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

RESEARCH ARTICLE

A Design Specification for a Retrieval Augmented Curriculum Compiler for Translating Engineering Competency Standards into Executable Learning Tasks

Songyi Zhuo[1, *]
[1] City Graduate School, City University Malaysia, Petaling Jaya, 46100, Malaysia
[*] Correspondence: 1102109793@qq.com

Received 26 December 2025 · Accepted 6 May 2026 · Published 30 June 2026

Abstract

Engineering programs translate broad competency standards into course outcomes, learning tasks, and assessment evidence, often across disconnected documents. This article specifies a curriculum compiler that preserves the interpretive decisions and source versions connecting those objects. Three design questions address traceable representation, approval that remains valid after edits, and export with explicit information loss. The architecture combines eligibility-filtered hybrid retrieval, educator-confirmed mappings, a curriculum intermediate representation, deterministic checks, and separate pedagogical review. A restricted conformance profile defines object types, relationship signatures, approval binding, and diagnostic codes. A complete constructed statics task demonstrates partial contribution to an ABET student outcome without claiming program-level attainment. A local reference checker accepts one valid fixture and detects the intended violations in 22 invalid fixtures. A deliberately unsafe but structurally valid prompt also passes after simulated approval, demonstrating the boundary of structural assurance. A Common Cartridge 1.3 adapter produces a task resource whose manifest and internal references are checked locally. These results concern the supplied fixtures and adapter, rather than a deployed retrieval or teaching system. The contribution is an inspectable design contract connecting source applicability, curricular interpretation, assessment evidence, and controlled export. A staged evaluation protocol specifies retrieval baselines, expert adjudication, workload accounting, change-impact metrics, and the evidence required for subsequent learning claims.

Keywords: Curriculum Compiler; Retrieval Augmented Generation; Engineering Education; Competency Standards; Curriculum Mapping; Provenance

How to cite: Songyi Zhuo (2026). A Design Specification for a Retrieval Augmented Curriculum Compiler for Translating Engineering Competency Standards into Executable Learning Tasks. TRTEE, 2(2), 89–137.
https://doi.org/10.63646/PVTB2102

1 Introduction

Engineering education asks students to integrate technical knowledge, problem formulation, analysis, communication, and professional judgment [1], [2], [3]. Program standards express these expectations broadly enough to span institutions and course sequences. An instructor needs a more concrete object: a task with an observable student action, defined resources and constraints, and evidence that can be assessed. The translation involves interpretations at several levels. When these decisions are distributed across accreditation documents, curriculum maps, syllabi, assignments, and learning platforms, their justification and revision history can become difficult to inspect.

Curriculum mapping makes relationships among outcomes, courses, and assessments visible [4]. Constructive alignment connects intended outcomes with learning activities and assessment [5]. Neither approach implies that a relevant standards clause uniquely determines an assignment. Faculty must decide which part of a broad capability a course can address, what learners already know, and what evidence would justify a bounded claim about performance. A useful information system should preserve these decisions and their limitations, rather than replace them with an unexplained similarity score.

Retrieval augmented generation can condition generated text on selected external material [S1], [6]. In curriculum work, retrieval can locate candidate clauses and a language model can draft task wording. However, relevance, applicability, educational alignment, and readiness for publication are different properties. A retriever can rank an inapplicable proposed revision highly. A well-formed task can assess the wrong construct. An approved task can become invalid when its rubric changes. These distinct failure modes motivate explicit data contracts and approval conditions. Reviews and conceptual work on educational AI distinguish technological possibilities, educator roles, and evidence of educational effects [7], [8], [9].

The compiler term denotes staged transformation of validated curriculum objects into a structured task package. Its justification depends on inspectable types, transformation rules, diagnostics, and a defined export target. This article develops those elements as a design specification and implements a small reference checker and export adapter. The complete

© 2026 TRTEE. Open Access under CC BY 4.0

architecture, including document ingestion, dense retrieval, model drafting, institutional authentication, and live LMS deployment, remains a proposal. The local demonstration supplies evidence about selected structural rules; it does not establish retrieval effectiveness, educator acceptance, or learning gains.

Three questions organize the design. DQ1 asks how a representation can preserve an inspectable path from an applicable source clause to a local outcome, task, and assessment evidence. DQ2 asks how mapping decisions and task approval remain attributable to exact versions and become invalid when relevant content changes. DQ3 asks how an approved task can be exported while making unsupported platform features and remaining publication decisions explicit. The resulting contributions are a requirement rationale, a typed conformance profile, a governed transformation procedure, a fully specified constructed case, and a bounded executable demonstration with a separate evaluation protocol.

2 Educational foundations and related work

2.1 Levels of curricular interpretation

The design distinguishes source requirements, program competencies, course outcomes, task performances, and assessment evidence. A program may distribute a competency across several courses. A course can introduce or practice a component without certifying the entire capability. A task can generate useful evidence while leaving other facets of the source outcome untested. These distinctions are recorded explicitly so that the existence of a graph path does not become an attainment claim.

Action verbs and taxonomy labels alone do not determine task difficulty. Revised educational taxonomies distinguish knowledge and cognitive processes, but disciplinary context, uncertainty, prior knowledge, and expected independence still matter [S2]. Cognitive descriptors are therefore optional explanatory metadata in the general architecture. They must not be treated as machine-verifiable proof that an activity reaches an intended educational level. Task design also requires attention to cognitive engagement [10] and to cognitive load during complex learning [11], [12].

© 2026 TRTEE. Open Access under CC BY 4.0

2.2 Alignment and feedback

Constructive alignment motivates separate linked objects for outcomes, activities, evidence, and rubric criteria [5]. Their identifiers allow a system to detect missing declarations. Their meanings require professional judgment. For example, a rubric can be linked correctly to an outcome identifier while rewarding polished presentation instead of problem formulation. This design treats identifier consistency and assessment validity as separate review obligations [13], [14].

The curriculum intermediate representation (CIR) also records feedback and revision opportunities. Learners should be able to compare their work with criteria, interpret a discrepancy, and explain a subsequent revision [15], [16]. This connects recorded evidence with student action, consistent with formative assessment and self-regulation principles [17], [18], [19]. Backward design further motivates selecting intended performance and evidence before drafting attractive activities [S3]. Feedback design should also address learner uptake and participation in evaluative judgment, including peer review [20], [21], [22].

Engineering tasks can involve problem-centered inquiry, model selection, tools, and trade-offs [23], [24], [25]. Such features require appropriate scaffolding and disciplinary knowledge [26], [27]. The proposed compiler records the intended scaffolding, resources, and workload. It does not infer that an open-ended task is automatically more valuable, or that a verb such as analyze guarantees an appropriate level of challenge.

2.3 Retrieval and bounded generation

Dense passage retrieval supports semantic matching beyond exact wording, while lexical retrieval can preserve clause identifiers and defined terms [S4], [S5]. Neural ranking research provides a broader basis for comparing retrieval architectures [28]. Their relative value for competency standards is an evaluation question. Broad, repetitive outcomes can be difficult for either method to discriminate. This specification selects hybrid retrieval as a configurable design choice and defines a reproducible reference profile, without asserting that it outperforms simpler search.

© 2026 TRTEE. Open Access under CC BY 4.0

Retrieval does not eliminate unsupported generation or establish faithful attribution [29], [30]. A source link can be syntactically valid yet fail to support the accompanying sentence. The generator therefore receives limited drafting permissions, and its output passes through both structural checks and substantive review. Factual engineering claims require appropriate disciplinary support or verification. Labeling a sentence as an instructional choice does not exempt its factual content from review.

2.4 Existing representations and adjacent systems

CASE already models competency statements, associations, and rubrics, including criteria and performance levels [S6]. IMS Learning Design provides an earlier language for expressing learning designs and their runtime organization [S7]. These precedents mean that typed educational objects and deployable learning descriptions cannot by themselves establish novelty. CIR should reuse suitable external identifiers and supply explicit mappings to existing standards.

Recent systems also combine language models with governed curriculum work. Genesis treats LLMs as bounded collaborators in a human-led curriculum design process [S8]. PathBuilder combines retrieval grounding, assessment resources, and a model-based validation loop for personalized learning pathways [S9]. Their purposes differ from standards-to-task compilation, but they make human oversight and multi-stage generation insufficient as stand-alone novelty claims. Table 1 identifies the comparisons relevant to this proposal.

Table 1. Existing standards and related systems

Prior work	Relevant capability	Question addressed by CIR
CASE [S6]	Competency items, associations and rubrics	Reuse external identifiers; add applicability decisions and version-bound task approval.
IMS Learning Design [S7]	Representation of learning designs and runtime organization	Specify the upstream standards interpretation and its trace into task evidence.
Genesis [S8]	Human-governed LLM curriculum design	Make selected provenance and approval rules executable and inspectable.
PathBuilder [S9]	RAG and validation in personalized learning	Separate candidate retrieval, curricular mapping and structural assurance.

© 2026 TRTEE. Open Access under CC BY 4.0

Prior work	Relevant capability	Question addressed by CIR
Common Cartridge 1.3 [S10]	Interchange of learning resources	Declare adapter losses and remaining platform publication work.

The contribution proposed here is the joint treatment of clause applicability, explicit partial curricular contribution, version-bound approval, and an export loss record. This is an integration claim that remains open to comparison with simpler databases and workflow tools. The restricted implementation demonstrates that several stated constraints are executable. It does not establish that CIR is a superior representation or that a full compiler is necessary in every department. Educational AI research additionally situates explainability and accountable educator involvement within wider ethical and institutional questions [31], [32], [33].

2.5 Meaning of executable

An executable learning-task specification supplies fields that an adapter can transform into a concrete delivery resource. Execution in this article means materializing the declared task, resources, evidence instructions, and rubric in an inspectable package. Teaching, grading, learner adaptation, and accreditation decisions remain outside that definition. The implemented target is a Common Cartridge 1.3 web-content resource with assignment intent. Platform-specific assignment settings and authenticated publication are additional deployment responsibilities.

3 Design method and requirements

3.1 Design procedure and evidence boundary

The design proceeds through problem decomposition, literature-informed requirements, representation and workflow decisions, and inspection using a constructed engineering task. Table 2 connects each requirement to its motivating failure mode, design mechanism, and available verification. The requirements are a reasoned selection for the stated use case, rather than a claim of exhaustive stakeholder elicitation. No interviews, Delphi exercise, or expert consensus process was conducted.

© 2026 TRTEE. Open Access under CC BY 4.0

The target department supplies its local outcomes, course sequence, prerequisites, teaching constraints, source-use permissions, and adopted standards cycle. The general architecture supports multiple sources and tasks. To make a first implementation inspectable, the RACC-P1 profile restricts one build to one source clause, one program competency, one course outcome, and one task. Multiple evidence records and rubric criteria are permitted. A full system would need additional profiles for alternative mappings, several standards, multilingual sources, and curriculum-wide assessment.

The evidence reported here consists of an executed local schema and graph checker, explicitly constructed approval records, a task export, and numerical consistency checks for the case. Retrieval rankings used in the walkthrough are supplied illustrative inputs. No embedding model or language model was run to produce the demonstration. This distinction allows the local results to be interpreted without attributing unimplemented behavior to a complete system.

Table 2. Requirements derived from failure modes

Requirement and basis	Failure addressed	Mechanism and evidence
R1 Traceability Harden [4]	A task loses its source interpretation.	Typed source-to-task path; executed path and reference checks.
R2 Accountable interpretation Biggs [5]	Relevance is mistaken for alignment.	Confirmed mappings plus separate contribution level; structural fields checked, judgment pending.
R3 Evidence alignment Biggs [5]; Wiggins and McTighe [S3]	Activities and rubric assess different constructs.	Explicit evidence and criterion links; meaning requires educator review.
R4 Version integrity DQ2	An edit retains obsolete approval.	Frozen payload and digest comparison; stale approval fixture blocked.
R5 Bounded generation Ji et al. [29]	Generated text changes authoritative records.	Read-only source and approval boundary; proposed deployment control.
R6 Instructional constraints Merrill [26]	Task omits prerequisites, access or workload.	Declared fields, prerequisite graph and time bounds; selected checks executed.
R7 Authorized source use Source conditions; UNESCO [S11]	Text enters an unauthorized channel.	Separate manual and retrieved routes; incompatible access fixture blocked.
R8 Explicit stopping DQ1-DQ3	An unresolved conflict reaches export.	Errors block the adapter; human review and live publication remain deployment obligations.

© 2026 TRTEE. Open Access under CC BY 4.0

3.2 Acceptance obligations

The design requires traceability of every standards-derived mapping, a declared interpretation boundary, and accountable review. It requires the system to stop on unresolved structural errors and on unknown source-use permission. It also requires the publication interface to expose unresolved substantive concerns. These obligations are stronger than requiring every screen to display a green status: a missing reviewer judgment must not be converted into approval by a model recommendation.

The local reference checker implements only the machine-checkable subset specified in Section 4 and Appendix A. It assumes that its input records come from a trusted workflow boundary. In an institutional implementation, reviewer identity, role assignment, source rights, locator verification, and audit events must be established outside model-controlled data. The example approval label fixture-reviewer is a test value, not evidence that a teacher evaluated the task.

4 Architecture and formal contract

4.1 Components and review boundaries

The architecture comprises a source registry, ingestion service, retrieval service, competency graph, compiler, and educator workspace. A project context pins source editions, local outcomes, course assumptions, schema versions, and export settings. The registry records source authority and permitted use. Ingestion preserves document identity and clause locators. Retrieval proposes candidates. A mapping reviewer decides whether each candidate applies and how the local curriculum contributes to it.

The compiler consumes confirmed mappings and a structured teaching brief. It produces a draft CIR object, invokes drafting only in permitted fields, and runs structural checks. Pedagogical and engineering review follows. A final gate compares the current payload with the version that was approved. Figure 1 puts mapping confirmation and task approval on every relevant path and sends substantive edits back through validation.

© 2026 TRTEE. Open Access under CC BY 4.0

4.2 Source applicability and provenance

A source record contains issuing organization, title, canonical URI, publication year, adopted edition, effective period where stated, disciplinary scope, permission status, and a digest of the retained source bytes. Clause records add the structural locator, normative status, and extraction or verification record. Publication year, review cycle, and local adoption date are distinct fields. A newer edition does not automatically govern an existing curriculum project. Data lineage and provenance models provide precedents for retaining source identity and transformation history [34], [35].

Normative status is checked at clause level. An eligible document can contain definitions, explanatory material, current requirements, and proposed amendments. For example, the ABET 2025-2026 criteria include a section of proposed changes in the same publication as the applicable requirements [S12]. Indexing the entire file under a single effective label would lose that distinction. The registry must retain section context during segmentation and prevent a proposed amendment from supplying a current requirement unless an explicit project policy calls for prospective planning.

© 2026 TRTEE. Open Access under CC BY 4.0

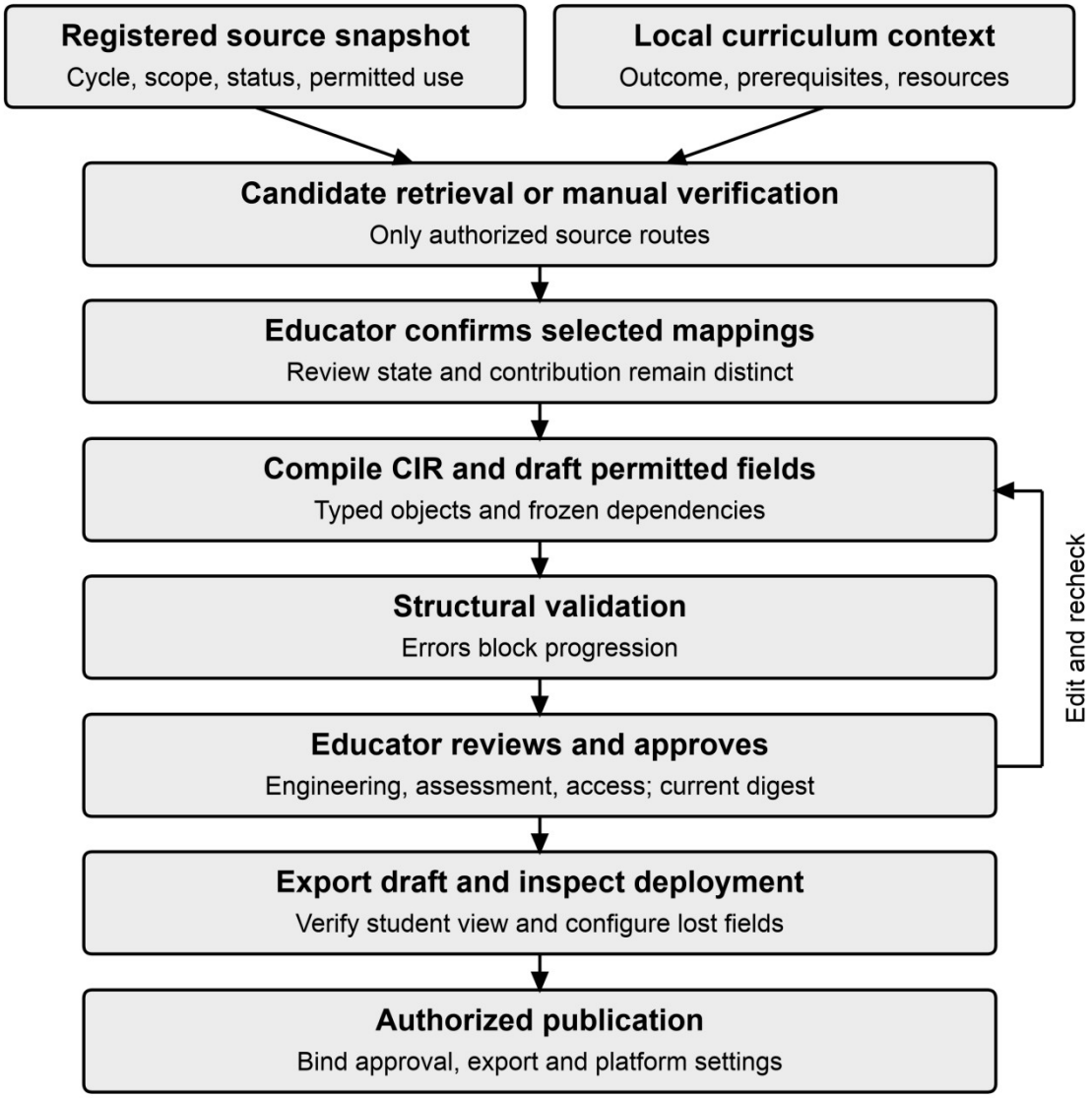


Figure 1. Governed Transformation and Publication Workflow

Superseded records remain available for historical explanation. A new build uses an explicitly adopted snapshot and a recorded applicability decision. A source steward may authorize a historical edition for a defined purpose, with scope and review date. This avoids silently equating newest with applicable. Unresolved edition identity, contradictory scope, and OCR uncertainty create a hold requiring source review.

The source digest verifies byte identity only against a retained snapshot. A digest does not establish publisher authenticity, permission, or the correctness of a locator. Those

© 2026 TRTEE. Open Access under CC BY 4.0

properties need trusted registration and inspection. The reference case records the downloaded ABET PDF digest and the inspected page location. Its locator binding detects later changes to that recorded pair; it does not autonomously understand the PDF or authenticate the publisher.

4.3 CIR objects and relationships

CIR separates instructional meaning from delivery format. Knowledge-graph research informs the use of explicit entities and typed relationships [36]. Table 3 summarizes the core objects. The complete executable schema generator appears in Appendix A and emits JSON Schema Draft 2020-12. Each object has a declared kind, stable identifier within the build, version, and descriptive content. Every required property is explicit; additional undeclared properties are rejected in RACC-P1. Future extensions require a new declared profile rather than silent acceptance by an old validator.

Table 3. CIR objects in the restricted P1 profile

Object	Required content	Cardinality or constraint
Node	id, kind, version, text	Unique id; positive integer version; one source, program, outcome and task.
Alignment edge	id, a, b, relation, review, contribution, reviewer, rationale	Unique id; legal endpoints; selected edges confirmed.
Source	node_id, edition, uri, locator, sha256, normative_status, access, mapping_route, verified_locator	One record; effective status and adopted edition; route compatible with permission.
Task	node_id, title, phases, resources, durations, accessibility, evidence_method, feedback, release	One record; nonempty fields; local duration bounds; draft export.
Rubric criterion	node_id, evidence_id, outcome_id, weight, levels	At least one; four levels; positive weights total 100.
Build	source digest, adopted edition, schema, validator, adapter, retrieval, model, prompt	Exact supported profile versions; explicit none for unused model.
Approval	reviewer, role, digest	Separate envelope; current payload digest; simulated identity in the capsule.

© 2026 TRTEE. Open Access under CC BY 4.0

The general design distinguishes an object's logical identity from its version identity. Within a frozen RACC-P1 payload, each identifier resolves to exactly one version. Edges use those identifiers and inherit the versions of their endpoints. Approval binds the entire payload, including node versions, edge rationale, source metadata, task text, and rubric. Across builds, dependency indexes use the pair of logical identifier and version.

Relationship signatures are directional. Table 4 specifies the allowed source and target kinds. A legal trace from task to source follows addresses, contributes_to, and derived_from. Evidence is connected through produces, and each rubric criterion uses assesses and criterion_for. The prerequisite subgraph alone is required to be acyclic; the presence of a cycle in some other relation is not automatically an error.

Table 4. Directional relationship signatures

Relation	Source kind to target kind	Purpose
derived_from	Program to source	Record the selected program interpretation.
contributes_to	Outcome to program	Record introduced, practiced or assessed contribution.
addresses	Task to outcome	Identify the local intended performance.
produces	Task to evidence	Declare the artifact or performance record.
assesses	Rubric to evidence	Locate evidence used by a criterion.
criterion_for	Rubric to outcome	Declare the construct being evaluated.
requires	Outcome or prerequisite to prerequisite	Declare dependency; this subgraph must be acyclic.

Mapping review state and instructional contribution are independent. The broader graph can retain proposed, confirmed, rejected or deferred edges; P1 requires every included edge to be confirmed. Contribution is introduced, practiced or assessed for contributes_to; other relations use na in the fixture. The checker rejects na for contributes_to but does not enforce na for other relations. Assessed denotes evidence collection, not learner mastery. Reviewers judge whether the declared contribution is defensible.

The P1 task must have an outcome path, at least one evidence record, and rubric links covering every declared evidence record. Each rubric criterion points to the same local

© 2026 TRTEE. Open Access under CC BY 4.0

outcome and an evidence record. Its weight must be positive, and all weights sum to 100. The supplied case uses four levels, scored 0 to 3. The schema requires exactly four textual levels, matching the scoring rule and adapter. A different scoring scale requires a new profile and explicit export support. Rubric research motivates reviewing the educational use and validity of the descriptors separately from their numerical consistency [13], [14], [37].

4.4 Validation classes and diagnostics

Deterministic checks test explicit representations: field types, reference resolution, permitted relationship signatures, version equality, approval binding, prerequisite cycles, weight totals, declared workload bounds, and adapter compatibility. Table 5 gives the diagnostic families. Passing them establishes consistency of those declarations. It does not establish that the evidence measures the intended construct or that the declared workload is realistic.

Table 5. Structural diagnostics and their limits

Code family	Blocked condition	Property not established
E_SCHEMA	Missing, extra or incorrectly typed property; unsupported rubric scale	Truth or completeness of text.
E_ID / E_REF / E_CARDINALITY	Duplicate ids, unresolved endpoints or unsupported core counts	Correctness of the selected interpretation.
E_TYPE / E_PATH / E_CYCLE	Illegal relation, broken source path or prerequisite cycle	Curricular adequacy or realistic prerequisites.
E_REVIEW / E_CONTRIBUTION	Unconfirmed edge or absent contribution declaration	Actual identity or quality of the reviewer.
E_SOURCE / E_EDITION / E_SOURCE_HASH / E_LOCATOR	Ineligible status or inconsistent source binding	Publisher authenticity or a newly asserted locator’s truth.
E_ACCESS	Blocked source or incompatible metadata-only route	Legal entitlement or enforcement of external permissions.
E_RUBRIC / E_EVIDENCE / E_WEIGHT	Broken assessment links or inconsistent weights	Assessment validity or reliability.
E_WORKLOAD / E_PROFILE / E_APPROVAL	Duration outside declared range, wrong profile or stale approval	Actual student time, adapter certification or authenticated approval.

© 2026 TRTEE. Open Access under CC BY 4.0

Heuristic checks may flag unsupported standards paraphrases, suspicious safety language, implausible prerequisites, or likely resource barriers. Their outputs require a reason and uncertainty label. They must be evaluated for false positives and false negatives. Educators separately judge engineering correctness, assessment validity, difficulty, accessibility, and the interpretation boundary. A model-based reviewer can assist this work, but its decision cannot be promoted into a deterministic type guarantee. Research on model explanations distinguishes explanation methods from the justification of consequential decisions; an intelligible explanation alone does not establish correctness [38], [39], [40].

For example, a node containing an unsafe request to prove a real structure safe can satisfy every structural rule. The executed counterexample in Section 8 demonstrates this limitation. A practical workflow should present such concerns to a qualified reviewer and require a resolved disposition before approval. The reference checker deliberately leaves this semantic responsibility visible.

5 Retrieval and compilation protocol

5.1 Reference retrieval profile

The proposed retrieval profile uses the local outcome text as its primary query, supplemented by discipline and learner-stage descriptors. Clause identifiers are preserved in the lexical channel. Sources are filtered by adopted edition, scope, normative status, and permission before retrieval. A document parser retains heading ancestry; a long clause is divided into spans of at most 300 tokenizer tokens with 50-token overlap within that clause. No span crosses an edition or normative-status boundary. These are reference configuration choices requiring sensitivity analysis, not experimentally selected optima. Graph-based retrieval is a distinct research direction [41]; it is not implemented in the P1 reference checker.

The sparse channel uses BM25 with $k1 = 1.2$ and $b = 0.75$, Unicode normalization and case folding, while retaining clause numbers. A dense implementation must record the embedding model identifier, immutable revision or file digest, tokenizer, truncation policy, and similarity function before it is eligible to run. This article does not name an unexecuted

© 2026 TRTEE. Open Access under CC BY 4.0

model as a validated component. Its reference checker sets the model field to none. A later comparison should freeze one concrete dense model and retain its generated vectors.

Each channel retrieves up to 20 eligible parent clauses. Span scores are aggregated by the best-ranked span for a parent clause before fusion, so repeated overlapping spans cannot create multiple votes. Reciprocal rank fusion combines the two ranks with a constant of 60 [S13]. A clause absent from a channel receives zero contribution from that channel. Ties are resolved by the stable clause identifier. The initial profile has no learned reranker; adding one creates a new profile and requires separate evaluation.

$$RRF(d) = \frac{1}{60 + r_{sparse}(d)} + \frac{1}{60 + r_{dense}(d)}$$

The interface displays up to 10 fused candidates, their source locators and neighboring context, and separate reject, defer, no-match, and applicability-conflict actions. Rank values support candidate ordering and are not interpreted as calibrated confidence. A retrieval trace preserves the query, filters, eligible universe, channel ranks, fusion configuration, and reviewer disposition. Table 6 illustrates the arithmetic using stipulated ranks; it is not a measured search experiment.

Table 6. Illustrative rank fusion inputs and output

Candidate	Sparse rank	Dense rank	Fused value
C1	2	1	0.032522
C2	1	4	0.032018
C3	4	2	0.031754
C4	3	Absent	0.015873
Proposed clause	Excluded	Excluded	Not ranked

The actual ABET walkthrough uses the metadata-only manual route. Its source text is not indexed or transmitted to a model. This provides a complete alternative when text use is restricted, while leaving the proposed automated retrieval path clearly separate from the executed demonstration.

© 2026 TRTEE. Open Access under CC BY 4.0

5.2 Mapping and constrained drafting

The reviewer sees the source clause, local outcome, contribution level, evidence proposal, and an explicit list of uncovered facets. A mapping record includes the reviewer, endpoint versions, relation type, rationale, and unresolved objections. Acceptance of one edge does not approve the full task. Alternative mappings can remain in the broader graph, but a P1 build selects a single confirmed path so that its validation result has an unambiguous scope. Research on teachers' AI-related professional knowledge supports retaining educator responsibility for such judgments [42].

A drafting brief contains the accepted outcome, learner prerequisites, disciplinary context, permitted tools, workload bounds, expected evidence, and constraints. The model may propose scenarios, activity phases, reflection questions, and feedback prompts. Source metadata, outcome identity, mapping confirmation, approval status, and publication controls are read-only. If a draft proposes a change to those fields, the system returns a change request to the responsible reviewer rather than modifying them during generation.

The trace distinguishes source requirements, local curriculum decisions, constructed numerical data, and engineering knowledge used in the task. Standards links support the source requirements. Local design choices carry their rationale. Factual calculations and disciplinary claims require an appropriate verification record. Presence of a source_link alone does not show that a standard supports a specific numerical assumption or instructional decision.

5.3 Transformation and failure behavior

Compilation freezes approved inputs, resolves permitted source material or the manual verification route, records mapping decisions, constructs CIR, validates it, and prepares a review package. Unresolved structural errors block approval and export. Warnings carry an owner and disposition. If a mandatory substantive review has not occurred, the task remains pending even when structural validation succeeds. Table 7 identifies the stage outputs and conditions needed to continue.

© 2026 TRTEE. Open Access under CC BY 4.0

Table 7. Compilation stages and progression conditions

Stage	Output	Condition for progression
Register	Pinned sources and course context	Authority, applicability and permitted route recorded.
Locate	Eligible candidates or manual locator record	No unresolved edition, extraction or access conflict.
Map	Selected interpretation edges	Named confirmation, contribution and boundary notes.
Compile	CIR task draft	Permitted fields only; generator has no approval access.
Validate	Errors and review package	No blocking structural errors.
Review and approve	Version-bound decision	Substantive review completed; mandatory concerns resolved.
Export and deploy	Draft package and platform preview	Losses addressed; learner view checked before publication.

Generation is an optional transformation inside this procedure. The reference fixture is human-authored and therefore remains reproducible without a model service. If generation is enabled later, each attempt must preserve its brief, permitted context, output, model configuration, and disposition. Retrying a model after failure creates another draft attempt, not an invisible replacement for the original. Evaluation must include unsuccessful and abstained attempts in its denominators. Prompting and language-model evaluation surveys inform the recording of prompt choices and the evaluation of complete attempt histories [43], [44].

6 Constructed engineering case

6.1 Source and interpretation boundary

The course and students in this example are fictional. The source is the real ABET Engineering Accreditation Commission criteria for the 2025-2026 review cycle, published with a 2024 copyright date. The selected location is Criterion 3, Student Outcome (1), on printed page 6, PDF page 8 [S12]. The institutional program competency and course outcome below are constructed interpretations rather than ABET wording.

© 2026 TRTEE. Open Access under CC BY 4.0

The program competency concerns formulating engineering problems using explicit models. The local outcome asks a second-year learner to explain assumptions, calculate forces in an ideal planar truss, compare independent checks, and state the limitations of the result. Prior learning comprises planar equilibrium and elementary spreadsheet formulas. The relationship is practiced. This example intentionally addresses a foundation for a broad outcome, without claiming that a determinate triangle truss demonstrates the full complexity or independence expected at graduation. Engineering education research treats conceptual understanding and the development of expertise as related but distinct educational concerns [45], [46].

Table 8 decomposes the contribution. The complex-problem component remains untested. A later task could add uncertain loading, competing model choices, or conflicting design constraints, but would require new scaffolding and evidence. Attaching the present task to the broad outcome does not authorize filling those missing facets by inference. Research on active learning motivates student reasoning and participation [47], [48]; it does not validate the learning effects of this constructed task.

Table 8. Partial contribution to the broad engineering outcome

Outcome facet	Evidence in this task	Boundary
Identify a problem	Student states the question and system boundary.	The scenario and numerical inputs are supplied.
Formulate a model	Assumptions, supports and free bodies.	The basic truss idealization is scaffolded.
Solve using engineering principles	Two equilibrium calculations with units.	Linear ideal statics only.
Interpret a computational check	Joint residuals and explanation of discrepancies.	Agreement does not validate the physical assumptions.
Address complex engineering problems	No complete evidence in this exercise.	Complexity and independent program-level attainment remain untested.

6.2 Complete student task

Task title: Diagnose a simplified roof-truss load path. Consider an ideal pin-jointed planar truss with nodes A at (0, 0), B at (6, 0), and C at (3, 4), in metres. Members are AB, AC, and BC. A is a pin support with horizontal and vertical reactions. B is a roller providing a

© 2026 TRTEE. Open Access under CC BY 4.0

vertical reaction. Member self-weight is neglected, loads act at joints, and the members transmit axial force only. AC and BC are each 5 m long; AB is 6 m. The model contains no material strength, section area, buckling, deflection, connection, or design-code checks.

At C, load case 1 has a 12 kN downward force and no horizontal force. Load case 2 has an 18 kN downward force and a 3 kN force to the right. Figure 2 supplies a geometric view; the coordinates and descriptions provide the same essential information in text. Use positive member force for tension, positive horizontal reaction to the right, and positive vertical reaction upward.

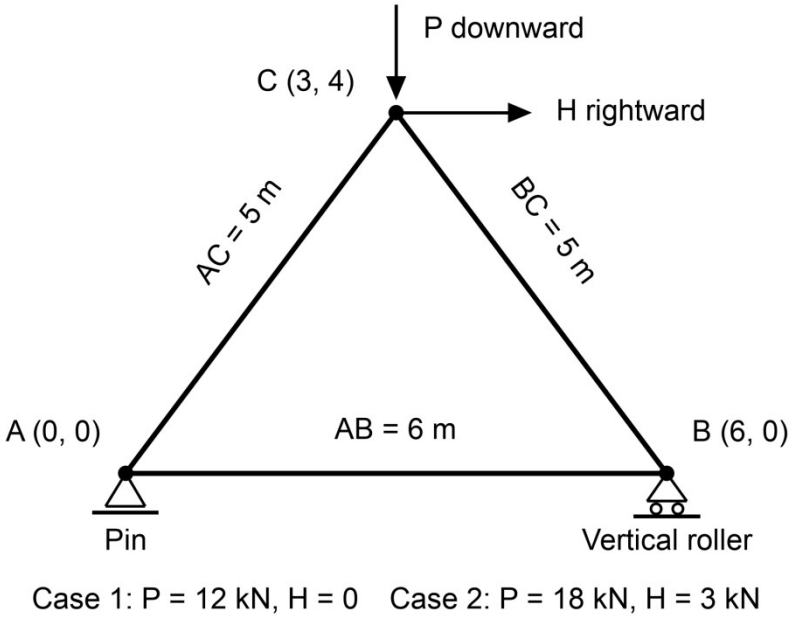


Figure 2. Ideal Planar Truss Model

Formulate the analysis question and distinguish the physical roof from the idealized model. List the assumptions and explain which observations would be needed to assess them. Draw the whole-truss and joint free bodies, calculate all support reactions and member forces for both cases, and show units and sign conventions. Implement the joint equilibrium residuals in a local spreadsheet or an equivalent calculation table. Explain any discrepancy before revising a calculation. End with a short account of what the model establishes and which additional information a real design decision would require.

© 2026 TRTEE. Open Access under CC BY 4.0

The activity allocates 15 minutes to framing and assumptions, 25 minutes to drawing and calculation, and 20 minutes to comparison and discussion. Sixty minutes of homework completes the portfolio and a reasoned revision. The declared total is 120 minutes within a local permitted interval of 90-150 minutes. These are design assumptions for the case, not measured completion times. Students may use structured text or an oral explanation instead of drawing where an equivalent record of forces, assumptions, and reasoning is retained.

6.3 Calculation and assessment evidence

Let P denote downward load magnitude and H the rightward horizontal load, both in kN. The whole-truss moment balance about A includes the horizontal load moment at height 4 m. For the stated sign convention, the reactions and member forces follow the expressions below. These calculations support the constructed task and do not assess a real structure.

$$A_x = -H; \quad A_y = \frac{P}{2} - \frac{2H}{3}; \quad B_y = \frac{P}{2} + \frac{2H}{3}$$

$$AC = -\frac{5P}{8} + \frac{5H}{6}; \quad BC = -\frac{5P}{8} - \frac{5H}{6}; \quad AB = \frac{3P}{8} + \frac{H}{2}$$

Table 9. Reference forces for the two constructed load cases

Quantity in kN	Case 1	Case 2	Meaning
P; H	12; 0	18; 3	Applied downward and rightward magnitudes.
Ax	0	-3	Horizontal support reaction.
Ay	6	7	Vertical reaction at A.
By	6	11	Vertical reaction at B.
AC	-7.50	-8.75	Compression under the positive-tension convention.
BC	-7.50	-13.75	Compression under the positive-tension convention.
AB	4.50	8.25	Tension.

The spreadsheet uses six residuals: two force balances at each of A, B, and C. At A they are Ax + AB + 0.6AC and Ay + 0.8AC. At B they are -AB - 0.6BC and By + 0.8BC. At C they are -0.6AC + 0.6BC + H and -0.8AC - 0.8BC - P. A student should show those residuals, their units, and the input cell references. Numerical agreement tests arithmetic consistency

© 2026 TRTEE. Open Access under CC BY 4.0

within this model. It does not verify the assumed joints, loads, supports, or omitted failure modes.

The submitted evidence is a single portfolio containing a problem statement, force representations, assumptions, two calculations, spreadsheet residuals, a limitation statement, and an explanation of one revision. Four rubric criteria link to that evidence and the local outcome. Table 10 gives observable level descriptions. The weighted score is the sum of each criterion's weight multiplied by its level divided by 3. This score expresses performance on the constructed task only.

Table 10. Analytic rubric for the constructed task

Criterion and weight	Level descriptors from 0 to 3
Formulation and assumptions 25%	0 Absent. 1 Question or assumptions unclear. 2 Question and most assumptions justified. 3 Question, boundaries and all key assumptions explicit and justified.
Equilibrium and units 35%	0 Absent. 1 Major equilibrium or unit errors. 2 Mostly correct, with minor local errors. 3 Both cases correct with units and signs.
Check and interpretation 20%	0 Absent. 1 Reports agreement only. 2 Computes residuals with limited interpretation. 3 Explains residuals, discrepancies and model limits.
Limitations and communication 20%	0 Absent. 1 Unqualified safety claim or unclear report. 2 States some limits clearly. 3 Clear report identifies omitted behavior and additional evidence needed for design.

6.4 CIR and diagnostic trace

The P1 fixture uses node identifiers s, p, c, t, e, q, and r1-r4 for the source, program competency, course outcome, task, evidence portfolio, prerequisite, and rubric criteria. All are version 1 in this build. Thirteen edges declare the source path, evidence path, prerequisite, and rubric relationships. Appendix A constructs the complete object, so no hidden database or model output is needed to reproduce it.

© 2026 TRTEE. Open Access under CC BY 4.0

The source record contains the inspected URI, cycle and locator, the retained PDF digest, effective normative status, metadata_only access, and manual mapping route. A simulated reviewer confirms the mapping with a partial-practice rationale. The task and rubric are then materialized and the payload is hashed. A simulated educator approval binds that digest. The first local check reports no structural error; the adapter consequently produces the package described in Section 7.5.

Changing the adopted edition produces E_EDITION. Changing the recorded locator without renewing its verification binding produces E_LOCATOR. Changing a task after approval produces E_APPROVAL. These are executed fixture behaviors. By contrast, the unsafe wording counterexample in Section 8 requires substantive judgment and is deliberately not claimed as a detected structural error.

7 Governance change management and export

7.1 Roles and approval transitions

A standards steward registers source authority, scope and permissions. A program committee or delegated reviewer confirms program mappings. An instructor owns the task, evidence and rubric. A platform administrator controls adapter configuration and deployment permissions. Institutions may combine roles, but the audit record must preserve which authority was exercised for each decision. The model has no reviewer identity and no publish capability. Explicit responsibility and oversight also respond to recurring concerns in AI ethics guidelines and algorithmic accountability [49], [50].

The conceptual workflow progresses through draft, structurally validated, substantively reviewed, approved, exported draft, deployment preview verified, and published. Approval records the reviewer, role, decision time, payload digest, and resolved warning dispositions. Publication additionally binds the exported artifact digest and the inspected platform configuration. Export and publication are separate actions: the reference adapter writes files locally and has no authenticated route to a live course.

An edit to task text, rubric, evidence, source applicability, mapping rationale, or other approved payload content invalidates that approval and returns the task to review. A final

© 2026 TRTEE. Open Access under CC BY 4.0

server-side comparison must be atomic with release so that another edit cannot occur between checking and publishing. The reference checker demonstrates digest mismatch detection only. It does not implement institutional identity, cryptographic signatures, concurrent editing, or transaction isolation. Those are deployment requirements rather than completed results.

7.2 Change classification and impact analysis

Source updates create new immutable records. A byte-level difference first triggers source inspection. The steward classifies it as a locator or formatting change, a substantive requirement change, a change in applicability, or an unresolved change. Identical wording in a newer edition is insufficient to declare unchanged applicability. A clause split or merge is represented by explicit predecessor-successor links, permitting one-to-many and many-to-one relationships without reusing a version identifier.

For a changed dependency set C, the impact queue contains objects reachable through the dependency relation from any member of C. Dependency edges are stored in the direction dependency to dependent. A traversal maintains a visited set and includes each affected object once. With a stored adjacency list, this inspection is linear in the reachable objects and links. It identifies candidates for review; it does not decide that every candidate needs a new pedagogical interpretation.

A source meaning change normally reopens dependent mappings and tasks. A corrected locator can retain a mapping judgment after the steward verifies its unchanged meaning, but the new locator and approval record still require a new version. A rubric change reopens assessment review and task approval. An index or embedding update changes future retrieval proposals; it does not by itself erase a previously justified human mapping. Rebuilding a task with the new index creates a distinct build record.

Reproducibility has two meanings here. Artifact replay recovers the exact stored CIR and exported bytes. Regeneration reruns transformations and may produce a different model draft. The build manifest records input snapshots, profile versions, prompts, model configuration and outputs, so differences can be explained. The local fixture uses deterministic data and fixed ZIP metadata to make its exported artifact reproducible. This

© 2026 TRTEE. Open Access under CC BY 4.0

Python JSON serialization is a local profile, not a claim of cross-language canonicalization for unrestricted JSON.

7.3 Restricted sources and information security

Source-use decisions precede retrieval and prompting. The workflow distinguishes permission to retain metadata, index text, send excerpts to a model, and redistribute content. When only metadata is available, an authorized reviewer opens the source independently, verifies the locator, records the interpretation and permitted paraphrase, and supplies a confirmed mapping. Compilation can then proceed from the local outcome and teaching brief. If use conditions remain unknown, the source is blocked. The ABET demonstration follows the metadata-only route and does not redistribute the standards PDF in the exported task.

Retrieved documents are untrusted data. Delimiters, an allow-list of tools, and schema-constrained output reduce some routes for indirect prompt injection but do not prove immunity. The threat model includes forged source metadata, instructions embedded in passages, and attempts to change approval fields through generated text. Trusted registration and approval records must reside outside the generator's write scope. The P1 checker assumes that trust boundary and does not test its implementation.

The design requires no student submissions or grades for initial task compilation. Any extension using learner data needs an institutional purpose, restricted access, retention rules and review. UNESCO guidance motivates preserving human agency in educational uses of generative AI [S11]. AI component documentation can draw on model cards, datasheets and risk-management practices, while recording what those documents do and do not substantiate [51], [S14], [S15].

7.4 Accessibility and equitable participation

The task representation records alternative formats, equipment assumptions, language demands, evidence alternatives and feedback opportunities. Missing declarations are detectable. Their adequacy requires review with the actual learner context. An accessibility field containing text cannot establish accessibility conformance. Equivalent evidence must preserve the assessed construct rather than merely offer a more convenient submission

© 2026 TRTEE. Open Access under CC BY 4.0

format. Any later learner-data-based adaptation would additionally need evaluation for algorithmic bias and fairness [52].

The case avoids a paid-tool dependency, gives coordinates in text, and permits an equivalent calculation table. Assistive-technology behavior, alternative assessment arrangements, and workload for specific learners remain untested. Personal accommodation records should remain within authorized institutional systems. The exported task contains general access provisions only. Research on digital media in higher education further motivates examining actual student and teacher usage rather than assuming that tool availability establishes effective participation [53].

7.5 Common Cartridge adapter and information loss

The implemented CC13-web-1.0 adapter targets Common Cartridge 1.3 as a portable interchange format. It creates imsmanifest.xml and webcontent/task.html. The manifest identifies one web-content resource with assignment intent and a navigation item pointing to it. The student-facing HTML contains the prompt, sequence, resources, evidence instructions, access provisions, feedback and rubric. Common Cartridge provides a packaging mechanism; assignment delivery settings still depend on the receiving platform [S10].

Table 11 states what is retained, transformed or left for deployment. Native gradebook criteria, due dates, submission attempts and conditional release are not created by this adapter. Approval records and the source PDF remain in the compiler record rather than the learner resource. The reference output therefore has a declared information-loss boundary and must not be described as a fully configured LMS assignment.

Table 11. Export mapping and explicit information losses

CIR content	Reference export	Remaining deployment work
Prompt, phases and resources	Escaped HTML text in task.html	Inspect appearance and resource suitability.
Evidence and rubric	Portfolio instructions, four-level descriptors and weights	Configure native submissions and gradebook if required.
Task title	HTML title, heading, manifest item and LOM title	Check title behavior in the receiving platform.

© 2026 TRTEE. Open Access under CC BY 4.0

CIR content	Reference export	Remaining deployment work
Accessibility and feedback	Human-readable provisions	Verify actual assistive-technology behavior and feedback process.
Source path and approval	Preserved in compiler-side fixture.json	Do not infer access protection from cartridge metadata.
Dates, attempts and release rules	Not implemented by this adapter	Set and verify in the LMS before publishing.

The exported manifest was checked against the published Common Cartridge 1.3 content-package XSD and its referenced schema files. Internal file references were resolved against the ZIP contents. The student task was rendered locally for inspection. These checks do not constitute full Common Cartridge certification or a live-platform import test. Before use, an administrator must record the exact LMS product and version, import into an unpublished course area, configure the unsupported settings, and verify the learner view and access behavior. Any platform edits must return to CIR or form a separately tracked local version.

8 Executed verification and evaluation protocol

8.1 Local reference execution

The Appendix A capsule was executed with Python 3.12.14 and jsonschema 4.25.1. It builds the Draft 2020-12 schema, constructs the single-task fixture, applies selected invalid mutations, runs the reference validator, exports the valid fixture, resolves its package references, and checks the statics example. No live model, retrieval corpus benchmark, teacher participant, or student participant was involved.

One valid fixture was accepted. Twenty-two invalid fixtures were blocked with their intended diagnostic present. Table 12 reports all cases, including extra diagnostics where one mutation breaks several dependent constraints. This is a selected regression set whose expected behavior was specified from the rules. It is not a random sample of deployment errors and provides no estimate of operational accuracy or general false-acceptance probability.

© 2026 TRTEE. Open Access under CC BY 4.0

Table 12. Executed fixture checks

Case	Mutation or control	Expected	Observed
01	valid	PASS	PASS
02	missing field	E_SCHEMA	E_SCHEMA
03	wrong number type	E_SCHEMA	E_SCHEMA
04	unknown field	E_SCHEMA	E_SCHEMA
05	duplicate node	E_ID	E_ID E_PATH E_REF
06	dangling edge	E_REF	E_PATH E_REF
07	wrong edge type	E_TYPE	E_PATH E_TYPE
08	unconfirmed edge	E_REVIEW	E_REVIEW
09	missing contribution	E_CONTRIBUTION	E_CONTRIBUTION
10	prerequisite cycle	E_CYCLE	E_CYCLE
11	proposed clause	E_SOURCE	E_SOURCE
12	wrong edition	E_EDITION	E_EDITION
13	source digest change	E_SOURCE_HASH	E_LOCATOR E_SOURCE_HASH
14	locator change	E_LOCATOR	E_LOCATOR
15	blocked source	E_ACCESS	E_ACCESS
16	unauthorized retrieval	E_ACCESS	E_ACCESS
17	broken evidence	E_EVIDENCE	E_EVIDENCE E_RUBRIC
18	bad weight	E_WEIGHT	E_WEIGHT
19	bad rubric scale	E_SCHEMA	E_SCHEMA
20	workload exceeded	E_WORKLOAD	E_WORKLOAD
21	wrong profile	E_PROFILE	E_PROFILE
22	edit after approval	E_APPROVAL	E_APPROVAL
23	unsupported role label	E_SCHEMA	E_SCHEMA

© 2026 TRTEE. Open Access under CC BY 4.0

The semantic counterexample asks for proof that a real truss is safe and passes structural checks after simulated approval. This does not establish engineering adequacy. Case 23 rejects the role label model; it does not authenticate reviewers or test authorization. The capsule name unauthorized-approver denotes this label test. Deployment requires authenticated reviewers, enforced authority and qualified substantive review, whose quality needs separate evaluation.

For each of the two numerical load cases, the six joint force residuals have a maximum absolute value of zero in the reported floating-point calculation. The assertion tolerance is 1e-10 kN. The check supports the numerical fixture within its stated idealization. The independently performed manifest XSD check and internal-reference check support the packaging result. These local checks do not validate source extraction, semantic mapping, accessibility, teaching effectiveness, or institutional security.

8.2 Retrieval and mapping evaluation

A subsequent retrieval study should assemble a permission-cleared corpus and queries representing exact identifiers, paraphrased outcomes, discipline terms, ambiguous requirements, and genuine no-match cases. Relevance labels should be separate from source eligibility labels. Two qualified reviewers should label independently, with disagreements adjudicated and ambiguous items retained as a separate category. A source-family split should keep near-duplicate editions and their paraphrases together when they would otherwise leak into tuning and testing.

The comparison should hold the eligible corpus and query set fixed across BM25, dense-only and fused retrieval. Candidate budgets and any tuning budget should be comparable. Report the distribution of relevant clauses per query, Recall@k and a ranking metric such as normalized discounted cumulative gain [54], and source-eligibility violations separately. A no-match query has no relevant set and should be evaluated through abstention and false-suggestion behavior rather than assigned an arbitrary recall value.

Mapping evaluation should use source clauses and course documentation, not similarity scores alone. Reviewers should judge applicability, relationship type, contribution level, and interpretation boundaries. Where practical, conceal the production method of the

© 2026 TRTEE. Open Access under CC BY 4.0

candidate mapping, while acknowledging that wording may reveal it. Agreement measures characterize consistency; adjudicated judgments and error explanations provide the reference assessment. None of these judgments alone demonstrates learner attainment.

8.3 Task quality workload and change impact

Task evaluation should compare educator-authored tasks, structured database-assisted tasks, and compiler-assisted tasks under the same brief. If the study tests the added value of generation, include a retrieval-and-template condition without generation. If it tests structural validation, compare the same drafts with and without those checks. This separates the effects of retrieval, drafting, and workflow controls instead of attributing every difference to the full architecture.

A predefined rubric should cover disciplinary correctness, alignment, evidence observability, prerequisite fit, workload plausibility, accessibility and clarity. Reviewers should provide criterion-level judgments and reasons, and the analysis should preserve course and institution groupings. Report uncertainty at the task or course unit actually sampled. Repeated variants of one task should not be treated as independent courses or learners.

Workload accounting includes source registration, initial mapping, draft review, correction, export configuration and later updates. Report setup and recurring effort separately. Time saved in drafting can be offset by additional verification. A within-instructor comparison should balance task order and account for familiarity and carry-over. Satisfaction and preference are useful process measures but cannot substitute for measured learning. Perceived usefulness and ease of use [55] and departmental adoption of educational innovations [56] should therefore be evaluated alongside, and separately from, learning outcomes.

For change-impact evaluation, create independently specified edits and adjudicate which objects require review. Include wording-only changes, clause splits, applicability changes, local outcome revisions, rubric changes and index updates. Evaluate both missed affected objects and unnecessary review requests. Table 13 defines denominators that prevent rejection of all tasks or indiscriminate re-review from appearing successful.

© 2026 TRTEE. Open Access under CC BY 4.0

Table 13. Evaluation measures and declared denominators

Measure	Definition	Interpretation requirement
Recall at k	Relevant eligible clauses in top k divided by all relevant eligible clauses for a matchable query.	Report k, query distribution and aggregation method.
Eligibility violation rate	Returned ineligible candidates divided by all returned candidates.	Report separately from relevance.
False acceptance	Invalid test objects accepted divided by all independently classified invalid test objects.	Report the error taxonomy and sampling method.
False blocking	Valid test objects blocked divided by all independently classified valid test objects.	Do not omit valid-but-difficult inputs.
Completion and abstention	Completed or abstained requests divided by all attempted requests, respectively.	Report errors and abandoned attempts separately.
Impact recall and precision	Correctly flagged affected objects divided by all affected objects, and by all flagged objects.	Use an independently adjudicated affected-object set.
Total workflow effort	Registration, mapping, review, correction, export and maintenance time per stated unit.	Separate setup from recurring effort and report failed attempts.

8.4 Deployment security and learning studies

Deployment testing must include untrusted source instructions, malformed inputs, unsupported model output, role spoofing, stale approval, concurrent edits, and attempts to publish an unreviewed task. Tests should distinguish a parser failure, a blocked action, and a semantic error that survives the pipeline. Adapter studies should report field losses and student-view differences for named LMS versions, including the treatment of post-import edits.

Learning claims require a separately defined educational intervention, suitable comparison, valid measures, learner context, and appropriate consent and privacy arrangements. A task-level structural pass rate cannot be used as evidence that students learn more. An initial pilot may instead examine whether the task elicits the intended reasoning and whether learners can use feedback to revise it. Any causal claim about competence would require a research design capable of supporting that inference.

© 2026 TRTEE. Open Access under CC BY 4.0

9 Discussion and limitations

9.1 What the representation contributes

The specification makes several otherwise implicit decisions inspectable. It distinguishes an applicable source from a merely relevant passage, mapping acceptance from the extent of curricular contribution, and task approval from platform publication. Its directed relations support a checkable trace, while its approval digest exposes edits that would otherwise leave an outdated approval attached to new content. These mechanisms help explain a task's declared justification and revision history.

The contribution is limited by what formalization can represent. Educators can agree on a formally complete but weak mapping. An evidence portfolio can contain all requested artifacts while failing to demonstrate the intended reasoning. A simple database may implement the same constraints at lower cost for a small curriculum. The architecture should therefore be judged against the concrete tasks it improves and the review burden it creates, rather than the breadth of its component list.

9.2 Limits of the present evidence

RACC-P1 is deliberately narrow. It handles one selected standards path and one task, with fixed profile versions and a small fixture. It does not implement document ingestion, model drafting, dense retrieval, a persistent graph service, multi-user authorization, or LMS publication. The reference approval is simulated. The selected invalid cases were derived from the stated rules and do not establish completeness of the rules or security of a deployment.

The constructed truss is a foundational statics exercise. Its calculations are reproducible, but its workload, rubric reliability, accessibility and educational value have not been validated with learners or independent instructors. Its partial mapping to a broad engineering outcome must remain visible. The local Common Cartridge output demonstrates serialization and selected package checks; import behavior and native assessment features remain platform-specific and untested.

Competency language can be contested and locally interpreted. Future profiles should preserve alternative interpretations, disagreement and explicit uncertainty. Source licensing

© 2026 TRTEE. Open Access under CC BY 4.0

can preclude automated retrieval even when manual review is possible. Model changes can alter candidate wording and review burden. Institutions need evidence that the proposed controls remain understandable and useful across these conditions before adopting the broader system.

10 Conclusion

This article specifies a curriculum compiler that connects applicable standards clauses, local curricular interpretations, learning tasks, assessment evidence and controlled export. The formal profile defines object types, directional relationships, structural diagnostics and approval binding. A complete constructed truss case and local reference execution make those mechanisms inspectable. The executed checks support selected structural and packaging claims, while a semantic counterexample demonstrates the limit of treating those checks as educational or engineering assurance.

Further work should implement and compare the retrieval and drafting components, evaluate mapping and task quality with independent educators, measure total workflow effort, and verify behavior in named learning platforms. Learning effectiveness remains a separate research question. The specification supplies a bounded starting point for that work and retains substantive curriculum decisions with the responsible educators.

© 2026 TRTEE. Open Access under CC BY 4.0

Acknowledgements

The author thanks colleagues in the Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, for discussion of an earlier outline of this manuscript, and two anonymous reviewers for their comments. No external funding was received for this article.

Author Contributions

S. Z.: Conceptualization, Methodology, Validation, Visualization, Supervision, Project Administration, Writing — Original Draft, Writing — Review & Editing. The author has read and agreed to the submitted version of the manuscript.

Conflicts of Interest

The author declares no conflicts of interest.

Generative AI Statement

Generative artificial intelligence was used to assist with language editing and drafting.

© 2026 TRTEE. Open Access under CC BY 4.0

References

[1] Passow, H. J. (2012). Which ABET Competencies Do Engineering Graduates Find Most Important in their Work? Journal of Engineering Education, 101(1), 95–118. https://doi.org/10.1002/j.2168-9830.2012.tb00043.x

[2] Passow, H. J., & Passow, C. H. (2017). What Competencies Should Undergraduate Engineering Programs Emphasize? A Systematic Review. Journal of Engineering Education, 106(3), 475–526. https://doi.org/10.1002/jee.20171

[3] Shuman, L. J., Besterfield-Sacre, M., & McGourty, J. (2005). The ABET “Professional Skills” - Can They Be Taught? Can They Be Assessed? Journal of Engineering Education, 94(1), 41–55. https://doi.org/10.1002/j.2168-9830.2005.tb00828.x

[4] Harden, R. M. (2001). AMEE Guide No. 21: Curriculum mapping: a tool for transparent and authentic teaching and learning. Medical Teacher, 23(2), 123–137. https://doi.org/10.1080/01421590120036547

[5] Biggs, J. (1996). Enhancing teaching through constructive alignment. Higher Education, 32(3), 347–364. https://doi.org/10.1007/bf00138871

[6] Huang, Y., & Huang, J. X. (2026). A Survey on Retrieval-Augmented Text Generation for Large Language Models. ACM Computing Surveys, 58(12), 1–38. https://doi.org/10.1145/3805774

[7] 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(1), 39. https://doi.org/10.1186/s41239-019-0171-0

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

[9] 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, 102274. https://doi.org/10.1016/j.lindif.2023.102274

[10] Chi, M. T. H., & Wylie, R. (2014). The ICAP Framework: Linking Cognitive Engagement to Active Learning Outcomes. Educational Psychologist, 49(4), 219–243. https://doi.org/10.1080/00461520.2014.965823

[11] van Merriënboer, J. J. G., & Sweller, J. (2005). Cognitive Load Theory and Complex Learning: Recent Developments and Future Directions. Educational Psychology Review, 17(2), 147–177. https://doi.org/10.1007/s10648-005-3951-0

[12] Sweller, J. (1988). Cognitive Load During Problem Solving: Effects on Learning. Cognitive Science, 12(2), 257–285. https://doi.org/10.1207/s15516709cog1202_4

[13] Jonsson, A., & Svingby, G. (2007). The use of scoring rubrics: Reliability, validity and educational consequences. Educational Research Review, 2(2), 130–144. https://doi.org/10.1016/j.edurev.2007.05.002

[14] Reddy, Y. M., & Andrade, H. (2010). A review of rubric use in higher education. Assessment & Evaluation in Higher Education, 35(4), 435–448. https://doi.org/10.1080/02602930902862859

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

© 2026 TRTEE. Open Access under CC BY 4.0

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

[17] 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

[18] 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

[19] Sadler, D. R. (1989). Formative assessment and the design of instructional systems. Instructional Science, 18(2), 119–144. https://doi.org/10.1007/bf00117714

[20] 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

[21] Nicol, D., Thomson, A., & Breslin, C. (2014). Rethinking feedback practices in higher education: a peer review perspective. Assessment & Evaluation in Higher Education, 39(1), 102–122. https://doi.org/10.1080/02602938.2013.795518

[22] 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

[23] Hmelo-Silver, C. E. (2004). Problem-Based Learning: What and How Do Students Learn? Educational Psychology Review, 16(3), 235–266. https://doi.org/10.1023/b:edpr.0000034022.16470.f3

[24] Jonassen, D., Strobel, J., & Lee, C. B. (2006). Everyday Problem Solving in Engineering: Lessons for Engineering Educators. Journal of Engineering Education, 95(2), 139–151. https://doi.org/10.1002/j.2168-9830.2006.tb00885.x

[25] 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

[26] Merrill, M. D. (2002). First principles of instruction. Educational Technology Research and Development, 50(3), 43–59. https://doi.org/10.1007/bf02505024

[27] Prince, M. J., & Felder, R. M. (2006). Inductive Teaching and Learning Methods: Definitions, Comparisons, and Research Bases. Journal of Engineering Education, 95(2), 123–138. https://doi.org/10.1002/j.2168-9830.2006.tb00884.x

[28] Guo, J., Fan, Y., Pang, L., Yang, L., Ai, Q., Zamani, H., Wu, C., Croft, W. B., & Cheng, X. (2020). A Deep Look into neural ranking models for information retrieval. Information Processing & Management, 57(6), 102067. https://doi.org/10.1016/j.ipm.2019.102067

[29] Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., & Fung, P. (2023). Survey of Hallucination in Natural Language Generation. ACM Computing Surveys, 55(12), 1–38. https://doi.org/10.1145/3571730

[30] Huang, L., Yu, W., Ma, W., Zhong, W., Feng, Z., Wang, H., Chen, Q., Peng, W., Feng, X., Qin, B., et al. (2025). A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. ACM Transactions on Information Systems, 43(2), 1–55. https://doi.org/10.1145/3703155

© 2026 TRTEE. Open Access under CC BY 4.0

[31] Khosravi, H., Shum, S. B., Chen, G., Conati, C., Tsai, Y.-S., Kay, J., Knight, S., Martinez-Maldonado, R., Sadiq, S., & Gašević, D. (2022). Explainable Artificial Intelligence in education. Computers and Education: Artificial Intelligence, 3, 100074. https://doi.org/10.1016/j.caeai.2022.100074

[32] Holmes, W., Porayska-Pomsta, K., Holstein, K., Sutherland, E., Baker, T., Shum, S. B., Santos, O. C., Rodrigo, M. T., Cukurova, M., Bittencourt, I. I., et al. (2022). Ethics of AI in Education: Towards a Community-Wide Framework. International Journal of Artificial Intelligence in Education, 32(3), 504–526. https://doi.org/10.1007/s40593-021-00239-1

[33] Williamson, B., & Eynon, R. (2020). Historical threads, missing links, and future directions in AI in education. Learning, Media and Technology, 45(3), 223–235. https://doi.org/10.1080/17439884.2020.1798995

[34] Bose, R., & Frew, J. (2005). Lineage retrieval for scientific data processing: a survey. ACM Computing Surveys, 37(1), 1–28. https://doi.org/10.1145/1057977.1057978

[35] Moreau, L., Clifford, B., Freire, J., Futrelle, J., Gil, Y., Groth, P., Kwasnikowska, N., Miles, S., Missier, P., Myers, J., et al. (2011). The Open Provenance Model core specification (v1.1). Future Generation Computer Systems, 27(6), 743–756. https://doi.org/10.1016/j.future.2010.07.005

[36] Hogan, A., Blomqvist, E., Cochez, M., D'amato, C., Melo, G. D., Gutierrez, C., Kirrane, S., Gayo, J. E. L., Navigli, R., Neumaier, S., et al. (2022). Knowledge Graphs. ACM Computing Surveys, 54(4), 1–37. https://doi.org/10.1145/3447772

[37] 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

[38] Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., & Pedreschi, D. (2019). A Survey of Methods for Explaining Black Box Models. ACM Computing Surveys, 51(5), 1–42. https://doi.org/10.1145/3236009

[39] Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., et al. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. Information Fusion, 58, 82–115. https://doi.org/10.1016/j.inffus.2019.12.012

[40] Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence, 1(5), 206–215. https://doi.org/10.1038/s42256-019-0048-x

[41] Peng, B., Zhu, Y., Liu, Y., Bo, X., Shi, H., Hong, C., Zhang, Y., & Tang, S. (2026). Graph Retrieval-Augmented Generation: A Survey. ACM Transactions on Information Systems, 44(2), 1–52. https://doi.org/10.1145/3777378

[42] Celik, I. (2023). Towards Intelligent-TPACK: An empirical study on teachers' professional knowledge to ethically integrate artificial intelligence (AI)-based tools into education. Computers in Human Behavior, 138, 107468. https://doi.org/10.1016/j.chb.2022.107468

[43] Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2023). Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. ACM Computing Surveys, 55(9), 1–35. https://doi.org/10.1145/3560815

© 2026 TRTEE. Open Access under CC BY 4.0

[44] Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., Chen, H., Yi, X., Wang, C., Wang, Y., et al. (2024). A Survey on Evaluation of Large Language Models. ACM Transactions on Intelligent Systems and Technology, 15(3), 1–45. https://doi.org/10.1145/3641289

[45] Streveler, R. A., Litzinger, T. A., Miller, R. L., & Steif, P. S. (2008). Learning Conceptual Knowledge in the Engineering Sciences: Overview and Future Research Directions. Journal of Engineering Education, 97(3), 279–294. https://doi.org/10.1002/j.2168-9830.2008.tb00979.x

[46] Litzinger, T., Lattuca, L. R., Hadgraft, R., & Newstetter, W. (2011). Engineering Education and the Development of Expertise. Journal of Engineering Education, 100(1), 123–150. https://doi.org/10.1002/j.2168-9830.2011.tb00006.x

[47] 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

[48] 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

[49] Jobin, A., Ienca, M., & Vayena, E. (2019). The global landscape of AI ethics guidelines. Nature Machine Intelligence, 1(9), 389–399. https://doi.org/10.1038/s42256-019-0088-2

[50] Mittelstadt, B. D., Allo, P., Taddeo, M., Wachter, S., & Floridi, L. (2016). The ethics of algorithms: Mapping the debate. Big Data & Society, 3(2), 2053951716679679. https://doi.org/10.1177/2053951716679679

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

[52] Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2022). A Survey on Bias and Fairness in Machine Learning. ACM Computing Surveys, 54(6), 1–35. https://doi.org/10.1145/3457607

[53] Bond, M., Marín, V. I., Dolch, C., Bedenlier, S., & Zawacki-Richter, O. (2018). Digital transformation in German higher education: student and teacher perceptions and usage of digital media. International Journal of Educational Technology in Higher Education, 15(1), 48. https://doi.org/10.1186/s41239-018-0130-1

[54] Järvelin, K., & Kekäläinen, J. (2002). Cumulated gain-based evaluation of IR techniques. ACM Transactions on Information Systems, 20(4), 422–446. https://doi.org/10.1145/582415.582418

[55] Davis, F. D. (1989). Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. MIS Quarterly, 13(3), 319–340. https://doi.org/10.2307/249008

[56] Borrego, M., Froyd, J. E., & Hall, T. S. (2010). Diffusion of Engineering Education Innovations: A Survey of Awareness and Adoption Rates in U.S. Engineering Departments. Journal of Engineering Education, 99(3), 185–207. https://doi.org/10.1002/j.2168-9830.2010.tb01056.x

© 2026 TRTEE. Open Access under CC BY 4.0

Source notes

[S1] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. In Advances in Neural Information Processing Systems (Vol. 33, pp. 9459–9474). https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html

[S2] Anderson, L. W., & Krathwohl, D. R. (Eds.). (2001). A taxonomy for learning, teaching, and assessing: A revision of Bloom’s taxonomy of educational objectives. Longman.

[S3] Wiggins, G., & McTighe, J. (2005). Understanding by design (2nd ed.). Association for Supervision and Curriculum Development.

[S4] Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W.-t. (2020). Dense passage retrieval for open-domain question answering. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (pp. 6769–6781). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.550

[S5] Robertson, S., & Zaragoza, H. (2009). The probabilistic relevance framework: BM25 and beyond. Foundations and Trends in Information Retrieval, 3(4), 333–389. https://doi.org/10.1561/1500000019

[S6] 1EdTech Consortium. (2025). Competencies and Academic Standards Exchange service information model (Version 1.1). https://standards.1edtech.org/case/specifications/standards/v1p1/im

[S7] IMS Global Learning Consortium. (2003). Learning design specification (Version 1.0). https://www.1edtech.org/standards/learning-design

[S8] Lusi, J. A., Nikiforova, A., Lorenz, B., & Pappel, I. (2026). An LLM-augmented framework for interdisciplinary curriculum design and innovation: Case of digital public infrastructure education. Quality & Quantity, 60, 13205-13239. https://doi.org/10.1007/s11135-026-02714-0

[S9] Arana, J. M., Mañacop, J. A., Manacop, J. A., Garcia, R. A., Piniera, K. R., Carandang, K. A. M., Casin, E. R., Alis, C., & Monterola, C. (2026). PathBuilder: A quality-controlled LLM system for personalized learning pathways. In Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations) (pp. 504-514). Association for Computational Linguistics. https://doi.org/10.18653/v1/2026.acl-demo.50

[S10] 1EdTech Consortium. (2014). Common Cartridge profile implementation (Version 1.3 final specification, May 30 update). https://www.imsglobal.org/cc/ccv1p3/imscc_Implementation-v1p3.html

[S11] UNESCO. (2023). Guidance for generative AI in education and research. https://unesdoc.unesco.org/ark:/48223/pf0000386693

[S12] ABET. (2024). Criteria for accrediting engineering programs, 2025-2026. https://www.abet.org/2025-2026_eac_criteria/

[S13] Cormack, G. V., Clarke, C. L. A., & Buettcher, S. (2009). Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 758-759). Association for Computing Machinery. https://doi.org/10.1145/1571941.1572114

© 2026 TRTEE. Open Access under CC BY 4.0

[S14] 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. In Proceedings of the Conference on Fairness, Accountability, and Transparency (pp. 220–229). Association for Computing Machinery. https://doi.org/10.1145/3287560.3287596

[S15] National Institute of Standards and Technology. (2023). Artificial intelligence risk management framework (AI RMF 1.0) (NIST AI 100-1). U.S. Department of Commerce. https://doi.org/10.6028/NIST.AI.100-1

© 2026 TRTEE. Open Access under CC BY 4.0

Appendix A Executable reference capsule

Save the following listing as reference_capsule.py. Use Python 3.12 and install jsonschema 4.25.1, then run python reference_capsule.py in an empty working directory. The script creates capsule_output with schema.json, fixture.json, results.json, task.html and task.imscc. It performs the 23 reported fixture checks, the semantic counterexample, package-reference checks and two statics checks. It does not fetch standards, run retrieval or call a model. The separate official-XSD verification described in Appendix B requires the published schema files and an XML Schema validator.

The exported JSON Schema and fixture are generated from the exact constructors below. All fixture content is included; there are no omitted payload fields. The seal function creates a simulated approval for testing and must not be exposed as an approval service in a deployed system. The listing is a reference demonstration, not an authenticated institutional application.

```
"""RACC-P1 v1.0: a bounded, synthetic conformance demonstration.
Run with Python 3.12 and jsonschema 4.25.1. No model or LMS is called.
"""
import copy
import hashlib
import html
import json
from pathlib import Path
import platform
import uuid
import xml.etree.ElementTree as ET
import zipfile
from jsonschema import Draft202012Validator


def obj(properties):
    return dict(type="object", properties=properties,
                required=list(properties), additionalProperties=False)


S = {"type": "string", "minLength": 1}
I = {"type": "integer", "minimum": 1}
H = {"type": "string", "pattern": "^[0-9a-f]{64}$"}


def arr(items, minimum=1):
    return dict(type="array", items=items, minItems=minimum)


def enum(*values):
    return {"enum": list(values)}
```

© 2026 TRTEE. Open Access under CC BY 4.0

```
NODE = obj(dict(id=S, kind=enum("source", "program", "outcome", "task",
    "evidence", "rubric", "prerequisite"), version=I, text=S))
EDGE = obj(dict(id=S, a=S, b=S, relation=S,
    review=enum("proposed", "confirmed", "rejected", "deferred"),
    contribution=enum("introduced", "practiced", "assessed", "na"),
    reviewer=S, rationale=S))
SOURCE = obj(dict(node_id=S, edition=S, uri=S, locator=S, sha256=H,
    normative_status=enum("effective", "proposed", "informative"),
    access=enum("metadata_only", "prompt_allowed", "blocked"),
    mapping_route=enum("manual", "retrieved"), verified_locator=H))
TASK = obj(dict(node_id=S, title=S, phases=arr(S), resources=arr(S),
    class_minutes=I, homework_minutes=I, min_total=I, max_total=I,
    accessibility=S, evidence_method=S, feedback=S, release=enum("draft")))
RUBRIC = obj(dict(node_id=S, evidence_id=S, outcome_id=S,
    weight={"type": "number", "exclusiveMinimum": 0},
    levels={**arr(S, 4), "maxItems": 4}))
BUILD = obj(dict(source_sha256=H, adopted_edition=S, schema=S,
    validator=S, adapter=S, retrieval=S, model=S, prompt=S))
PAYLOAD = obj(dict(profile=enum("RACC-P1-1.0"), nodes=arr(NODE),
    edges=arr(EDGE), source=SOURCE, task=TASK, rubric=arr(RUBRIC),
    build=BUILD))
SCHEMA = obj(dict(payload=PAYLOAD,
    approval=obj(dict(reviewer=S, role=enum("educator"), digest=H))))
SCHEMA["$schema"] = "https://json-schema.org/draft/2020-12/schema"

SIGNATURE = {
    "derived_from": ("program", "source"),
    "contributes_to": ("outcome", "program"),
    "addresses": ("task", "outcome"),
    "produces": ("task", "evidence"),
    "assesses": ("rubric", "evidence"),
    "criterion_for": ("rubric", "outcome"),
}


def digest(value):
    b = json.dumps(value, sort_keys=True, ensure_ascii=True,
                   separators=(",", ":"), allow_nan=False).encode("ascii")
    return hashlib.sha256(b).hexdigest()


def seal(record):
    record["approval"] = dict(reviewer="fixture-reviewer", role="educator",
                              digest=digest(record["payload"]))
    return record


def check(record):
    if list(Draft202012Validator(SCHEMA).iter_errors(record)):
        return ["E_SCHEMA"]
    p = record["payload"]
    errors = set()
    nodes = {n["id"]: n for n in p["nodes"]}
    if len(nodes) != len(p["nodes"]):
        errors.add("E_ID")
    ids = [e["id"] for e in p["edges"]]
```

© 2026 TRTEE. Open Access under CC BY 4.0

```
if len(set(ids)) != len(ids):
    errors.add("E_ID")
bykind = lambda kind: [n["id"] for n in p["nodes"]
                       if n["kind"] == kind]
if any(len(bykind(k)) != 1 for k in
       ("source", "program", "outcome", "task")):
    errors.add("E_CARDINALITY")
links, requires = set(), {}
for e in p["edges"]:
    a, b, rel = e["a"], e["b"], e["relation"]
    if a not in nodes or b not in nodes:
        errors.add("E_REF")
        continue
    pair = (nodes[a]["kind"], nodes[b]["kind"])
    valid = SIGNATURE.get(rel) == pair
    if rel == "requires":
        valid = pair[0] in ("outcome", "prerequisite")
        valid = valid and pair[1] == "prerequisite"
        requires.setdefault(a, []).append(b)
    if not valid:
        errors.add("E_TYPE")
    if e["review"] != "confirmed":
        errors.add("E_REVIEW")
    if rel == "contributes_to" and e["contribution"] == "na":
        errors.add("E_CONTRIBUTION")
    links.add((a, rel, b))
def cycle(v, trail):
    if v in trail:
        return True
    return any(cycle(w, trail | {v}) for w in requires.get(v, []))
if any(cycle(v, set()) for v in requires):
    errors.add("E_CYCLE")
s, t, b = p["source"], p["task"], p["build"]
if (s["node_id"] not in bykind("source") or
        t["node_id"] not in bykind("task")):
    errors.add("E_REF")
if s["normative_status"] != "effective":
    errors.add("E_SOURCE")
if s["edition"] != b["adopted_edition"]:
    errors.add("E_EDITION")
if s["sha256"] != b["source_sha256"]:
    errors.add("E_SOURCE_HASH")
if s["verified_locator"] != digest([s["sha256"], s["locator"]]):
    errors.add("E_LOCATOR")
if s["access"] == "blocked" or (s["access"] == "metadata_only"
                                and s["mapping_route"] != "manual"):
    errors.add("E_ACCESS")
if "E_CARDINALITY" not in errors:
    src, prog, out, task = [bykind(k)[0] for k in
                            ("source", "program", "outcome", "task")]
    path = [(prog, "derived_from", src),
            (out, "contributes_to", prog), (task, "addresses", out)]
    if not all(e in links for e in path):
        errors.add("E_PATH")
    covered = set()
    for r in p["rubric"]:
        rid, eid, oid = r["node_id"], r["evidence_id"], r["outcome_id"]
        if (rid not in bykind("rubric") or eid not in bykind("evidence")
```

© 2026 TRTEE. Open Access under CC BY 4.0

```
                    or oid != out):
                errors.add("E_RUBRIC")
            for edge in [(task, "produces", eid), (rid, "assesses", eid),
                         (rid, "criterion_for", out)]:
                if edge not in links:
                    errors.add("E_EVIDENCE")
            covered.add(eid)
        if not covered or covered != set(bykind("evidence")):
            errors.add("E_EVIDENCE")
        if len(p["rubric"]) != len(set(r["node_id"] for r in p["rubric"])):
            errors.add("E_RUBRIC")
    if abs(sum(r["weight"] for r in p["rubric"]) - 100) > 1e-9:
        errors.add("E_WEIGHT")
    total = t["class_minutes"] + t["homework_minutes"]
    if not t["min_total"] <= total <= t["max_total"]:
        errors.add("E_WORKLOAD")
    if (b["schema"], b["validator"], b["adapter"]) != (
            "P1-1.0", "P1-1.0", "CC13-web-1.0"):
        errors.add("E_PROFILE")
    if record["approval"]["digest"] != digest(p):
        errors.add("E_APPROVAL")
    return sorted(errors)


def fixture():
    records = [
        ("s", "source", "ABET EAC Criterion 3 Student Outcome 1"),
        ("p", "program", "Formulate engineering problems using models."),
        ("c", "outcome", "Explain assumptions and compare truss results."),
        ("t", "task", "Analyze A(0,0), B(6,0), C(3,4) in metres. "
         "Use pin A and vertical roller B. At C apply (H,-P) kN: "
         "case 1 (0,-12), case 2 (3,-18). Bars are AB, AC, BC. "
         "Ideal pins and weightless axial bars are assumed. "
         "Use tension-positive member forces and rightward/upward-positive "
         "reactions. Strength, buckling, deflection and connections are "
         "outside this model. "
         "Formulate the problem, draw free bodies, solve all member "
         "forces and reactions, and check joint residuals in a sheet. "
         "Explain limitations; no real structure is certified."),
        ("e", "evidence", "One portfolio: question, diagrams, assumptions, "
         "two calculations, sheet residuals and a limitations note."),
        ("q", "prerequisite", "Planar equilibrium and spreadsheet formulas"),
    ]
    constructs = ["Formulation and assumptions", "Equilibrium and units",
                  "Check and interpretation", "Limitations and communication"]
    records += [("r" + str(i), "rubric", v)
                for i, v in enumerate(constructs, 1)]
    nodes = [dict(id=i, kind=k, version=1, text=v) for i, k, v in records]
    triples = [("p", "derived_from", "s"),
               ("c", "contributes_to", "p"), ("t", "addresses", "c"),
               ("t", "produces", "e"), ("c", "requires", "q")]
    for i in range(1, 5):
        triples += [("r" + str(i), "assesses", "e"),
                    ("r" + str(i), "criterion_for", "c")]
    edges = [dict(id="g" + str(i), a=a, relation=r, b=b,
        review="confirmed", contribution="practiced" if r ==
        "contributes_to" else "na", reviewer="fixture-reviewer",
        rationale="Constructed partial-practice mapping; not attainment.")
```

© 2026 TRTEE. Open Access under CC BY 4.0

```
        for i, (a, r, b) in enumerate(triples)]
    sha = "89d3124c5a6c77d6d21e5d58f0c3f7948273876f27c093883ed05e6740e8d031"
    loc = "PDF page 8; printed page 6; Criterion 3; Student Outcome (1)"
    source = dict(node_id="s", edition="2025-2026",
        uri="https://www.abet.org/2025-2026_eac_criteria/", locator=loc,
        sha256=sha, normative_status="effective", access="metadata_only",
        mapping_route="manual", verified_locator=digest([sha, loc]))
    task = dict(node_id="t", title="Diagnose a simplified roof truss",
        phases=["Frame the question and assumptions (15 min)",
                "Draw and calculate (25 min)", "Discuss checks (20 min)",
                "Complete the portfolio and revise (60 min homework)"],
        resources=["Geometry and loads in the prompt; paper or text diagram",
                   "Any local spreadsheet or an equivalent calculation table"],
        class_minutes=60, homework_minutes=60, min_total=90, max_total=150,
        accessibility="Geometry is provided as text coordinates. A structured "
        "text or oral explanation may replace drawing, while preserving "
        "force and assumption evidence. No paid software is required.",
        evidence_method="Submit one portfolio after class and homework.",
        feedback="Compare residuals and rubric criteria; explain one revision.",
        release="draft")
    anchors = [
        ["Absent", "Question or assumptions unclear", "Question and most "
         "assumptions justified", "Question, boundaries and all key "
         "assumptions "
         "explicit and justified"],
        ["Absent", "Major equilibrium or unit errors", "Mostly correct; "
         "minor local errors", "Both cases correct with units and signs"],
        ["Absent", "Reports agreement only", "Computes residuals with limited "
         "interpretation", "Explains residuals, discrepancies and "
         "model limits"],
        ["Absent", "Unqualified safety claim or unclear report", "States some "
         "limits clearly", "Clear report identifies omitted behavior and "
         "additional evidence needed for design"],
    ]
    rubric = [dict(node_id="r" + str(i + 1), evidence_id="e", outcome_id="c",
                   weight=w, levels=a)
              for i, (w, a) in enumerate(zip([25, 35, 20, 20], anchors))]
    build = dict(source_sha256=sha, adopted_edition="2025-2026",
        schema="P1-1.0", validator="P1-1.0", adapter="CC13-web-1.0",
        retrieval="RRF illustrative ranks only", model="none",
        prompt="human-authored fixture 1.0")
    return seal(dict(payload=dict(profile="RACC-P1-1.0", nodes=nodes,
        edges=edges, source=source, task=task, rubric=rubric, build=build)))


def export(record, destination):
    if check(record):
        raise ValueError("Compilation blocked: " + ",".join(check(record)))
    p, esc = record["payload"], html.escape
    t = p["task"]
    nodes = {n["id"]: n for n in p["nodes"]}
    heading = lambda x: "<h2>" + esc(x) + "</h2>"
    para = lambda x: "<p>" + esc(x) + "</p>"
    page = '<!doctype html><html lang="en"><meta charset="utf-8">'
    page += '<title>' + esc(t["title"]) + '</title><body>'
    page += '<h1>' + esc(t["title"]) + '</h1>'
    page += para(nodes[t["node_id"]]["text"])
    page += heading("Sequence") + "".join(map(para, t["phases"]))
```

© 2026 TRTEE. Open Access under CC BY 4.0

```
    page += heading("Resources") + "".join(map(para, t["resources"]))
    for key in ["evidence_method", "accessibility", "feedback"]:
        page += heading(key.replace("_", " ").title()) + para(t[key])
    page += heading("Evidence")
    page += "".join(para(n["text"]) for n in p["nodes"]
                    if n["kind"] == "evidence")
    page += heading("Rubric")
    for r in p["rubric"]:
        page += para(nodes[r["node_id"]]["text"] +
                     "; weight " + str(r["weight"]) + "%")
        page += "".join(para(str(i) + ": " + x)
                        for i, x in enumerate(r["levels"]))
    page += para("Weighted score = sum(weight x level / 3).")
    page += "</body></html>"
    ns = "http://www.imsglobal.org/xsd/imsccv1p3/imscp_v1p1"
    ET.register_namespace("", ns)
    tag = lambda x: "{" + ns + "}" + x
    guid = lambda x: "G" + str(uuid.uuid5(uuid.NAMESPACE_URL, digest(p) + x))
    root = ET.Element(tag("manifest"), identifier=guid(digest(p)))
    md = ET.SubElement(root, tag("metadata"))
    ET.SubElement(md, tag("schema")).text = "IMS Common Cartridge"
    ET.SubElement(md, tag("schemaversion")).text = "1.3.0"
    lomns = "http://ltsc.ieee.org/xsd/imsccv1p3/LOM/manifest"
    ET.register_namespace("lom", lomns)
    lomtag = lambda x: "{" + lomns + "}" + x
    lom = ET.SubElement(md, lomtag("lom"))
    general = ET.SubElement(lom, lomtag("general"))
    title = ET.SubElement(general, lomtag("title"))
    ET.SubElement(title, lomtag("string"), language="en").text = t["title"]
    orgs = ET.SubElement(root, tag("organizations"))
    org = ET.SubElement(orgs, tag("organization"),
                        identifier=guid("org"), structure="rooted-hierarchy")
    group = ET.SubElement(org, tag("item"), identifier=guid("group"))
    item = ET.SubElement(group, tag("item"), identifier=guid("item"),
                         identifierref=guid("resource"))
    ET.SubElement(item, tag("title")).text = t["title"]
    rs = ET.SubElement(root, tag("resources"))
    r = ET.SubElement(rs, tag("resource"), identifier=guid("resource"),
        type="webcontent", intendeduse="assignment",
        href="webcontent/task.html")
    ET.SubElement(r, tag("file"), href="webcontent/task.html")
    data = ET.tostring(root, encoding="utf-8", xml_declaration=True)
    destination = Path(destination)
    destination.mkdir(parents=True, exist_ok=True)
    (destination / "task.html").write_text(page, encoding="utf-8")
    with zipfile.ZipFile(destination / "task.imscc", "w") as z:
        for name, value in [("imsmanifest.xml", data),
                            ("webcontent/task.html", page.encode())]:
            z.writestr(zipfile.ZipInfo(name, (2026, 1, 1, 0, 0, 0)), value)
    return digest(p)


def run(destination):
    base = fixture()
    cases = [("valid", None, None, None, False)]
    cases += [
        ("missing-field", ["task", "accessibility"], None, "E_SCHEMA", True),
        ("wrong-number-type", ["task", "class_minutes"],
```

© 2026 TRTEE. Open Access under CC BY 4.0

```
     "60", "E_SCHEMA", True),
    ("unknown-field", ["task", "extra"], 1, "E_SCHEMA", True),
    ("duplicate-node", ["nodes", 1, "id"], "s", "E_ID", True),
    ("dangling-edge", ["edges", 0, "b"], "missing", "E_REF", True),
    ("wrong-edge-type", ["edges", 0, "a"], "t", "E_TYPE", True),
    ("unconfirmed-edge", ["edges", 0, "review"],
     "proposed", "E_REVIEW", True),
    ("missing-contribution", ["edges", 1, "contribution"], "na",
     "E_CONTRIBUTION", True),
    ("prerequisite-cycle", ["edges", 4, "a"], "q", "E_CYCLE", True),
    ("proposed-clause", ["source", "normative_status"], "proposed",
     "E_SOURCE", True),
    ("wrong-edition", ["source", "edition"],
     "2024-2025", "E_EDITION", True),
    ("source-digest-change", ["source", "sha256"], "0" * 64,
     "E_SOURCE_HASH", True),
    ("locator-change", ["source", "locator"], "page 99", "E_LOCATOR", True),
    ("blocked-source", ["source", "access"], "blocked", "E_ACCESS", True),
    ("unauthorized-retrieval", ["source", "mapping_route"], "retrieved",
     "E_ACCESS", True),
    ("broken-evidence", ["rubric", 0, "evidence_id"], "missing",
     "E_EVIDENCE", True),
    ("bad-weight", ["rubric", 0, "weight"], 30, "E_WEIGHT", True),
    ("bad-rubric-scale", ["rubric", 0, "levels"],
     ["0", "1", "2", "3", "4"], "E_SCHEMA", True),
    ("workload-exceeded", ["task", "homework_minutes"], 1000,
     "E_WORKLOAD", True),
    ("wrong-profile", ["build", "schema"], "P1-9", "E_PROFILE", True),
    ("edit-after-approval", ["task", "title"], "Changed title",
     "E_APPROVAL", False),
    ("unauthorized-approver", None, None, "E_SCHEMA", False),
]
results = []
for name, path, value, expected, reseal in cases:
    item = copy.deepcopy(base)
    if path:
        target = item["payload"]
        for key in path[:-1]:
            target = target[key]
        if value is None:
            del target[path[-1]]
        else:
            target[path[-1]] = value
    if name == "unauthorized-approver":
        item["approval"]["role"] = "model"
    if reseal:
        seal(item)
    got = check(item)
    passed = expected in got if expected else not got
    assert passed, (name, got, expected)
    results.append(dict(case=name, expected=expected or "PASS",
                        actual=got or ["PASS"]))
# Explicit non-guarantee: semantic falsehood can pass structural checks.
unsafe = copy.deepcopy(base)
unsafe["payload"]["nodes"][3]["text"] = "Prove the real truss is safe."
seal(unsafe)
assert check(unsafe) == []
export(base, destination)
```

© 2026 TRTEE. Open Access under CC BY 4.0

```
    out = Path(destination)
    with zipfile.ZipFile(out / "task.imscc") as z:
        root = ET.fromstring(z.read("imsmanifest.xml"))
        for f in root.findall(".//{*}file"):
            assert f.attrib["href"] in z.namelist()
    # Independent equilibrium residuals for the two constructed load cases.
    statics = []
    for P, Hh in [(12., 0.), (18., 3.)]:
        ax, ay, by = -Hh, P / 2 - 2 * Hh / 3, P / 2 + 2 * Hh / 3
        ac, bc = -5 * P / 8 + 5 * Hh / 6, -5 * P / 8 - 5 * Hh / 6
        ab = 3 * P / 8 + Hh / 2
        residuals = [ax + ab + .6 * ac, ay + .8 * ac,
                     -ab - .6 * bc, by + .8 * bc,
                     -.6 * ac + .6 * bc + Hh, -.8 * ac - .8 * bc - P]
        assert max(map(abs, residuals)) < 1e-10
        statics.append(dict(P=P, H=Hh, Ax=ax, Ay=ay, By=by,
                            AC=ac, BC=bc, AB=ab,
                            residual=max(map(abs,residuals))))
    report = dict(python=platform.python_version(), checks=results,
        semantic_counterexample=(
            "structurally accepted after simulated approval"),
        package_files=2, manifest_references="resolved", statics=statics,
        payload_digest=digest(base["payload"]))
    for name, value in [("schema.json", SCHEMA), ("fixture.json", base),
                        ("results.json", report)]:
        (out / name).write_text(json.dumps(value, indent=2), encoding="utf-8")
    print(json.dumps(report, indent=2))
    return report


if __name__ == "__main__":
    run("capsule_output")
```

© 2026 TRTEE. Open Access under CC BY 4.0

Appendix B Provenance and verification record

Reference capsule

RACC-P1-1.0; Python 3.12.14; jsonschema 4.25.1

Fixture outcomes

1 valid accepted; 22 selected invalid fixtures blocked; semantic counterexample accepted after simulated approval.

Source

ABET 2025-2026 EAC criteria; copyright 2024; Criterion 3 Student Outcome (1); PDF page 8, printed page 6.

Source PDF SHA256

`89d3124c5a6c77d6d21e5d58f0c3f7948273876f27c093883ed05e6740e8d031`

Approved fixture payload SHA256

`60c9646676081c18f9262b70062dc839200aba01b6322bdf6ab4ec880a96c307`

Reference script SHA256

`8329cc3bfdc73e5fb458d801d7792c9799ebfa1d15621c5126002ac6834edc4c`

Export ZIP SHA256

`90063eda28b9459a208c943bc5c6d3c360b97fa2cdccb1391b4f211dafb4dff2`

XSD check

Passed against six fetched schema files; lxml 6.1.1; full certification and LMS import not performed.

Main XSD

https://www.imsglobal.org/profile/cc/ccv1p3/ccv1p3_imscp_v1p2_v1p0.xsd

Original main XSD SHA256

`ad96ef19dfb66d2d22c20722a46a3c3b37b9ba8a155d496d18b425fa887b035f`

For the external XML check, extract imsmanifest.xml from task.imscc. Obtain the Common Cartridge 1.3 content-package XSD at the URL recorded above and recursively obtain its import and include dependencies from their declared locations. Preserve the original files and record their digests; a local copy may rewrite schemaLocation values to corresponding local paths. Validate the manifest using an XML Schema 1.0 processor. The reported run used lxml 6.1.1. The script digest refers to the UTF-8 listing with LF line endings, trailing spaces removed and one final newline. Successful XSD validation establishes the XML constraints encoded in those schemas; it does not execute the complete Common Cartridge conformance suite.

To check package integrity, resolve each manifest file reference to an exact member of the ZIP and reject missing files or unsafe paths. The demonstrated exporter emits only fixed local filenames and escaped text. Inspect task.html locally, then, for deployment, repeat the learner-view inspection in the actual LMS after configuring its assignment settings. The

© 2026 TRTEE. Open Access under CC BY 4.0

published example has no learner data, external scripts, answer key, authentication credential or source-standard text in its exported HTML.

© 2026 TRTEE. Open Access under CC BY 4.0