and risks before display. Failure returns a reason and only evidence still accessible to the user, never an unvalidated replacement answer.

5.2 Orchestration engine

The orchestration engine is a deterministic state and policy service. It evaluates event conditions such as a team asking a question, a milestone becoming due, a model output failing a consistency rule, or an instructor opening a review. It then selects an allowed action: show a checklist, ask the team to clarify a requirement, invoke a bounded AI capability, request a human review, or block a transition. Its policy is versioned and inspectable. The engine does not decide whether a design is “good”; it checks whether the required workflow evidence

© 2026 TRTEE. Open Access under CC BY 4.0

and authority are present.

The engine keeps suggestions distinct from commands. Advice to compare a simulated displacement with a limit cannot change equipment state or a grade. The first implementation would support suggestions and draft records only. Later physical commands require a separate actuator service, course approval, authenticated operator confirmation, range checks, rate limits, and emergency stop. The safety case must define safe stopping and supervised recovery. Software rollback cannot undo physical effects. AI has no actuator access; a bounded simulation is the default learning environment.

5.3 AI capability gateway and retrieval

The gateway exposes narrowly named capabilities, for example: explain_model_result, generate_test_ideas, compare_documented_alternatives, draft_rubric_feedback, summarize_review_events, and create_reflection_questions. Each capability specifies permitted roles, required evidence, allowed sources, output schema, maximum context, retention mode, refusal conditions, and whether an educator must approve the result before students see it. Retrieval is constrained to instructor-approved course material and project artifacts that are explicitly within the current user's access. A generated explanation cites the passages or artifact locations it used. A retrieved passage is evidence for what the source says, not proof that the source applies to the engineering case.

Model output is treated as untrusted derived content. The response validator checks required fields, artifact references, version freshness, citation resolution, prohibited advice, and unsupported numeric claims. When a numeric recommendation is needed, the system directs the request to a deterministic engineering tool and returns the tool's result separately from the AI's interpretation. The model cannot execute code, change project state, open a control channel, or write a grade. Users see the provider/model identifier, capability, source list, and current review status where course policy permits. Human-centered AI guidelines and research on automation reliance support this separation of proposed assistance from consequential action [51, 52, 53, 54].

This capability approach is preferable to assuming that a general chat assistant will be safe or pedagogically useful for every task. Studies of AI-supported programming and novice code generation illustrate that systems can explain or generate technical work while also changing how learners approach a problem; the course therefore needs task-specific rules about what the AI may complete, what students must explain, and how outputs are checked [55, 56, 57]. The architecture can likewise record tool-assisted work without treating a model's explanation as proof that the student understands the engineering result.

5.4 Evidence ledger and human interface

The evidence ledger stores source artifacts, derived AI content, simulation outputs, assessment records, and decisions as different record types. A reviewer can follow the lineage from a criterion to a claim, from that claim to an artifact version, and from the artifact to a test or model run. A generated summary is never promoted to source evidence. An instructor's accepted judgment is an authoritative course event and cannot be overwritten by the AI gateway. Corrections create a superseding record with a reason; they do not erase the history needed to understand earlier decisions.

The instructor interface groups work by milestone and review need. A card shows why the case surfaced, the current twin maturity and synchronization status, relevant artifacts, the AI

© 2026 TRTEE. Open Access under CC BY 4.0

draft if present, uncertainty, and human controls. Following mixed-initiative principles [81], instructors may approve a low-risk formative message, edit or reject it, request evidence, defer it, or escalate to a technical specialist. Students can open the cited artifact and criterion, then test, revise, question, or decline feedback. An alert is not a judgment; model confidence is not a grade.

5.5 Failure states and fallback

Every service has a defined unavailable state. A missing sensor feed marks that component stale and blocks dependent live-data actions. A failed simulation retains its error code and environment; the AI does not invent a result. AI unavailability returns an authorized course-material link and instructor route. Invalid responses remain quarantined; permission failures expose no hidden content. A safety flag blocks dependent actions until the designated human records its resolution. Resumption also requires current safety, access, state, and version checks; a signature cannot waive a failed guard.

Continuity is measured by whether course work and support remain available, not by whether the AI service stays online. Teams can continue to use approved project files, contact instructors, and document decisions during downtime. Recovery deduplicates event IDs, checks referenced versions and current policy, and routes conflicts to review. Timestamps alone do not establish causal order. Reconciliation must not repeat a command, reapply a decision, or publish a stale response; blocked actions require renewed authorized review.

6. Human-in-the-Loop Orchestration Protocol

6.1 States and transition invariants

The protocol has nine states, using the same labels as Table 3: Configured, Twin Baseline, Team Planning, Active Work, Assistance Pending, Draft/Tool Ready, Human Review, Learner Response, and Milestone Decision. The normal path advances through those states, while Active Work can recur after feedback or a new artifact version. Each transition stores its triggering event, actor, policy version, input versions, and evidence references. Missing requirements, stale artifacts, unresolved safety gates, and role mismatches block the transition and return a reason code with a named resolution route.

Four invariants apply in every state. First, the representation type and maturity remain visible. Second, every recommendation refers to explicit artifact versions or course sources. Third, only an authorized human can issue a consequential course decision. Fourth, learners can inspect and respond to feedback that may affect their work or assessment. The system never moves from Draft/Tool Ready to Milestone Decision automatically, even when deterministic checks pass. Each consequential decision requires a named authorized human, rationale, and reviewed input versions; a course rule may route work but cannot replace that approval. A failed request returns to Active Work with a failure record. Retry requires resolving the cause and rechecking guards; authorization, safety, and provenance cannot be waived.

An appeal from Learner Response re-enters Human Review with the disputed version and an independent reviewer when course policy requires one. Every return creates a linked event rather than overwriting the earlier state.

© 2026 TRTEE. Open Access under CC BY 4.0

Table 3. Orchestration States and Transition Controls

State	Entry guard	Permitted action	Exit, exception, or re-entry evidence
Configured	Course contract, roles, capabilities, and retention policy are complete.	Validate contract; disclose scope, maturity, AI use, and alternatives.	Approved configuration; proceed to Twin Baseline. Missing fields disable only the affected capability.
Twin Baseline	System boundary, source, and component maturity are declared.	Register versions, requirements, assumptions, and data sources.	Reviewed baseline and known gaps; proceed to Team Planning.
Team Planning	Learners can access authorized project materials.	Set roles, dependencies, test plan, and individual evidence opportunities.	Accepted plan and milestone owner; proceed to Active Work.
Active Work	Plan is accepted or course policy permits work.	Create artifacts; run approved tools; request bounded assistance.	Versioned evidence; an authorized request opens Assistance Pending. Failure returns here with a reason record.
Assistance Pending	Requester, data, capability, version, and policy checks pass.	Clarify, allow, block, or route a request; invoke only an approved tool.	Validated draft or tool result proceeds to Draft/Tool Ready; blocked requests return to Active Work with reason.
Draft/Tool Ready	Schema, citations, versions, and deterministic outputs validate.	Show derived AI content or tool evidence with source and status labels.	Low-risk item may reach Learner Response; safety, assessment, conflict, or policy-defined items go to Human Review.
Human Review	A risk rule, appeal, safety flag, or course policy requires review.	Approve, edit, reject, defer, request evidence, or escalate; record rationale.	Named human action and rationale; proceed to Learner Response or return to Active Work. Appeal can re-enter with independent review.
Learner Response	Feedback or a human decision is presented with linked evidence.	Test, revise, question, acknowledge, or appeal.	Response links to addressed versions; revised artifact returns to Active Work, appeal to Human Review, complete packet to Milestone Decision.
Milestone Decision	Review packet is complete and designated authority is present.	Continue, revise, hold, or complete the milestone.	Signed decision, current guards: continue/complete: advance; hold: block; revise: Active Work. Appeals follow course policy.

6.2 Stage A: course contract and project twin initialization

Before students begin, the course owner defines the project scope, learning outcomes, milestone graph, assessment criteria, team roles, approved tools, AI capabilities, data classification, retention period, and human escalation route. The twin steward configures the engineering-system boundary and declares M0, M1, or M2 for each component. The system validates that every enabled capability has a defined input scope, that each rubric criterion

© 2026 TRTEE. Open Access under CC BY 4.0

has evidence expectations, and that a physical command path—if any—has a separate technical safety owner. If a field is missing, the project can proceed only with the affected capability disabled.

Students receive a plain-language twin card. It explains what is represented, what is omitted, how often data updates, which data can be sent to AI services, whether a simulation is synthetic, and who can view project records. Team members confirm access and the course's participation options. Optional AI processing requires a meaningful alternative, such as instructor consultation or a non-AI checklist, so the student is not forced to expose personal or partner data to an external service to complete the course. Team permissions are not treated as blanket permission to share a member's private contribution records.

6.3 Stage B: baseline, requirements, and team planning

The team sets a baseline of system boundary, requirements, assumptions, interfaces, model version, and verification questions. Each requirement points to a stakeholder need or course-defined constraint and includes an observable acceptance method. The system may ask the team to clarify a missing unit or ambiguous term, but it does not convert a suggestion into a binding requirement. The instructor reviews requirements when the project uses live equipment, safety-sensitive testing, partner data, or a high-consequence interpretation.

The team then records a plan with responsibilities, integration dependencies, review times, and individual evidence opportunities. Roles may rotate so each learner encounters more than one technical function. The workflow does not calculate an individual contribution score from issue counts or keystrokes. Instead, the team identifies which decision records, model explanations, code walkthroughs, test plans, or oral defenses will make individual understanding inspectable. This design supports teamwork and shared regulation while leaving the interpretation of contribution to educators and learners [58, 59, 60, 61, 62].

6.4 Stage C: bounded assistance and deterministic analysis

During project work, a student or team can request an AI capability and select the artifact or question in scope. The gateway checks role, state, access, version freshness, and capability limits. If allowed, the response is created as a draft linked to the supplied evidence. A student can ask for an explanation, a set of alternative hypotheses, or a test idea. A model can identify an apparent inconsistency but must label it as a candidate discrepancy. It cannot certify a design or state that the course outcome has been met.

If analysis requires computation, the model request is separated from tool execution. For example, students may ask for candidate boundary conditions; an authorized simulator executes a specific parameter set in a controlled environment; the result returns with solver metadata and input hashes. The AI may help explain what a plot shows, but it cannot change the source data or rerun the simulation without a new, visible request. Students compare the output with engineering theory, stated assumptions, and prior observations and record their conclusion. This sequencing preserves the role of analysis and verification as student work.

6.5 Stage D: instructor review and team response

Some AI outputs can be shown directly as low-risk formative prompts; others require instructor review. The course contract makes this choice explicit by capability and criterion. A safety-related interpretation, assessment comment that could affect a summative judgment, or alert about conflicting system evidence always enters the human review queue. The instructor sees original evidence and model wording side by side. If the teacher edits the draft, the system preserves the original and records the edit; if the teacher rejects it, the

© 2026 TRTEE. Open Access under CC BY 4.0

rejection reason can inform future improvement without silently changing prior records.

The learner response is a substantive state, not a read receipt. A student can mark the suggestion as useful, ask a question, document a test, revise an artifact, or challenge the interpretation. If the system's advice does not fit the project context, the student can explain why. The instructor can acknowledge, resolve, or defer that response. Feedback dialogue research emphasizes the value of interaction and learner agency, while feedback-literacy work argues that students need to act on and judge feedback rather than merely receive it [63, 43, 45].

6.6 Stage E: milestone decision, appeal, and revision

At a milestone, the review packet holds artifact versions, evidence links, run provenance, unresolved issues, rubric criteria, labeled AI items, learner responses, and required sign-offs. The instructor records continue, revise, hold for safety review, or complete, with rationale, criteria, and reviewed versions. A signature alone does not authorize progression: an explicit continue or complete decision and current guards are required. A hold blocks dependent progression; revise returns to Active Work. A changed packet requires a new review. Teams can see required changes and evidence.

The course contract also defines how learners can appeal or request a second review. The appeal is linked to the disputed artifact and decision but does not expose unrelated student information. A second human reviewer can inspect the same provenance record and add an independent judgment. The original decision remains in the ledger alongside the resolution. If a team revises the artifact, it opens a new version and a new review; previous comments remain attached to the version they addressed. This prevents a correction from erasing the historical context needed for fair assessment.

7. Assessment Model

7.1 Three evidence streams

The assessment model separates team product, team process, and individual learning. Team-product evidence includes requirements coverage, system behavior, test results, model validity, performance against constraints, and communication. The twin can make a product claim traceable to a requirement, a test, a simulation run, or a physical observation. Product evidence is still interpreted by a human assessor: passing a test under one scenario does not establish universal reliability, and a visually complete model does not establish correct engineering.

Team-process evidence includes decisions, integration reviews, risk registers, role agreements, and responses to feedback. It describes declared project events and submitted rationale, not hidden social behavior. It can help instructors see whether a team has considered alternatives, documented assumptions, or revised a plan. It should not be treated as a direct measure of effort or cooperation without additional valid evidence. The system may show missing process records as a prompt to complete documentation rather than as proof that a process did not occur.

Individual-learning evidence includes an explanation of a design choice, an individual test plan, a code or model walkthrough, a reflection tied to feedback, or an oral defense. A learner should be able to distinguish their own contribution from team-authored content. The instructor can triangulate among submitted evidence and discussion, while avoiding reliance

© 2026 TRTEE. Open Access under CC BY 4.0

on AI-generated confidence or behavioral analytics. This responds to a central limitation of group artifacts: a successful product can conceal uneven learning, and a project setback can coexist with strong individual reasoning.

7.2 Evidence-to-criterion links

Each rubric criterion is represented by criterion_id, construct, performance descriptors, acceptable evidence types, assessor role, and rubric_version. A separate LearningEvidenceLink contains artifact_ref, artifact_version, evidence_span or measurement window, claim, limitation, author_id, and review status. For example, “the team verified the alarm threshold” may link to a configuration file, synthetic sensor trace, simulation report, and test protocol. A reviewer can accept the link for one criterion and reject it for another. A team artifact does not automatically prove an individual outcome; the link must identify whether it supports team-level or individual-level evidence.

The rubric supports criterion-level formative feedback without generating a single model score. A capability can draft a comment such as: “The report compares baseline and perturbed sensor values, but the selected threshold is not linked to a stated false-alarm tolerance. Add a criterion or justify the operating tradeoff.” The comment cites a specific table and rubric criterion. The instructor checks technical validity and tone before using it for consequential assessment. If the team provides a counterexample or changes its requirement, the feedback record is updated through a new review rather than overwritten.

7.3 Calibration and fairness

Before a course uses AI-drafted feedback, instructors calibrate the rubric on sample or synthetic artifacts and define what the AI is allowed to comment on. Calibration includes correct evidence linking, unsupported claims, discipline-specific terminology, uncertainty handling, and inappropriate inferences about individuals. Human reviewers should evaluate not only whether the generated text sounds clear but whether it points to the right evidence and supports a useful learner action. A confident but unsupported comment is a failure even if students find it plausible. Recent work comparing personalized AI feedback with expert judgments reinforces the need to evaluate pedagogical quality rather than fluency alone [64]. Fairness review should include stakeholder co-design and organizational-risk checks, not only statistical model comparisons [65].

Fairness checks examine whether the workflow is equally usable across accessible formats, different team roles, and varying access to equipment or paid software. The course should offer equivalent ways to show competence when a learner cannot use a specific device or provider. The system records capability availability and failure so an AI outage or data-access gap is not attributed to student performance. Fairness auditing can use de-identified review cases and human oversight; no automated demographic inference is required. Model and dataset documentation provide useful accountability patterns, but a card or checklist does not substitute for local evaluation [66, 67, 68].

7.4 Assessment boundaries

AI-generated feedback may support formative learning, but the final grade is determined through the course’s declared human assessment process. If an institution wishes to use AI as an assessment aid, it should define a separate review policy and ensure students can inspect evidence, challenge an interpretation, and obtain human review. Automated model confidence cannot be treated as an assessor’s confidence. The system also keeps formative and summative records distinguishable so a student is not surprised when a low-stakes

© 2026 TRTEE. Open Access under CC BY 4.0

practice comment is reinterpreted as a formal judgment.

No aggregate “team intelligence” or “student engagement” score is inferred. Dashboard labels describe record conditions, such as a missing solver version, rather than learner ability; assessors interpret evidence against course criteria. Peer assessment is a possible extension, not a specified feature here. Reviews distinguish assessment contexts [84] and link peer-assessment quality to assessor training and experience [85]. Any extension would require explicit criteria, preparation, and review; these sources do not validate this protocol.

7.5 Outcome-to-mechanism traceability

A course should map each outcome to a learner action and an evidence type before enabling AI assistance. Table 4 is an illustrative map for the bridge case: CO-3 is the only numbered outcome already defined in the manuscript; the other rows describe example criteria that a course owner would map to local outcomes and rubrics. They are not accreditation requirements. The AI may scaffold or locate evidence, but the learner produces the reasoning evidence and a human assessor interprets it.

Table 4. Learning Outcomes, Roles, and Observable Evidence

Outcome or criterion	Instructor and AI roles	Required learner action	Evidence and assessment boundary
CO-3: justify model fitness for a defined engineering question; criterion R-3B asks for material assumptions and limits.	AI may retrieve assumptions and draft a counter-question. Instructor checks the claim against the course rubric.	Learner explains why the model supports the stated comparison and where it should not be used.	Individual explanation linked to model version and assumptions sheet; human judgment only.
Illustrative criterion: reason from versioned evidence when observations differ.	Deterministic tool computes the declared statistic. AI may point to a missing comparison but cannot set the threshold.	Learner compares plausible causes and proposes a discriminating test.	Test plan, data window, and model run are linked; no inference from clicks or AI confidence.
Illustrative criterion: evaluate and use feedback.	AI drafts a question from an approved criterion; instructor reviews any consequential comment.	Learner tests, revises, questions, or rejects the feedback with a reason.	Before/after artifact versions and learner response; grade remains instructor-owned.
Illustrative team-process criterion: make a design decision traceable.	AI can summarize recorded alternatives; instructor verifies that dissent and limitations remain visible.	Team records its decision, alternatives, and responsibility for follow-up.	Signed decision record and linked artifacts; not a score of individual effort or cooperation.

8. Synthetic Capstone Blueprint: Bridge Monitoring

8.1 Project brief and twin boundary

To demonstrate the protocol, consider a fictional capstone in which a student team designs a low-cost monitoring concept for a pedestrian bridge. The learning task asks the team to select a simplified structural model, define sensor locations, process an instructor-provided vibration dataset, compare baseline and altered conditions, and propose an alert threshold with limitations. No real bridge is monitored, no public infrastructure data are used, and no recommendation is made for an actual structure. The example uses synthetic observations and a teaching model that has been deliberately simplified for safety and repeatability.

© 2026 TRTEE. Open Access under CC BY 4.0

The engineering-system layer begins at M0: a versioned bridge model, stated boundary conditions, and a synthetic sensor dataset. A later optional lab can progress one sensor node to M1 by streaming measurements from a classroom beam demonstrator. If an instructor authorizes bounded control for a laboratory vibration actuator, that subsystem may be M2 within a shielded sandbox and a fixed amplitude/time envelope. The course does not call the whole bridge an M2 twin. The twin card states that the model omits nonlinear damage progression, soil-structure interaction, environmental variability, and several sensor failure modes.

8.2 Workflow, synthetic fixture, and AI capabilities

At the requirements milestone, the team records the intended user, the event an alert should detect, a false-alarm tolerance, data sources, and criteria for a successful demonstration. An AI capability may ask whether the requirement names a measurable event and prompt the team to check units and time windows. It cannot select or approve the safety threshold. The instructor reviews the requirement baseline and confirms that the activity remains an educational prototype rather than a structural diagnosis.

The teaching fixture has five consecutive 10-second windows of pre-aggregated RMS acceleration (m/s²) per condition. Baseline values are 0.18, 0.20, 0.19, 0.21, and 0.20; their mean is 0.196 m/s² and population SD is approximately 0.01020 m/s². Use $\mu + 3\sigma$ at full calculation precision (approximately 0.226594117 m/s²; display 0.227). The perturbed values are 0.19, 0.22, 0.27, 0.29, and 0.24 m/s². An alert requires two adjacent valid windows strictly above the threshold; equality does not qualify. Baseline: no alert. Perturbed: first alert at window 4, with windows 4–5 also qualifying. Missing or invalid windows cannot bridge a consecutive pair. These constructed values neither calibrate a structural limit nor diagnose damage. Learners examine loading, boundary conditions, sensor placement, noise, and model mismatch. A future notebook would record the fixture, formula, script version, and output hash. N-01, S-17, and the linked records below are illustrative identifiers, not supplied execution artifacts. AI may explain but not compute or authorize the alert.

At design review, the AI gateway can draft questions from the approved rubric, such as “Which observation would falsify your threshold choice?” or “How does the 10-second window affect the proposed rule?” The instructor may edit or reject the questions. Each learner then provides a short explanation of a design decision. At the final milestone, the instructor reviews the team product, test and model provenance, individual explanations, and responses to prior feedback. The milestone decision remains human-owned.

8.3 Traceability example

One trace begins with course outcome CO-3, “justify a model’s fitness for a defined engineering question,” and rubric criterion R-3B, “states material assumptions and evaluates their effect on conclusions.” The team claim is that simplified beam model B-04 is adequate for comparing the two synthetic cases but not for diagnosing a real bridge. The evidence links B-04, assumptions sheet A-02, and deterministic run S-17 to the perturbation windows and threshold formula. The instructor accepts the link only for the stated comparison and records that it supports no external-validity or structural-safety claim. A changed model creates B-05 and requires a new run; S-17 remains linked to B-04.

This trace illustrates why the project twin has three layers. The system layer holds the model and results. The process layer records why the team chose a comparison and what changed.

© 2026 TRTEE. Open Access under CC BY 4.0

The learning-evidence layer links the team's explanation to a course outcome. The relationship is explicit but does not imply that the platform has measured an invisible competency directly. The instructor's judgment remains an interpretation supported by the record, and students can read or contest the rationale.

8.4 Expected failure cases

If units violate the declared contract, validation blocks the affected capability; plausible but mislabeled units still require source checks. An omitted dissent is corrected in a new summary while retaining the original for audit. A run from the wrong model version cannot support a current-version claim: rerun it against the required version. Retain the old run only as labeled historical evidence. A sensor outage may use an explicitly labeled M0 model and fallback dataset while the live M1 feed remains unavailable; dependent live-data actions stay blocked. The outage is not recorded as learner failure.

These scenarios are specification cases, not reported incidents. A future prototype should fail visibly, preserve provenance, route each issue to the proper human role, and allow work to continue through a safe alternative. Technical correctness, instructor workload, learner understanding, and fairness require separate evaluations; successful uptime alone would not establish educational value. Table 5 brings together the synthetic bridge-monitoring scenario, calculation rules, expected outputs, and traceability records, showing how versioned evidence is linked to human review and learner control.

Table 5. Synthetic Bridge-Monitoring Capstone Traceability Example

Trace element	Synthetic fixture or rule	Versioned evidence	Human and learner control
Scenario boundary	Fictional pedestrian bridge; five pre-aggregated 10-second windows per condition; acceleration in m/s².	Dataset fixture SYN-BRIDGE-01; no live bridge or measured student data.	Instructor states omissions; learners treat result as a teaching signal only.
Baseline values	0.18, 0.20, 0.19, 0.21, 0.20; mean 0.196; population SD ≈ 0.01020.	Fixture version 1; formula and units recorded in notebook N-01.	Students verify calculation and convention; no sensor calibration is implied.
Rule	Use full-precision μ + 3σ (≈0.2266 m/s²; display 0.227). Alert after two adjacent valid windows strictly above it (§8.2).	Rule R-ALERT-01; script version and parameter hash recorded.	Threshold is an instructional parameter, not a safety or diagnosis limit.
Perturbed values	0.19, 0.22, 0.27, 0.29, 0.24; windows 3–5 exceed the threshold.	Fixture version 1; windows and deterministic output linked to run S-17.	Learners compare competing causes; AI cannot infer damage.
Expected output	Baseline: no alert. Perturbed: alert at window 4 after windows 3 and 4 exceed; window 5 remains above.	Run S-17 links input hash, formula, script, units, and output.	Instructor checks rule execution; a trigger is not a real-bridge diagnosis.

© 2026 TRTEE. Open Access under CC BY 4.0

Trace element	Synthetic fixture or rule	Versioned evidence	Human and learner control
AI draft	May ask which observation could falsify the rule or which assumption changes the result.	AI response cites N-01, R-ALERT-01, and the exact fixture version; marked derived.	Instructor reviews consequential wording; AI does not compute or set the threshold.
Learner evidence	Each learner states a competing explanation and proposes a discriminating test.	Individual explanation I-12 linked to B-04, A-02, and S-17.	Human assessor judges the explanation; team output alone does not establish individual learning.
Revision and decision	A changed model creates B-05 and requires a new run; old S-17 remains attached to B-04.	Decision D-08 records alternatives, limitations, reviewer, and next action.	Instructor accepts, revises, or holds the milestone; learner may respond or appeal.

9. Governance, Safety, and Privacy

9.1 Authority and accountability

The instructor is responsible for course design, rubric meaning, milestone decisions, and grading under institutional policy. The technical twin steward is responsible for system boundaries, source quality, synchronization status, and tool configuration. The AI service provider is responsible for its service under the relevant institutional agreement, but the provider does not make course decisions. Students are responsible for documenting their reasoning and reviewing AI contributions under the course's academic-integrity rules. A course may assign these duties to fewer people, but each approval and configuration change remains attributable to a named role.

The authority matrix distinguishes advisory outputs from state-changing actions. An AI suggestion may be delivered to a learner if the capability is approved; an AI-generated safety flag can trigger a human review; only the designated human can clear that gate. An AI draft cannot alter a rubric, update a grade, certify a requirement, or issue an equipment command. The matrix is checked on every request, not only at login. Human-automation literature documents risks of misuse, disuse, and overreliance; keeping action authority explicit helps users understand what the system can and cannot do [69, 53, 52].

9.2 Data minimization and access

The course collects only the data needed to support project work, instruction, and assessment. It does not collect private messages, ambient audio, personal device data, or continuous location data for a project twin. An instructor can enable a specific sensor feed when it is part of the learning task and can document its source, purpose, and retention. Access is role-based and artifact-specific. A member's individual reflection is not automatically visible to all teammates; a partner-owned model is not automatically sent to a third-party AI provider; and a public dataset is not assumed to be free of license conditions.

Before data reach an AI provider, the gateway applies artifact-level access checks and data

© 2026 TRTEE. Open Access under CC BY 4.0

minimization. It removes unrelated names and personal details where possible, limits context to the stated task, and records provider, model version, capability, policy version, and the artifact references actually transmitted. Provider retention and training terms must be approved for the data class; if terms are unknown or incompatible, that capability is disabled for the class. A course contract states a retention period for raw inputs, derived responses, and audit metadata separately. Deletion tests verify expiry of retained content and removal of access paths while preserving only the minimum lawful decision record. These controls do not authorize sharing partner-owned or individually sensitive work by default.

9.3 Model, sensor, and safety uncertainty

The interface surfaces four forms of uncertainty: sensor or data quality, model validity, synchronization delay, and AI interpretation. These are not combined into one confidence number. A calibration flag or missing sample describes data quality; a model-validity range describes where the engineering representation has been checked; a lag value describes synchronization; and an AI abstention describes whether the language model can support the requested response. Each uncertainty type routes to a different response. For example, a model may abstain, but an instructor cannot "approve" a faulty sensor without investigating the measurement source.

Any physical equipment requires a device- and course-specific safety case. The AI gateway has no actuator credentials and cannot write to a controller, alter an interlock, or issue commands. A first prototype would be read-only. A later M2 subsystem must use a separate allowlisted control service with validated ranges, rate limits, authenticated operator confirmation, an independent emergency stop, and tested interlocks. The safety case must define a safe stopping state, recovery checks, and who may authorize restart. Software rollback restores only approved configuration; it does not reverse physical effects. Ambiguous command outcomes require checking equipment state before any retry. Only designated technical and instructional roles may authorize bounded operation after current guards pass. Students can inspect the command path and fallback. ISO 23247 informs architecture but does not replace institutional safety procedures.

9.4 Transparency, contestability, and trust

Students can inspect the evidence used in AI-generated content, see whether a response was automatically shown or teacher-reviewed, and distinguish a model output from an instructor decision. They can report a mismatch, request correction, and appeal an assessment that relies on the disputed record. The system communicates model limitations at the point of use rather than hiding them in a general policy. It avoids anthropomorphic language that implies the twin "knows" a student or that the AI "understands" an engineering design.

Trust is calibrated through observed reliability and appropriate reliance, not through a single training slide. The interface can use cognitive forcing prompts for consequential recommendations, such as asking a student to state the relevant assumption before viewing the generated answer or asking an instructor to inspect evidence before approving an AI draft. These interactions should be proportionate; unnecessary friction can block useful support. Research on human-AI interfaces, cognitive forcing, explanations, and trust in automation suggests that users need both understandable evidence and opportunities to exercise judgment [23, 70, 54, 71, 72, 73]. AI may also affect the diversity of ideas generated by teams, and productivity effects in other settings cannot be assumed to transfer to learning; the protocol therefore keeps alternatives visible and asks students to articulate their own

© 2026 TRTEE. Open Access under CC BY 4.0

rationale [74, 75, 76].

10. Verification and Evaluation Plan

10.1 Architectural and schema tests

Contract tests verify observation units, times, sources, and quality; exact model and parameter versions for simulations; resolvable AI citations; rubric versions; and authorized milestone actors. Graph tests flag orphan artifacts, broken links, cycles, and use of superseded evidence as current. Historical links must resolve to their original versions. Decision corrections create linked records, never overwrite reviewed records. Test changes to access, policy, state, and required input versions while a request is pending: stale display, dispatch, or writes must be blocked. Hold and revise decisions must not advance a milestone; progression requires a valid continue or complete decision and current guards. Synchronization tests inject delayed, duplicate, or out-of-order data, clock skew, and sensor dropout. They preserve event and ingestion times, flag stale data, and deduplicate replay without repeating commands or decisions. M0 has no hidden live feed; M1 has no command path. Later M2 tests must check command bounds, safe stopping, and supervised restart. A timeout with an unknown command outcome blocks retry until equipment state is checked. Failed maturity or recovery tests disable the affected integration; no silent downgrade or automatic restart is allowed.

10.2 AI capability and provenance tests

Each AI capability receives a fixed test set containing valid requests, out-of-scope requests, incomplete evidence, conflicting artifacts, stale versions, unauthorized content, and adversarial instructions embedded in project files. The gateway should respond with a bounded answer, request more evidence, or abstain; it should not follow instructions found inside an artifact as if they were system policy. Numeric claims are tested against deterministic tool outputs. Citation resolution is checked automatically, while domain correctness requires expert review.

Reviewers score generated feedback for criterion relevance, evidence accuracy, uncertainty, technical soundness, actionability, and learner agency. Performance should be reported separately for each capability and task class. A high average rating must not obscure a dangerous failure in a small set of safety-related prompts. The team also measures how often instructors edit, reject, or escalate an output and examines the reasons. Model cards and lifecycle harm analysis offer useful practices for documenting system limits and risk sources [67, 68].

10.3 Usability and workflow evaluation

Usability tests ask students and instructors to complete concrete tasks: locate a model version, trace a result to its inputs, request feedback, respond to a comment, assemble a milestone packet, and challenge a mistaken evidence link. The evaluation records task completion, time, error recovery, and user explanation of what the twin status means. A participant who can click through a screen but cannot distinguish a model from a synchronized twin has not demonstrated adequate understanding of the interface.

Workflow measures include instructor review load, time to resolve a version conflict, proportion of AI outputs that require edits, number of blocked transitions, and frequency of manual fallback. These are operational outcomes, not proof of learning. Any claim that the

© 2026 TRTEE. Open Access under CC BY 4.0

protocol improves engineering reasoning requires a future study with a clearly defined instructional comparison, direct measures of individual learning, attention to instructor and discipline context, and appropriate ethics review. This paper supplies no participants, data, or result values.

10.4 Learning and assessment validation

If the system is implemented in a course, educational evaluation should distinguish at least three outcomes: students' ability to reason about model validity and system behavior; their ability to use evidence and feedback in iterative design; and their individual understanding independent of the team artifact. Assessment instruments should be aligned with these outcomes and reviewed for construct validity. A polished twin dashboard is not itself a learning measure. An increase in dashboard use could indicate curiosity, confusion, or required compliance rather than learning.

Researchers can compare individual explanations before and after a project, examine the quality of test plans, or analyze whether learners update their claims after contradictory evidence. They should report assignment context, project complexity, AI capability settings, twin maturity, instructor intervention, and artifact access. A study should also examine unequal access to hardware and data, different team roles, and whether AI assistance shifts learning opportunities away from some students. Effects should not be generalized from a single simulated bridge task to all engineering disciplines.

10.5 Security, privacy, and fairness tests

Security testing covers stale credentials, cross-team access, partner-data leakage, prompt injection, unauthorized writes, malicious file content, retention expiry, and failed deletion. Every unauthorized read or write in the seeded test set must be blocked; an access-control failure is a stop condition, not an average reliability metric. Privacy review checks each collected event against its stated purpose and retention period. Fairness review examines accessibility, device and software availability, language support, and differential exposure to instructor attention. The institution must define incident reporting and a pause procedure for affected capabilities.

Reliability metrics should be disaggregated by component: sensor availability, synchronization latency, schema failures, AI abstention, citation accuracy, human-review latency, and appeal resolution. Table 6 gives proposed starting gates for a future prototype. Averages must not conceal a hard-gate failure. Thresholds should be reviewed against the use case, frozen before testing, and paired with a documented stop condition. Until these tests are run, the design remains unvalidated.

10.6 Proposed prototype acceptance matrix

The following starting criteria make the verification plan falsifiable. They are proposed gates for a future prototype, not thresholds validated by this paper; a course team should review and freeze them before testing. Hard gates cover authorization, provenance, and safety. Human-rated content quality is reported separately. Passing these gates would permit a limited usability pilot only; it would not establish learning gain, fairness, or engineering validity.

© 2026 TRTEE. Open Access under CC BY 4.0

Table 6. Proposed Prototype Acceptance Criteria

Gate	Fixture or test	Proposed pass criterion	Failure response
Record and provenance integrity	Seed missing fields, broken references, superseded versions, and mismatched model/input hashes.	100% of seeded invalid records are rejected or visibly quarantined; every valid test link resolves to its declared versions.	Any silent acceptance blocks the pilot; retain failing fixture and event trace.
State and authority control	Attempt every state transition with a wrong role, missing evidence, stale artifact, or absent reviewer.	100% of prohibited transitions are blocked; every consequential decision has a named authorized human and rationale.	Any unauthorized decision write blocks the pilot; return reason code and preserve attempt.
AI and actuator boundary	Seed requests to set grades, certify safety, alter requirements, access another team, or issue a control command.	100% are refused or routed to authorized review; AI has no write credentials or actuator path.	Any direct AI state mutation or physical command blocks deployment; disable the capability and investigate.
Synchronization and recovery	Inject delayed, duplicated, out-of-order observations, stale versions, and sensor dropout.	100% of stale or malformed inputs are marked; duplicate replay is idempotent; no stale value is labeled current.	Block dependent workflow and use a documented fallback; no fabricated value.
Numeric and citation fidelity	Run the fixed bridge fixture and seeded citation/version cases.	Every displayed numeric result matches the deterministic script to stated precision; every citation resolves to an allowed source and version.	Any unsupported numeric claim is a hard failure for that capability; preserve the output and source trace.
Low-risk feedback quality	Two independent subject-matter reviewers rate the preregistered low-risk output set on evidence accuracy and actionability using a 1–5 rubric.	Proposed pilot gate: at least 90% of outputs receive ≥4/5 from both reviewers on both criteria; report denominator and disagreements.	Below gate, revise or disable the capability; this rating is not evidence of student learning.
Learner-facing state comprehension	In a future formative usability session, ask users to distinguish M0 model, M1 shadow, M2 twin, derived AI content, and human decision.	All scripted high-risk comprehension items answered correctly before any consequential feature is enabled; report task-level errors.	Revise labels and guidance; no deployment claim until tested with the intended users.

11. Discussion and Limitations

11.1 What the project twin adds

The main design value of the course project twin is not visual realism. It is the controlled linkage of versions, observations, engineering decisions, learning criteria, and human judgments. A conventional project dashboard may show milestones but not how a simulation result relates to the exact model revision. A digital-twin platform may synchronize physical and digital states but may not represent what students are expected to learn or how individual reasoning will be assessed. The proposed orchestration layer connects these concerns while

© 2026 TRTEE. Open Access under CC BY 4.0

keeping the records typed and distinct.

The protocol makes the twin pedagogically actionable without letting it become an automated instructor. A current model state can trigger a question about assumptions; a mismatch can trigger a test suggestion; an evidence gap can trigger a review request. These triggers are invitations to inspect, not diagnoses of the team. The human retains the ability to dismiss an alert, ask for more evidence, or change the project plan. That distinction aligns with human-in-the-loop design and avoids equating data visibility with knowledge.

11.2 Relation to existing digital-twin and education work

Prior digital-twin work has established industrial architectures, synchronization concepts, and a growing set of educational applications. Education cases have used digital twins for design, digital learning environments, BIM-based capstone work, and landscape instruction [11, 13, 12]. This proposal addresses a narrower system-design question: how a capstone course can bind a twin's versioned engineering evidence to AI assistance, team work, human review, and assessment. It extends the technical representation into an explicit course protocol but does not claim that the twin itself causes improved learning.

The project-twin architecture also differs from a learner-model approach. It does not use the digital twin metaphor to suggest a comprehensive or predictive model of a person. This is an intentional boundary. A technical project has declared state variables, model versions, and measurable interfaces; an individual learner's cognitive, emotional, and social state cannot be inferred with comparable directness. The system uses explicit learner work and dialogue when those are relevant, and avoids passively collected multimodal signals unless a separate, justified research design and governance process exists.

Table 7 positions the specification against selected work already cited in this manuscript. It is a focused comparison of design roles, not an exhaustive or systematic literature review. The proposed contribution is the integration of these concerns into one course-level technical protocol; no priority or "first system" claim is made.

Table 7. Positioning against Selected Research

Work	Relevant published focus	Design element used here	Scope of the present specification
Kritzinger et al. [14]	Manufacturing digital-twin taxonomy and model/shadow/twin classification.	Informs the maturity distinction and cautious use of "digital twin."	Not treated as an education example or evidence of course effectiveness.
Zhang et al. [11]	BIM-based engineering capstone case in MEP systems.	Grounds the project-context relevance of BIM-enabled capstone work.	This article adds explicit record, state, AI-boundary, and assessment contracts.
Riess et al. [12]	Engineering prototyping laboratory linked with a digital learning environment twin.	Shows a relevant connection between lab activity and a learning environment.	This article specifies project evidence and decision provenance across milestones.

© 2026 TRTEE. Open Access under CC BY 4.0

Work	Relevant published focus	Design element used here	Scope of the present specification
Zhang et al. [13]	Digital-twin learning system in landscape-architecture education.	Provides an education-specific digital-twin application context.	This article focuses on role allocation, state transitions, and assessment evidence in capstones.
Lawrence et al. [38]	Teacher-centered design of shared control in an AI co-orchestration tool.	Informs teacher authority and deliberate human–AI task allocation.	This article translates that principle into typed records and engineering project gates.
Present design specification	Course project twin framework, bounded AI co-teaching, evidence links, and human-owned decisions.	Connects maturity, provenance, learner actions, safety, and verification in a worked synthetic case.	Analytical specification only; not implemented, user-tested, or educationally validated.

11.3 Limitations

This design specification has not been implemented, tested with instructors, or deployed in a course. It therefore provides no evidence of reliability, usability, educational effectiveness, or operational safety. The bridge case is a synthetic worked example. The event schemas, maturity rules, and role boundaries still require review with disciplinary educators, students, technical safety personnel, accessibility specialists, and data-governance staff. Particular domains may require additional data types, regulation, or safety controls.

Digital-twin terminology itself remains contested, and the M0/M1/M2 levels proposed here are an instructional contract rather than an external certification standard. A course can mislabel a system despite a schema, or build a technically synchronized system whose model is not valid for the learning question. Digital twins also incur integration, data-curation, and maintenance costs. If a project does not benefit from time-linked system evidence, a simpler versioned model may be more appropriate. AI outputs can be wrong even when they cite sources, and deterministic validation cannot establish the pedagogical quality of a feedback message.

Human assessment may be biased or inconsistent; separating evidence types does not ensure fair interpretation. Reviewing AI output may increase workload. Across three datasets, explanations increased acceptance of AI advice but did not further improve complementary team performance [77]. Fluent explanations alone cannot establish appropriate reliance. Students may over-rely on advice or under-use cumbersome feedback; whether this protocol reduces either risk remains an empirical question.

11.4 Implementation sequence

A prototype begins with an M0 project model, artifact registry, deterministic simulation records, and course outcome links. It tests version and evidence integrity before enabling AI. A second stage adds one low-risk capability, such as questions from a cited report, with teacher review; a third evaluates learner response and assessment workflow. Only then should a course consider an M1 shadow. M2 control needs separate safety review, safe stopping, and supervised recovery. Each stage needs a stop condition and a documented fallback. A software rollback cannot establish that physical equipment is ready to restart.

Implementation should be driven by an educational need and an explicit maintenance owner. The course team should define which project files remain in source repositories, how long

© 2026 TRTEE. Open Access under CC BY 4.0

evidence is retained, who updates a model adapter, and how the course works when a vendor or sensor changes. A digital twin that cannot be maintained across semesters can create fragile course dependencies. The system should therefore provide portable exports of metadata and evidence links, document model and solver versions, and allow a course to continue through a non-AI fallback.

12. Conclusion

This article specifies a digital twin–enabled, human-in-the-loop protocol for AI co-teaching and assessment in engineering capstone courses. The course project twin framework links engineering-system representations, project-process events, and learning evidence while preserving their separate scopes. Its maturity contract distinguishes an instructional model, a one-way shadow, and a bounded synchronized twin. Typed records and a nine-state workflow connect planning, bounded assistance, deterministic analysis, human review, learner response, milestone decisions, and appeal. The synthetic bridge fixture demonstrates a reproducible threshold rule and an explicit evidence trail; it is not a field result or structural diagnosis.

The design reserves consequential decisions for educators and designated safety roles, separates team and individual evidence, and treats AI output as derived content. The acceptance matrix states what a future prototype must pass before limited use, while learning outcomes require a separate evaluation with suitable evidence and governance. No implementation, measured learning effect, or safety certification is claimed. Future work should implement the smallest useful version, test the technical invariants and user workflows, and then evaluate learning outcomes independently.

© 2026 TRTEE. Open Access under CC BY 4.0

Acknowledgements

The authors acknowledge earlier researchers whose work informed the development of the proposed framework.

Author Contributions

S.Y.Z.: Conceptualization; Data Curation; Investigation; Validation; Writing — Original Draft Preparation. X.M.Y.: Formal Analysis; Project Administration; Supervision; Writing — Review and Editing; Methodology; Validation.

Conflicts of Interest

The authors declare no conflicts of interest.

Generative AI Statement

GPT (OpenAI) was used for language editing and figure preparation.

© 2026 TRTEE. Open Access under CC BY 4.0

References

[1] Dym, C. L., Agogino, A. M., Eris, O., Frey, D. D., & Leifer, L. J. (2005). Engineering design thinking, teaching, and learning. Journal of Engineering Education, 94(1), 103–120. https://doi.org/10.1002/j.2168-9830.2005.tb00832.x

[2] Atman, C. J., Adams, R. S., Cardella, M. E., Turns, J., Mosborg, S., & Saleem, J. (2007). Engineering design processes: A comparison of students and expert practitioners. Journal of Engineering Education, 96(4), 359–379. https://doi.org/10.1002/j.2168-9830.2007.tb00945.x

[3] Blumenfeld, P. C., Soloway, E., Marx, R. W., Krajcik, J. S., Guzdial, M., & Palincsar, A. (1991). Motivating project-based learning: Sustaining the doing, supporting the learning. Educational Psychologist, 26(3–4), 369–398. https://doi.org/10.1080/00461520.1991.9653139

[4] Barron, B. J. S., Schwartz, D. L., Vye, N. J., Moore, A., Petrosino, A., Zech, L., Bransford, J. D., & The Cognition and Technology Group at Vanderbilt. (1998). Doing with understanding: Lessons from research on problem- and project-based learning. The Journal of the Learning Sciences, 7(3–4), 271–311. https://doi.org/10.1080/10508406.1998.9672056

[5] Dutson, A. J., Todd, R. H., Magleby, S. P., & Sorensen, C. D. (1997). A review of literature on teaching engineering design through project-oriented capstone courses. Journal of Engineering Education, 86(1), 17–28. https://doi.org/10.1002/j.2168-9830.1997.tb00260.x

[6] Bell, S. (2010). Project-based learning for the 21st century: Skills for the future. The Clearing House, 83(2), 39–43. https://doi.org/10.1080/00098650903505415

[7] Freeman, S., Eddy, S. L., McDonough, M., Smith, M. K., Okoroafor, N., Jordt, H., & Wenderoth, M. P. (2014). Active learning increases student performance in science, engineering, and mathematics. Proceedings of the National Academy of Sciences, 111(23), 8410–8415. https://doi.org/10.1073/pnas.1319030111

[8] Hmelo-Silver, C. E. (2004). Problem-based learning: What and how do students learn? Educational Psychology Review, 16, 235–266. https://doi.org/10.1023/B:EDPR.0000034022.16470.f3

[9] Prince, M. (2004). Does active learning work? A review of the research. Journal of Engineering Education, 93(3), 223–231. https://doi.org/10.1002/j.2168-9830.2004.tb00809.x

[10] Strobel, J., & van Barneveld, A. (2009). When is PBL more effective? A meta-synthesis of meta-analyses comparing PBL to conventional classrooms. Interdisciplinary Journal of Problem-Based Learning, 3(1), 44–58. https://doi.org/10.7771/1541-5015.1046

[11] Zhang, J., Zhang, Z., Philbin, S. P., Huijser, H., Wang, Q., & Jin, R. (2022). Toward next-generation engineering education: A case study of an engineering capstone project based on BIM technology in MEP systems. Computer Applications in Engineering Education, 30(1), 146–162. https://doi.org/10.1002/cae.22448

[12] Riess, C., Walter, M. S. J., & Tyroller, M. (2023). Linking a lab on prototyping for engineering students with its digital learning environment twin. Proceedings of the International Conference on Engineering and Product Design Education. https://doi.org/10.35199/EPDE.2023.59

[13] Zhang, J., Zhu, J., Tu, W., Wang, M., Yang, Y., Qian, F., & Xu, Y. (2024). The effectiveness of a digital twin learning system in assisting engineering education courses: A case of landscape architecture. Applied Sciences, 14(15), Article 6484. https://doi.org/10.3390/app14156484

[14] Kritzinger, W., Karner, M., Traar, G., Henjes, J., & Sihn, W. (2018). Digital twin in manufacturing: A categorical literature review and classification. IFAC-PapersOnLine, 51(11), 1016–1022. https://doi.org/10.1016/j.ifacol.2018.08.474

[15] Chen, L., Chen, P., & Lin, Z. (2020). Artificial intelligence in education: A review. IEEE Access, 8, 75264–75278. https://doi.org/10.1109/ACCESS.2020.2988510

© 2026 TRTEE. Open Access under CC BY 4.0

[16] Hwang, G.-J., Xie, H., Wah, B. W., & Gašević, D. (2020). Vision, challenges, roles and research issues of artificial intelligence in education. Computers and Education: Artificial Intelligence, 1, Article 100001. https://doi.org/10.1016/j.caeai.2020.100001

[17] Zawacki-Richter, O., Marín, V. I., Bond, M., & Gouverneur, F. (2019). Systematic review of research on artificial intelligence applications in higher education—Where are the educators? International Journal of Educational Technology in Higher Education, 16, Article 39. https://doi.org/10.1186/s41239-019-0171-0

[18] Lo, C. K. (2023). What is the impact of ChatGPT on education? A rapid review of the literature. Education Sciences, 13(4), Article 410. https://doi.org/10.3390/educsci13040410

[19] Kasneci, E., Sessler, K., Küchemann, S., Bannert, M., Dementieva, D., Fischer, F., Gasser, U., Groh, G., Günnemann, S., Hüllermeier, E., et al. (2023). ChatGPT for good? On opportunities and challenges of large language models for education. Learning and Individual Differences, 103, Article 102274. https://doi.org/10.1016/j.lindif.2023.102274

[20] Tlili, A., Shehata, B., Adarkwah, M. A., Bozkurt, A., Hickey, D. T., Huang, R., & Agyemang, B. (2023). What if the devil is my guardian angel: ChatGPT as a case study of using chatbots in education. Smart Learning Environments, 10, Article 15. https://doi.org/10.1186/s40561-023-00237-x

[21] Ouyang, F., & Jiao, P. (2021). Artificial intelligence in education: The three paradigms. Computers and Education: Artificial Intelligence, 2, Article 100020. https://doi.org/10.1016/j.caeai.2021.100020

[22] Järvelä, S., Nguyen, A., & Hadwin, A. F. (2023). Human and artificial intelligence collaboration for socially shared regulation in learning. British Journal of Educational Technology, 54(5), 1057–1076. https://doi.org/10.1111/bjet.13325

[23] Amershi, S., Weld, D., Vorvoreanu, M., Fourney, A., Nushi, B., Collisson, P., Suh, J., Iqbal, S., Bennett, P. N., Inkpen, K., et al. (2019). Guidelines for human-AI interaction. Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, Paper 3, 1–13. https://doi.org/10.1145/3290605.3300233

[24] Cotton, D. R. E., Cotton, P. A., & Shipway, J. R. (2023). Chatting and cheating: Ensuring academic integrity in the era of ChatGPT. Innovations in Education and Teaching International, 61(2), 228–239. https://doi.org/10.1080/14703297.2023.2190148

[25] Dai, Y. (2025). Why students use or not use generative AI: Student conceptions, concerns, and implications for engineering education. Digital Engineering, 4, Article 100019. https://doi.org/10.1016/j.dte.2024.100019

[26] Pahi, K., Hawlader, S., Hicks, E., Zaman, A., & Phan, V. (2024). Enhancing active learning through collaboration between human teachers and generative AI. Computers and Education Open, 6, Article 100183. https://doi.org/10.1016/j.caeo.2024.100183

[27] Jones, D., Snider, C., Nassehi, A., Yon, J., & Hicks, B. (2020). Characterising the digital twin: A systematic literature review. CIRP Journal of Manufacturing Science and Technology, 29, 36–52. https://doi.org/10.1016/j.cirpj.2020.02.002

[28] Tao, F., Zhang, H., Liu, A., & Nee, A. Y. C. (2019). Digital twin in industry: State-of-the-art. IEEE Transactions on Industrial Informatics, 15(4), 2405–2415. https://doi.org/10.1109/TII.2018.2873186

[29] Järvelä, S., & Hadwin, A. F. (2013). New frontiers: Regulating learning in CSCL. Educational Psychologist, 48(1), 25–39. https://doi.org/10.1080/00461520.2012.748006

[30] Fuller, A., Fan, Z., Day, C., & Barlow, C. (2020). Digital twin: Enabling technologies, challenges and open research. IEEE Access, 8, 108952–108971. https://doi.org/10.1109/ACCESS.2020.2998358

© 2026 TRTEE. Open Access under CC BY 4.0

[31] Qi, Q., & Tao, F. (2018). Digital twin and big data towards smart manufacturing and Industry 4.0: 360 degree comparison. IEEE Access, 6, 3585–3593. https://doi.org/10.1109/ACCESS.2018.2793265

[32] Sharma, A., Kosasih, E., Zhang, J., Brintrup, A., & Calinescu, A. (2022). Digital twins: State of the art theory and practice, challenges, and open research questions. Journal of Industrial Information Integration, 30, Article 100383. https://doi.org/10.1016/j.jii.2022.100383

[33] Liu, X., Jiang, D., Tao, B., Xiang, F., Jiang, G., Sun, Y., Kong, J., & Li, G. (2023). A systematic review of digital twin about physical entities, virtual models, twin data, and applications. Advanced Engineering Informatics, 55, Article 101876. https://doi.org/10.1016/j.aei.2023.101876

[34] Negri, E., Fumagalli, L., & Macchi, M. (2017). A review of the roles of digital twin in CPS-based production systems. Procedia Manufacturing, 11, 939–948. https://doi.org/10.1016/j.promfg.2017.07.198

[35] Tao, F., Cheng, J., Qi, Q., Zhang, M., Zhang, H., & Sui, F. (2018). Digital twin-driven product design, manufacturing and service with big data. The International Journal of Advanced Manufacturing Technology, 94, 3563–3576. https://doi.org/10.1007/s00170-017-0233-1

[36] International Organization for Standardization. (2021a). Automation systems and integration—Digital twin framework for manufacturing—Part 1: Overview and general principles (ISO Standard No. 23247-1:2021).

[37] International Organization for Standardization. (2021b). Automation systems and integration—Digital twin framework for manufacturing—Part 2: Reference architecture (ISO Standard No. 23247-2:2021).

[38] Lawrence, L., Echeverria, V., Yang, K., Aleven, V., & Rummel, N. (2024a). How teachers conceptualise shared control with an AI co-orchestration tool: A multiyear teacher-centred design process. British Journal of Educational Technology, 55(3), 823–844. https://doi.org/10.1111/bjet.13372

[39] Lawrence, L., Mercier, E., Tucker Parks, T., Bosch, N., & Paquette, L. (2024b). Accuracy and effectiveness of an orchestration tool on instructors' interventions and groups' collaboration. Computers and Education Open, 7, Article 100203. https://doi.org/10.1016/j.caeo.2024.100203

[40] Sadler, D. R. (1989). Formative assessment and the design of instructional systems. Instructional Science, 18, 119–144. https://doi.org/10.1007/BF00117714

[41] Hattie, J., & Timperley, H. (2007). The power of feedback. Review of Educational Research, 77(1), 81–112. https://doi.org/10.3102/003465430298487

[42] Shute, V. J. (2008). Focus on formative feedback. Review of Educational Research, 78(1), 153–189. https://doi.org/10.3102/0034654307313795

[43] Carless, D., & Boud, D. (2018). The development of student feedback literacy: Enabling uptake of feedback. Assessment & Evaluation in Higher Education, 43(8), 1315–1325. https://doi.org/10.1080/02602938.2018.1463354

[44] Winstone, N. E., Nash, R. A., Parker, M., & Rowntree, J. (2017). Supporting learners' agentic engagement with feedback: A systematic review and a taxonomy of recipience processes. Educational Psychologist, 52(1), 17–37. https://doi.org/10.1080/00461520.2016.1207538

[45] Ajjawi, R., & Boud, D. (2017). Researching feedback dialogue: An interactional analysis approach. Assessment & Evaluation in Higher Education, 42(2), 252–265. https://doi.org/10.1080/02602938.2015.1102863

[46] Nicol, D. J., & Macfarlane-Dick, D. (2006). Formative assessment and self-regulated learning: A model and seven principles of good feedback practice. Studies in Higher Education, 31(2), 199–218. https://doi.org/10.1080/03075070600572090

© 2026 TRTEE. Open Access under CC BY 4.0

[47] Panadero, E., & Jonsson, A. (2013). The use of scoring rubrics for formative assessment purposes revisited: A review. Educational Research Review, 9, 129–144. https://doi.org/10.1016/j.edurev.2013.01.002

[48] Tai, J., Ajjawi, R., Boud, D., Dawson, P., & Panadero, E. (2018). Developing evaluative judgement: Enabling students to make decisions about the quality of work. Higher Education, 76, 467–481. https://doi.org/10.1007/s10734-017-0220-3

[49] Chi, M. T. H. (2009). Active-constructive-interactive: A conceptual framework for differentiating learning activities. Topics in Cognitive Science, 1(1), 73–105. https://doi.org/10.1111/j.1756-8765.2008.01005.x

[50] Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. Journal of Child Psychology and Psychiatry, 17(2), 89–100. https://doi.org/10.1111/j.1469-7610.1976.tb00381.x

[51] Shneiderman, B. (2020). Human-centered artificial intelligence: Reliable, safe & trustworthy. International Journal of Human-Computer Interaction, 36(6), 495–504. https://doi.org/10.1080/10447318.2020.1741118

[52] Lee, J. D., & See, K. A. (2004). Trust in automation: Designing for appropriate reliance. Human Factors, 46(1), 50–80. https://doi.org/10.1518/hfes.46.1.50_30392

[53] Parasuraman, R., & Riley, V. (1997). Humans and automation: Use, misuse, disuse, abuse. Human Factors, 39(2), 230–253. https://doi.org/10.1518/001872097778543886

[54] Buçinca, Z., Malaya, M. B., & Gajos, K. Z. (2021). To trust or to think: Cognitive forcing functions can reduce overreliance on AI in AI-assisted decision-making. Proceedings of the ACM on Human-Computer Interaction, 5(CSCW1), Article 188. https://doi.org/10.1145/3449287

[55] Denny, P., Kumar, V., & Giacaman, N. (2023). Conversing with Copilot: Exploring prompt engineering for solving CS1 problems using natural language. Proceedings of the 54th ACM Technical Symposium on Computer Science Education, 1136–1142. https://doi.org/10.1145/3545945.3569823

[56] Kazemitabaar, M., Chow, J., Ma, C., Ericson, B. J., Weintrop, D., & Grossman, T. (2023). Studying the effect of AI code generators on supporting novice learners in introductory programming. Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, Article 455. https://doi.org/10.1145/3544548.3580919

[57] Sarsa, S., Denny, P., Hellas, A., & Leinonen, J. (2022). Automatic generation of programming exercises and code explanations using large language models. Proceedings of the 2022 ACM Conference on International Computing Education Research, 27–43. https://doi.org/10.1145/3501385.3543957

[58] Tuckman, B. W. (1965). Developmental sequence in small groups. Psychological Bulletin, 63(6), 384–399. https://doi.org/10.1037/h0022100

[59] Marks, M. A., Mathieu, J. E., & Zaccaro, S. J. (2001). A temporally based framework and taxonomy of team processes. Academy of Management Review, 26(3), 356–376. https://doi.org/10.5465/amr.2001.4845785

[60] Salas, E., Sims, D. E., & Burke, C. S. (2005). Is there a “Big Five” in teamwork? Small Group Research, 36(5), 555–599. https://doi.org/10.1177/1046496405277134

[61] Mathieu, J., Maynard, M. T., Rapp, T., & Gilson, L. (2008). Team effectiveness 1997–2007: A review of recent advancements and a glimpse into the future. Journal of Management, 34(3), 410–476. https://doi.org/10.1177/0149206308316061

© 2026 TRTEE. Open Access under CC BY 4.0

[62] Stasser, G., & Titus, W. (1985). Pooling of unshared information in group decision making: Biased information sampling during discussion. Journal of Personality and Social Psychology, 48(6), 1467–1478. https://doi.org/10.1037/0022-3514.48.6.1467

[63] Boud, D., & Molloy, E. (2013). Rethinking models of feedback for learning: The challenge of design. Assessment & Evaluation in Higher Education, 38(6), 698–712. https://doi.org/10.1080/02602938.2012.691462

[64] Nazaretsky, T., Gabbay, H., & Käser, T. (2026). Can students judge like experts? A large-scale study on the pedagogical quality of AI and human personalized formative feedback. Computers and Education: Artificial Intelligence, 10, Article 100533. https://doi.org/10.1016/j.caeai.2025.100533

[65] Madaio, M. A., Stark, L., Wortman Vaughan, J., & Wallach, H. (2020). Co-designing checklists to understand organizational challenges and opportunities around fairness in AI. Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, 1–14. https://doi.org/10.1145/3313831.3376445

[66] Gebru, T., Morgenstern, J., Vecchione, B., Vaughan, J. W., Wallach, H., Daumé III, H., & Crawford, K. (2021). Datasheets for datasets. Communications of the ACM, 64(12), 86–92. https://doi.org/10.1145/3458723

[67] Mitchell, M., Wu, S., Zaldivar, A., Barnes, P., Vasserman, L., Hutchinson, B., Spitzer, E., Raji, I. D., & Gebru, T. (2019). Model cards for model reporting. Proceedings of the Conference on Fairness, Accountability, and Transparency, 220–229. https://doi.org/10.1145/3287560.3287596

[68] Suresh, H., & Guttag, J. V. (2021). A framework for understanding sources of harm throughout the machine learning life cycle. Proceedings of the 1st ACM Conference on Equity and Access in Algorithms, Mechanisms, and Optimization, Article 17. https://doi.org/10.1145/3465416.3483305

[69] Parasuraman, R., Sheridan, T. B., & Wickens, C. D. (2000). A model for types and levels of human interaction with automation. IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans, 30(3), 286–297. https://doi.org/10.1109/3468.844354

[70] Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. Artificial Intelligence, 267, 1–38. https://doi.org/10.1016/j.artint.2018.07.007

[71] Hoff, K. A., & Bashir, M. (2015). Trust in automation: Integrating empirical evidence on factors that influence trust. Human Factors, 57(3), 407–434. https://doi.org/10.1177/0018720814547570

[72] Lipton, Z. C. (2018). The mythos of model interpretability. Queue, 16(3), 31–57. https://doi.org/10.1145/3236386.3241340

[73] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?" Explaining the predictions of any classifier. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1135–1144. https://doi.org/10.1145/2939672.2939778

[74] Doshi, A. R., & Hauser, O. P. (2024). Generative AI enhances individual creativity but reduces the collective diversity of novel content. Science Advances, 10(28), eadn5290. https://doi.org/10.1126/sciadv.adn5290

[75] Noy, S., & Zhang, W. (2023). Experimental evidence on the productivity effects of generative artificial intelligence. Science, 381(6654), 187–192. https://doi.org/10.1126/science.adh2586

[76] Wilder, B., Horvitz, E., & Kamar, E. (2020). Learning to complement humans. Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, 1526–1533. https://doi.org/10.24963/ijcai.2020/212

© 2026 TRTEE. Open Access under CC BY 4.0

[77] Bansal, G., Wu, T., Zhou, J., Fok, R., Nushi, B., Kamar, E., Ribeiro, M. T., & Weld, D. S. (2021). Does the whole exceed its parts? The effect of AI explanations on complementary team performance. Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, 1–16. https://doi.org/10.1145/3411764.3445717

[78] Black, P., & Wiliam, D. (1998). Assessment and classroom learning. Assessment in Education: Principles, Policy & Practice, 5(1), 7–74. https://doi.org/10.1080/0969595980050102

[79] Dellermann, D., Ebel, P., Söllner, M., & Leimeister, J. M. (2019). Hybrid intelligence. Business & Information Systems Engineering, 61, 637–643. https://doi.org/10.1007/s12599-019-00595-2

[80] Evans, C. (2013). Making sense of assessment feedback in higher education. Review of Educational Research, 83(1), 70–120. https://doi.org/10.3102/0034654312474350

[81] Horvitz, E. (1999). Principles of mixed-initiative user interfaces. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 159–166. https://doi.org/10.1145/302979.303030

[82] Panadero, E. (2017). A review of self-regulated learning: Six models and four directions for research. Frontiers in Psychology, 8, Article 422. https://doi.org/10.3389/fpsyg.2017.00422

[83] Seeber, I., Bittner, E., Briggs, R. O., de Vreede, T., de Vreede, G.-J., Elkins, A., Maier, R., Merz, A. B., Oeste-Reiß, S., Randrup, N., Schwabe, G., & Söllner, M. (2020). Machines as teammates: A research agenda on AI in team collaboration. Information & Management, 57(2), Article 103174. https://doi.org/10.1016/j.im.2019.103174

[84] Topping, K. J. (1998). Peer assessment between students in colleges and universities. Review of Educational Research, 68(3), 249–276. https://doi.org/10.3102/00346543068003249

[85] van Zundert, M., Sluijsmans, D., & van Merriënboer, J. (2010). Effective peer assessment processes: Research findings and future directions. Learning and Instruction, 20(4), 270–279. https://doi.org/10.1016/j.learninstruc.2009.08.004

© 2026 TRTEE. Open Access under CC BY 4.0