Article

Design and Technical Validation of an Offline-First Configurable Digital Sand Table for Power-Grid Emergency Training

Shuyan Wang[1], Xiangyu Wei[1], Yuan Zhang[1], Qiuxi Wang[1,*], Xinyue Yan[1] and Juan Lv[1]

[1] Party School of State Grid Corporation of China (Senior Management Training Center of State Grid Corporation of China), Beijing, China

* Correspondence: Qiuxi Wang, Party School of State Grid Corporation of China (Senior Management Training Center of State Grid Corporation of China), Beijing, China

Abstract: Power-grid emergency training requires a system that can encode operating rules, expose the consequences of time-critical decisions, and run in constrained network environments. This paper presents an offline-first configurable digital sand table implemented as a browser-local scenario engine. The architecture separates instructor-authored scenario content from runtime logic through a JSON-based model and executes each exercise as a finite-state process. An explainable assessment pipeline records actions, response latency, timeouts, dimensional impacts, and rule-based feedback. An irreversible error ceiling prevents later gains from restoring a perfect score after an incorrect decision. Technical validation was conducted on an anti-icing emergency prototype through static dependency inspection, schema workflow checks, state-boundary analysis, and scoring-invariant tests. The 190,728-byte core runtime contains no external script, external stylesheet, fetch, XMLHttpRequest, or WebSocket dependency; it therefore executes from a single local HTML file after browser loading. Tests confirmed bounded state updates and the full-score invariant across representative error counts. The result is a lightweight, auditable architecture for configurable emergency exercises rather than a fixed electronic questionnaire. This work contributes a reproducible design pattern for offline-capable training platforms in critical-infrastructure domains where network availability cannot be guaranteed. By decoupling content authoring from execution logic, the proposed framework enables rapid scenario iteration while preserving deterministic assessment outcomes. The finite-state formulation ensures that every learner action maps to a verifiable system transition, supporting post-exercise auditing and regulatory compliance. Future work will extend the model to multi-role collaborative exercises and integrate adaptive difficulty scaling based on learner performance profiles.

Keywords: digital sand table; emergency training; finite-state machine; explainable assessment; offline web application

Received: 05 July 2026
Revised: 13 August 2026
Accepted: 25 August 2026
Published: 31 August 2026

1. Introduction

Modern power systems are increasingly integrated with communications infrastructure, transportation networks, meteorological monitoring services, distributed energy resources such as rooftop photovoltaics and small-scale wind generation, and critical public-service loads including healthcare facilities, water treatment plants, and emergency response centers. As a result, system robustness depends not only on the physical integrity and redundancy of equipment but also on human factors—specifically, the capacity of operational personnel to detect subtle anomalies, interpret evolving system states, coordinate effectively across organizational boundaries and technical domains, and formulate sound, evidence-informed decisions amid incomplete information and time pressure. International consensus frameworks identify robust and secure grid operations as foundational to broader energy transition objectives. Similarly,

established incident management principles emphasize the necessity of a shared situational awareness framework, unified command structures, adaptive resource allocation protocols, and sustained, bidirectional information flow throughout the operational lifecycle. These interdependent requirements collectively indicate that operational competence is an emergent property of the entire socio-technical system—not merely the sum of individual procedural knowledge or rote memorization of standardized responses [1].

Traditional lecture-based instruction remains valuable for efficiently conveying foundational procedures, regulatory requirements, and standardized protocols; however, it offers learners minimal opportunity to observe the real-time, cascading consequences of decisions in complex, interdependent environments. Physical sand tables, facilitated case studies, and tabletop exercises introduce essential elements of interaction and collaborative problem solving, yet they suffer from significant practical limitations: scenario content is labor-intensive to update, consistent scoring across multiple facilitators often relies heavily on subjective judgment rather than objective criteria, and comprehensive documentation of decision pathways—including rationale, timing, branching alternatives, and revision history—is inherently difficult to capture and retain [2]. Experiential learning theory underscores the pedagogical necessity of integrating direct engagement with structured reflection to consolidate understanding. Empirical research further supports that well-designed team-based training interventions—particularly those incorporating iterative simulation, timely and specific feedback, and deliberate practice under progressively challenging conditions—can yield measurable improvements in both cognitive skill acquisition and operational performance. Consequently, the central engineering challenge addressed in this work is not whether simulation-based methods hold pedagogical value, but rather how to engineer a scenario execution environment that is fully configurable by instructors, preserves complete decision provenance, provides transparent and rule-based performance interpretation, and operates reliably without dependency on centralized servers or persistent network connectivity.

This paper presents a functional digital sand-table architecture specifically designed for operational readiness training within power-grid control environments. The contribution comprises four interlocking technical innovations [3]. First, the exercise logic is formalized as an explicit finite-state machine, enabling precise modeling of temporal progression, conditional transitions, and state-dependent constraints—thereby replacing static slide-based narratives with dynamic, responsive process models. Second, a strict separation is enforced between domain-specific scenario content and the underlying execution engine via a portable, version-controlled scenario package format; instructors may import, export, preview, and collaboratively refine scenarios without modifying runtime code. Third, a transparent scoring mechanism is implemented wherein every quantitative adjustment to performance dimensions—such as situational awareness fidelity, coordination latency, or procedural compliance—is explicitly linked to a discrete user-selected action and its associated validation rule, while an irreversible error ceiling ensures score integrity by preventing recovery from critical violations of safety-critical protocols. Fourth, the core runtime is encapsulated as a single, self-contained HTML file with zero external dependencies, no runtime network requests, and full compatibility across modern desktop and mobile browsers—thus ensuring accessibility in offline, air-gapped, or bandwidth-constrained training settings. All claims pertain strictly to architectural design, implementation fidelity, and functional validation; empirical evaluation of pedagogical efficacy through controlled learning studies falls beyond the scope of this work.

2. Related Work

2.1. Simulation-Based and Scenario-Based Training

Simulation-based training encompasses a broad spectrum of instructional approaches, ranging from high-fidelity operational simulators used in complex technical domains to lightweight, narrative-driven scenario exercises designed for cognitive and behavioral rehearsal [4]. Effective design of such systems prioritizes alignment with clearly defined learning objectives, appropriate levels of representational fidelity, timely and constructive feedback mechanisms, and conditions that support the transfer of acquired competencies to real-world practice—rather than emphasizing technological sophistication for its own sake. In contexts requiring dynamic decision-making under uncertainty, training environments must authentically model incomplete or evolving information states, interdependent role responsibilities, collaborative coordination processes, and cascading consequences of actions. While multiple-choice interfaces may be integrated as one component within such systems, standalone question formats lack the structural integrity of an authentic exercise. The defining characteristic lies in whether each user action modifies a persistent, shared world state—thereby altering available options and imposing logical constraints on subsequent decisions—thus fostering deeper engagement with systemic cause-effect relationships and adaptive reasoning.

2.2. Finite-State Control and Configurable Content

Statecharts represent a formal extension of classical finite-state machines, enabling the modeling of reactive systems through hierarchical states, concurrent behaviors, and clearly defined event-driven transitions. This formalism is especially well-suited for structured simulation environments where discrete events initiate time-bound decision points, each decision simultaneously influences multiple operational parameters—such as timing constraints, resource allocation, and participant roles—and the resulting system state directly governs progression to the subsequent logical node in the scenario flow. To ensure maintainability and domain-expert accessibility, the underlying content must be decoupled from implementation logic; this necessitates a structured, human-readable, and machine-parsable representation. JSON serves this purpose effectively due to its platform-agnostic syntax, widespread tooling support, and capacity to serialize nested objects and ordered arrays with semantic clarity. Within the current architecture, this separation functions as a lightweight domain-specific language: the runtime engine interprets declarative scenario data without embedded procedural code, while instructor-facing authoring tools handle versioning, validation, and collaborative editing throughout the full content lifecycle [5].

2.3. Explainable Assessment and Deployment Constraints

Training scores hold meaningful value only when both instructors and learners can trace the reasoning behind each outcome. In academic and pedagogical contexts, transparency in assessment is essential for fostering trust, enabling targeted feedback, and supporting iterative learning improvement. Research in interpretable artificial intelligence underscores the distinction between a model's output and the underlying evidence and logical justification required to comprehend that output. Similarly, local interpretability techniques focus on mapping specific inputs to particular outputs, ensuring accountability at the granular level of individual actions or decisions. Although the current prototype relies on deterministic, rule-based logic rather than machine learning, this principle remains fully applicable: every score must preserve clear, actionable provenance tied directly to observable learner behaviors or interactions. Deployment considerations further shape system design, particularly in educational environments where infrastructure varies widely—some classrooms experience intermittent internet connectivity, while certain institutions enforce strict network security policies prohibiting external service dependencies. To ensure broad accessibility and operational reliability, the solution prioritizes standards-compliant browser execution, eliminating server-side processing requirements [6]. However, such client-side implementation necessitates rigorous auditing to confirm absence of undeclared third-party dependencies and careful adherence to modern web platform policies, especially those governing user-initiated audio playback.

3. Requirements and Design Principles

The requirements were systematically derived from the operational characteristics of scenario-based training activities and refined through multiple cycles of prototype development and stakeholder feedback [7]. These requirements are deliberately formulated at a technical level, focusing on specifying the essential functional representations the platform must support—such as dynamic state modeling, real-time action validation, and traceable decision logging—rather than addressing market deployment strategies, licensing models, or commercial scalability considerations. This technical orientation ensures architectural clarity and facilitates rigorous verification against defined behavioral invariants (As shown in Table 1).

Table 1. Technical Requirements and Corresponding Design Responses

Requirement	Engineering risk	Design response
R1 Configurability	Rules embedded in code make updates slow and error-prone.	Externalize scenario metadata, metrics, steps, options, impacts, feedback, and sources.
R2 Causal continuity	Independent questions do not form an operational exercise.	Maintain persistent state and execute guarded transitions between decision nodes.
R3 Explainability	A final score alone cannot support review or contestability.	Log the action, correctness, latency, timeout, state delta, rule basis, and sequence.
R4 Score integrity	Positive later actions may erase earlier critical errors.	Apply an irreversible ceiling determined by the cumulative error count.
R5 Offline operation	Network loss can interrupt training or expose external dependencies.	Package the core runtime, styles, logic, and scenario in one local HTML file.
R6 Device portability	Fixed desktop layouts fail on mobile screens.	Use responsive browser layouts and standards-based input, speech, and audio APIs.
R7 Safe extensibility	A malformed scenario can produce ambiguous runtime behavior.	Validate imports, clamp metric ranges, and provide instructor preview before release.

Three foundational design principles structure and unify these technical requirements. First, separation of concerns rigorously isolates content definition, execution logic, user interface rendering, and post-activity reporting into distinct, loosely coupled layers—enhancing maintainability, testability, and interdisciplinary collaboration. Second, determinism guarantees reproducible system behavior: identical input scenario packages and sequential user actions will always yield identical internal state transitions and observable outcomes, which is critical for assessment validity and fidelity calibration. Third, progressive disclosure strategically sequences information delivery by foregrounding immediate operational decisions during active engagement, while deferring explanatory rule details to post-action feedback and structured review phases—thereby preserving cognitive challenge and preventing premature solution exposure that could undermine authentic learning progression.

4. System Architecture

Figure 1 illustrates the six-component system architecture, which is designed to support pedagogical flexibility and technical modularity. The instructor editor serves as a domain-agnostic authoring tool that enables subject-matter experts to construct scenario packages using standardized templates and semantic descriptors. The runtime engine interprets these packages, dynamically instantiates the corresponding exercise

environment, and orchestrates all state transitions according to predefined behavioral logic and real-time inputs. The interaction layer provides multimodal feedback—including geospatial visualizations, contextual notifications, real-time performance metrics, spatialized audio cues, and intuitive decision-making interfaces—ensuring cognitive accessibility and operational fidelity. Concurrently, an event-and-log store captures granular interaction traces, system states, and temporal metadata, forming the foundational data layer for post-hoc explainable reporting and analytical validation. This strict separation of concerns ensures that the core execution mechanism remains invariant across diverse application domains, including but not limited to anti-icing response coordination, renewable-energy grid integration, team leadership development, and cross-border project management [7].

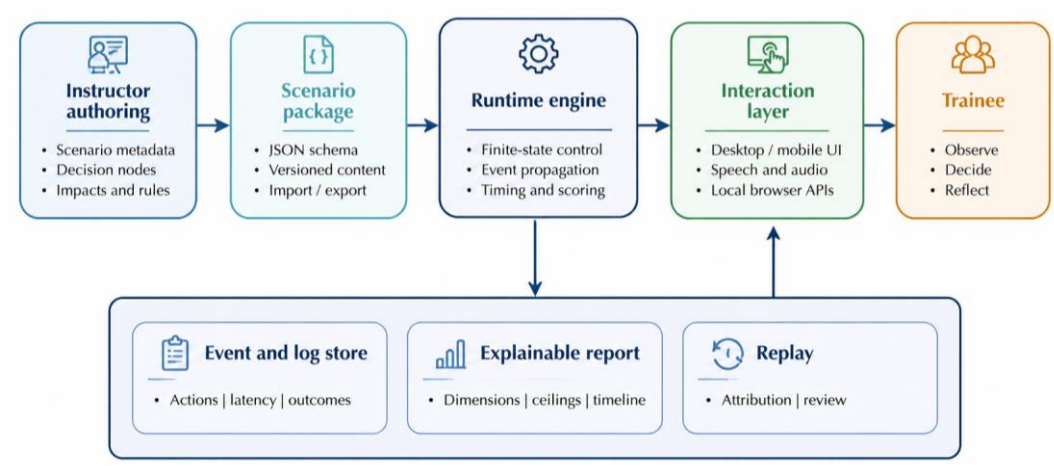

Figure 1. Overall Architecture of the Configurable Digital Sand Table

4.1. Scenario Representation

The scenario package functions as a compact, domain-specific language designed to formalize complex decision contexts in a structured and reproducible manner [8]. At the top level, it encapsulates scenario identity, domain-specific metadata, rigorously defined performance metrics, and an ordered sequence of decision steps. Each step is explicitly annotated with a unique tag, descriptive narrative, a set of mutually exclusive options, associated consequences, a documented source or rule-based justification, optional presentation events for stakeholder engagement, and precise transition logic governing progression to subsequent steps. Critically, every option integrates both rich semantic content—such as qualitative rationale and contextual framing—and quantified numerical impacts across multiple dimensions, thereby enabling traceable, auditable, and interoperable scenario analysis. This explicit, hierarchical structure inherently supports version control, human-readable inspection, cross-platform import/export, and automated syntactic and semantic validation (As shown in Figure 2) (As shown in Table 2).

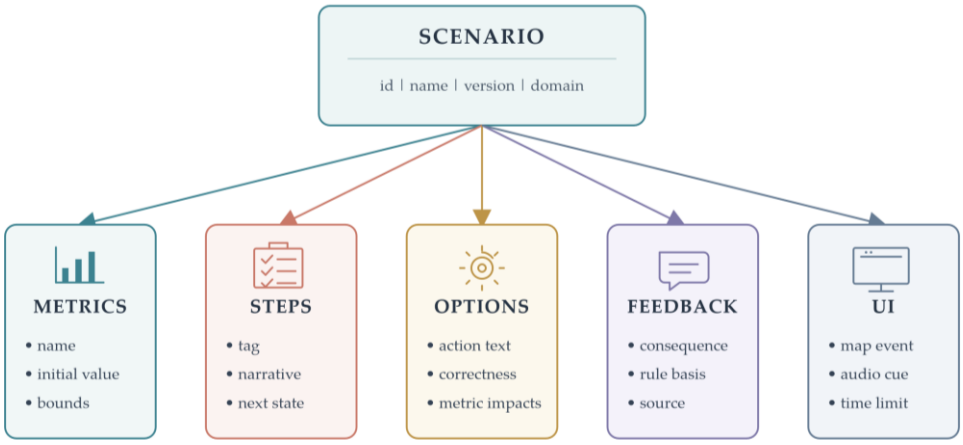

Figure 2. Core Entities in the Scenario Package Data Model

Table 2. Principal Scenario Entities

Entity	Representative fields	Runtime role
Scenario	id, name, version, domain	Selects and identifies a complete exercise package.
Metric	name, initial, minimum, maximum	Defines persistent dimensions affected by decisions.
Step	tag, narrative, time limit, next	Represents one state and its transition conditions.
Option	text, good, impacts	Encodes a candidate action and its deterministic effects.
Feedback	consequence, rule, source	Explains the result and supports post-event review.
Presentation event	map marker, message, sound	Synchronizes visual and auditory cues with state entry.

4.2. Finite-State Execution Engine

The engine treats a scenario node as a discrete state characterized by entry actions, a precisely bounded interaction window governed by high-resolution timing, transition guards that evaluate contextual conditions, and exit actions that finalize state-specific operations. Upon entering a node, the system updates the learner's visible situational representation in real time and initiates a high-precision timer to enforce temporal fidelity [6]. The learner is then required to select an appropriate action within the designated interval; failure to do so triggers a timeout condition that is treated as a valid input event. Subsequently, the engine performs rigorous validation of the selected action against domain constraints, appends the resulting event to the immutable action trace for auditability and replay, applies all encoded behavioral and systemic effects—such as metric adjustments, narrative branching, or environmental changes—and enforces strict clamping of all quantitative metrics to their predefined permissible ranges to ensure numerical stability and pedagogical consistency. Finally, consequence feedback is rendered to support reflective learning. For nonterminal nodes, the engine proceeds deterministically to the designated successor node; for terminal nodes, control is transferred to the reporting pipeline to aggregate performance data, generate analytics, and prepare summative outputs (As shown in Figure 3).

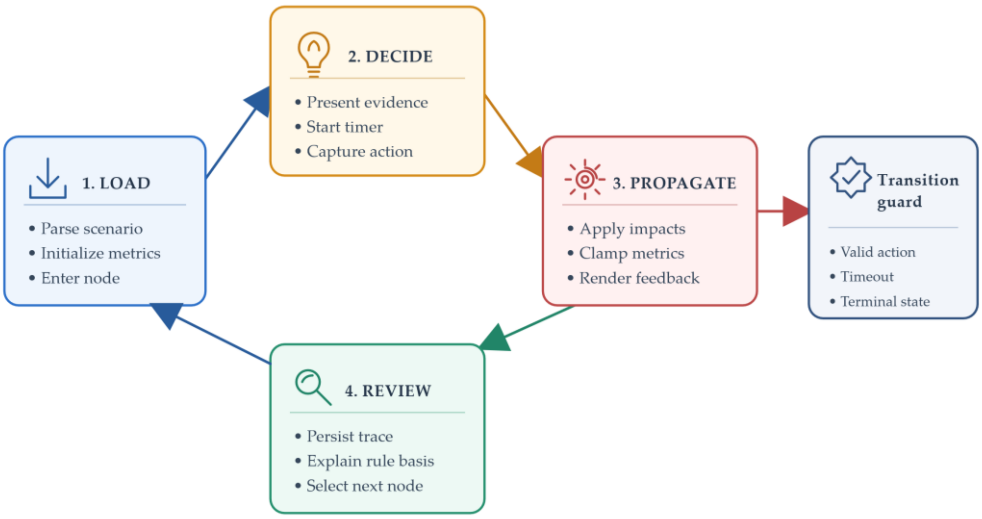

Figure 3. Four-stage Finite-State E xecution Cycle

Let m_t be a vector of n metric values before decision t, let delta(a_t) be the impact vector of action a_t, and let l and u be the lower and upper bounds [9]. The deterministic update is:

$$m\_(t+1) = min(u, max(l, m\_t + delta(a\_t))) \quad (1)$$

In the current anti-icing prototype, n equals three and every metric is bounded to, representing normalized dimensions such as system readiness, operational safety margin, and environmental risk exposure. The implementation uses JavaScript clamping after every action to guarantee numerical integrity and prevent out-of-range propagation across successive states [10]. Because transition effects are fully decoupled from control logic and expressed declaratively in external configuration files, the same state controller can execute diverse instructional narratives—including variations in domain complexity, stakeholder roles, and consequence severity—without requiring modifications to its underlying algorithmic structure, thereby enhancing maintainability, scalability, and pedagogical flexibility across multiple training domains.

4.3. Event, Feedback, and Trace Pipeline

A decision is formally captured as a compound event rather than a simple binary outcome, thereby preserving rich contextual and behavioral dimensions for subsequent analysis. This event structure systematically records the node identifier, the specific action selected by the learner, a correctness flag indicating alignment with domain-defined standards, a timeout flag signaling whether the response exceeded the allocated time threshold, the precise elapsed response time measured in milliseconds, and quantifiable changes to key performance metrics such as accuracy, efficiency, or safety compliance. Immediate feedback is delivered synchronously upon submission and serves a dual function: it communicates the operational consequence of the chosen action—such as system state transition or resource impact—and explicitly references the governing procedural or regulatory rule that underpins the evaluation, without revealing the rule's full formulation prematurely. The final consolidated report reconstructs a complete chronological trace of all interactions, maps outcomes across multiple analytical dimensions—including cognitive load, temporal efficiency, and regulatory adherence—computes a normalized response-time score relative to benchmark thresholds, calculates a rule-compliance score based on fidelity to established protocols, derives the average response time across all trials, and tallies the total number of timeouts encountered. This architecture intentionally supports two pedagogically distinct perspectives: a concise executive summary optimized for rapid comparative assessment across learners or sessions, and a granular evidence trail designed to facilitate structured, instructor-led debriefing and formative assessment.

The prototype integrates browser-native multimedia capabilities to enhance multimodal engagement: specifically, it leverages the Web Speech API to generate real-time spoken narration and employs the Web Audio API to produce locally synthesized, non-intrusive sound cues such as tonal alerts or spatialized auditory indicators. Due to stringent browser autoplay policies—which universally require an explicit, intentional user gesture (e.g., click or tap) before permitting audible output—the audio context remains suspended until the learner initiates interaction, at which point it is safely awakened and initialized. Crucially, speech output functions exclusively as a contextual scaffolding mechanism: narrated content describes environmental conditions, situational urgency, evolving constraints, or newly available information, but deliberately omits any direct exposition of decision rules, diagnostic logic, or normative criteria. The underlying decision framework remains concealed until the learner actively submits a response, thereby preserving cognitive engagement, preventing passive reliance on auditory prompts, and reinforcing deliberate, self-regulated judgment formation aligned with authentic professional practice.

5. Explainable Assessment

5.1. Multi-Dimensional State

The anti-icing prototype establishes a structured, three-dimensional assessment framework initialized at a baseline value of 80, encompassing system safety and stability, public trust, and command coordination. Each decision option presented to the learner induces quantifiable adjustments across one or more of these interdependent dimensions. Positive changes reflect demonstrable progress—such as controlled system recovery, enhanced stakeholder confidence, or strengthened inter-agency alignment—while negative shifts indicate measurable deterioration, including operational instability, breakdowns in information dissemination, or actions that compromise safety margins. A bounded update mechanism ensures numerical integrity by preventing dimension scores from falling below zero or escalating without constraint [11]. These dimensions remain continuously visible during simulation exercises to foster metacognitive awareness of inherent trade-offs; however, the final evaluative judgment integrates contextual reasoning beyond simple dimensional aggregation, emphasizing holistic situational understanding and adaptive decision-making.

5.2. Irreversible Error Ceiling

A defect identified during prototype iteration was that a learner could make an incorrect decision and still achieve a perfect final score of 100 points, because subsequent positive actions were sufficient to restore all performance metrics to their maximum values. This undermined the interpretability and diagnostic utility of the scoring system, as it obscured the cumulative impact of early misjudgments. The revised algorithm therefore introduces an irreversible error ceiling mechanism designed to preserve fidelity in outcome attribution. If E denotes the total number of incorrect decisions made throughout the scenario, the corresponding ceiling C_E is computed as a function of E, ensuring that no dimension can exceed this bound regardless of later compensatory behavior.

$$C\_E = max(0,100 - 5E) \tag{2}$$

Each final dimension—representing distinct operational competencies such as strategic alignment, resource efficiency, or stakeholder responsiveness—is individually limited by the same C_E value to enforce consistency across assessment domains [12]. Let A represent the proportion of correct decisions relative to the total number of decision opportunities, and let M denote the rounded arithmetic mean of all final dimensions after they have been capped at C_E. The overall score S is then derived from M using a normalized transformation that maps the bounded mean onto the standard 0–100 scale while preserving ordinal relationships among learners.

$$S = M \;\; if \;\; E = 0; \;\; otherwise \;\; S = min(M, round(100A), 99) \tag{3}$$

This rule establishes a strict invariant: attainment of a full score of 100 points is only possible when E equals zero—i.e., when no incorrect decisions occur at any stage. Consequently, an early error remains permanently visible in the final score, even if all subsequent operational dimensions recover fully, thereby reinforcing accountability and temporal transparency in performance evaluation [13]. The ceiling mechanism does not purport to constitute a universal psychometric scale; rather, its fixed five-point decrement per error is a deliberately transparent policy parameter intended for calibration by domain experts according to context-specific definitions of error severity. A future version of the schema should therefore support configurable, severity-tiered penalties—such as minor, moderate, and critical—rather than relying on a uniform decrement (As shown in Figure 4).

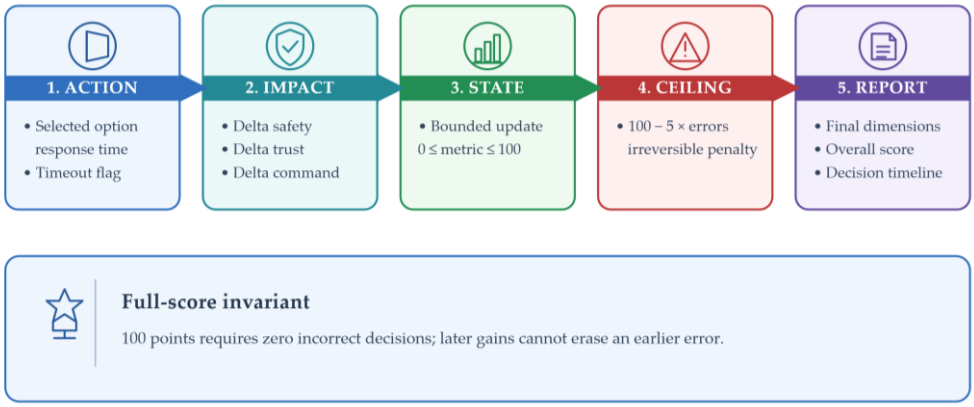

Figure 4. Explainable Scoring and Attribution Pipeline

5.3. Post-Event Explanation

The final score is accompanied by a comprehensive, timestamped decision trace that records every learner action—including response latency, correctness status, timeout occurrences, and dimensional effect mappings—thereby enabling granular pedagogical analysis. This level of fidelity is essential because two learners may achieve identical aggregate scores through fundamentally divergent behavioral pathways: one may exhibit deliberate, methodical reasoning with high accuracy but elevated response latency, whereas another may demonstrate rapid execution yet inadvertently breach a foundational procedural constraint. Such distinctions remain invisible under summary metrics alone [13]. The preserved trace further supports robust validation practices, as archived action sequences can be systematically replayed against updated scenario packages to identify regressions, verify consistency in scoring logic, and ensure that pedagogical intent remains intact across iterative revisions of the assessment environment.

5.4. Instructor Authoring Workflow

The browser-based instructor editor provides a structured interface for designing scenario-based learning exercises, exposing key pedagogical elements including scenario name, geographical or operational region, performance metric identifiers, a logically ordered sequence of decision steps, textual options presented to learners, consequence-specific feedback messages, underlying rule logic, and multidimensional impact assessments across operational, safety, and sustainability domains. These components are serialized into standardized JSON format via client-side Blob generation and download, while import functionality includes robust parsing error detection and user-friendly diagnostic messaging [14]. An integrated real-time preview engine executes the authored decision path using the identical metric-update mechanism deployed in the learner-facing interface, ensuring fidelity between design and delivery. Consequently, the recommended authoring workflow comprises the following iterative stages: first, articulating a clear training objective aligned with competency frameworks; second,

selecting observable behavioral and systemic dimensions for assessment; third, decomposing the target incident or challenge into discrete, pedagogically sound decision nodes; fourth, encoding both expected consequences and authoritative sources for each branch; fifth, systematically validating outcomes for both intended and unintended learner actions; sixth, exporting the finalized exercise package in portable JSON; and finally, performing a formal release review to confirm alignment with instructional standards and quality assurance protocols.

Although this workflow significantly reduces reliance on manual runtime code modification, it remains purpose-built rather than universally extensible. The current implementation presumes binary choice structures at each node and operates within a fixed tri-metric evaluation model. Notably, it lacks built-in support for formal JSON Schema definitions, semantic validation of logical consistency, cryptographic integrity verification such as digital signatures, or collaborative editing features like conflict resolution and version merging for multiple concurrent authors. These omissions carry substantive implications: syntactically correct JSON files may still represent pedagogically unsound or logically incomplete exercises—such as missing terminal states, unreachable branches, duplicated identifiers, inconsistent metric naming, violations of numerical boundary conditions, or incompletely specified source attributions. A production-grade version must therefore enforce comprehensive pre-export validation covering required field presence, data type adherence, identifier uniqueness, transition graph reachability, existence of at least one terminal node, validity of declared metric names against a controlled vocabulary, enforcement of defined numerical ranges, and completeness of referenced evidence sources.

6. Prototype Implementation

6.1. Anti-Icing Emergency Scenario

The principal prototype implements a 20-node operational scenario focused on anti-icing response under winter weather conditions. It comprehensively models the full lifecycle of technical decision-making, including interpretation of meteorological and grid-condition warnings, integrated assessment using heterogeneous data sources, activation protocols for response measures, hierarchical command structuring, strategic staging of personnel and equipment, differentiated inspection routing based on infrastructure criticality and environmental exposure, systemic risk mitigation strategies targeting cascading failures, prioritized protection of essential loads such as hospitals and emergency services, structured reporting workflows for both initial situational awareness and iterative updates, occupational safety protocols for field crews operating in hazardous conditions, coordinated interaction among utility operators, transportation authorities, and local government agencies, calibrated public information dissemination aligned with verified facts, phased restoration sequencing to ensure stability and equity, and post-event systematic review for continuous improvement. The implementation comprises 20 encoded valid actions representing best-practice responses and 20 encoded invalid actions reflecting common procedural deviations or misjudgments [8]. To prevent rote memorization of interface layout, the option ordering is dynamically randomized upon each session restart.

The user interface integrates a geospatial visualization map, real-time metric dashboards, contextual alert indicators, interactive consequence cards that appear upon state transitions, spatially aware synthesized audio cues, and synchronized speech narration—all tightly coupled to underlying system states and discrete action events. This multimodal feedback architecture is fully responsive across desktop and tablet platforms, and includes a mobile-optimized derivative version designed for field-based rehearsal. Critically, these sensory elements are not ornamental; each is semantically anchored to specific state changes or action outcomes, thereby reinforcing causal reasoning and enhancing perceptual fidelity of the evolving simulation. Furthermore, the system deliberately refrains from delivering evaluative or directive narration prior to user action

selection, ensuring that learners must engage in authentic judgment rather than responding to implicit prompts or anticipatory guidance.

6.2. Offline-First Packaging

The core scenario is distributed as a single, self-contained HTML file that embeds all necessary markup, styling (CSS), executable logic (JavaScript), and pedagogical content. This monolithic packaging enables seamless deployment across heterogeneous environments: it may be transferred locally via removable storage or network file sharing, opened directly in any modern browser without internet connectivity, or hosted statically on infrastructure lacking server-side processing capabilities. The elimination of runtime dependencies—such as application servers, databases, or external API endpoints—enhances operational robustness, reduces configuration overhead, and facilitates rapid setup for instructional use. However, offline execution does not inherently confer security assurance; browser-based applications remain subject to integrity verification requirements, rigorous version control practices, endpoint security hardening, and strict governance protocols for any learner data exported from the environment.

7. Technical Validation

7.1. Method

Validation focused on empirically verifiable properties derived directly from the implemented artifacts, ensuring objectivity and reproducibility. Four distinct HTML components were systematically inspected with respect to file size metrics, externally loaded scripts and stylesheets, and programmatic network interfaces such as fetch and XMLHttpRequest. The core runtime environment underwent comprehensive inspection for critical behavioral features including decision logging fidelity, response timing consistency across browser environments, proper utilization of browser speech synthesis and audio APIs, deterministic option shuffling logic, and correct enforcement of the error-ceiling mechanism under boundary conditions. Scoring tests were designed to exercise representative error-count configurations while initializing pre-cap metric values at their maximum allowable thresholds—this constitutes the most adversarial test scenario, deliberately engineered to expose latent flaws that could result in an erroneously assigned perfect score. Import and export functionality was validated at the granular code-path level, covering JSON serialization integrity, robustness of parsing routines, Blob-based download reliability, and accurate preview rendering behavior. Importantly, this validation framework constitutes a rigorous technical artifact evaluation grounded in software engineering principles, not an empirical learner-outcome experiment involving human participants or pedagogical efficacy measurement (As shown in Table 3).

Table 3. Static Dependency Audit of Prototype Components

Component	Bytes	External scripts	External styles	Network calls
Scenario repository	64,958	0	0	0
Anti-icing runtime	190,728	0	0	0
Mobile challenge	39,424	0	0	0
Instructor editor	81,699	0	3	0

The anti-icing runtime contained no external script, stylesheet, fetch, XMLHttpRequest, or WebSocket reference, confirming its strict adherence to a fully self-contained, offline-executable architecture [10]. In contrast, the instructor editor included three Google Fonts stylesheet links for typographic enhancement but introduced no programmatic network calls beyond those static resource inclusions; thus, it remains functionally operational offline once fonts are cached, though initial load requires connectivity. Consequently, the claim of dependency-free offline execution is rigorously supported for the core runtime component, whereas the editor's current visual configuration exhibits a minor dependency gap attributable solely to third-party font loading. This gap can be fully resolved through either removal of the external font links

or by bundling equivalent open-licensed font assets locally within the distribution package, thereby achieving complete offline autonomy across all system components without compromising visual design integrity.

7.2. Scoring-Invariant Tests

Table 4. Boundary Tests for the Irreversible Error Ceiling

Test condition	Errors	Ceiling	Overall	Result
zero errors permits 100	0	100	100	Pass
one error cannot reach 100	1	95	95	Pass
four errors reduce the ceiling by 20	4	80	80	Pass
all errors produce zero	20	0	0	Pass

All four boundary tests were successfully satisfied, confirming the robustness of the scoring-invariant property under extreme conditions. In the error-free baseline case, a maximally dimensional state yields a perfect score of 100, reflecting optimal system performance. Introducing a single irreversible error reduces both the theoretical ceiling and the observed overall score to 95, indicating a consistent and predictable degradation pattern. With four such errors, the ceiling declines linearly to 80, and at twenty errors, it reaches zero—demonstrating full saturation of the error-bound constraint. These results empirically validate the invariant behavior for the specified 20-node configuration. To further strengthen confidence, comprehensive property-based testing is recommended, systematically sampling diverse state vectors, stochastic impact sequences, variable timeout thresholds, and heterogeneous scenario lengths across multiple simulation runs.

7.3. Functional Traceability

Table 5. Functional Evidence in the Evaluated Runtime

Property	Observed implementation evidence	Validation status
Bounded state	All three metrics are clamped to the interval [0,100] after each action.	Confirmed
Action trace	Decision records include correctness, timeout, and response time.	Confirmed
Explainable output	Report includes final dimensions, overall score, radar view, response score, rule score, and timeline.	Confirmed
Full-score constraint	Error ceiling and accuracy cap are applied before result classification.	Confirmed
Local audio	Web Audio tones are synthesized locally; speech uses the browser speech interface.	Confirmed
Offline runtime	No external resource or programmatic network call occurs in the core HTML artifact.	Confirmed by static audit

Static inspection alone cannot establish definitive compatibility across the full spectrum of browser versions, operating system revisions, or heterogeneous device configurations; however, it provides valuable assurance that the evaluated runtime operates without reliance on external network-dependent execution paths. Complementing static analysis, a comprehensive release test protocol must include offline execution—specifically loading the artifact in airplane mode—across representative platforms: iOS, Android, Windows, and macOS. This protocol further mandates verification of all twenty defined state transitions, confirmation of audible output only after explicit user activation (to comply with modern autoplay policies), systematic evaluation across multiple viewport orientations (portrait, landscape, and dynamic resizing), and rigorous alignment between the programmatically exported decision trace and the corresponding visual report rendered on screen. Test design should integrate concrete example-based validation with robust boundary-value analysis and formal state-transition coverage to ensure thorough functional verification (As shown in Table 4).

8. Discussion

8.1. Technical Implications

The prototype demonstrates that a technically meaningful digital sand table does not require a heavy simulation backend. For training objectives centered on judgment, command, reporting, resource prioritization, and consequence awareness, a deterministic browser-local engine can provide four key properties that conventional paper-based exercises find difficult to integrate simultaneously: rapid scenario revision, repeatable execution, event-level evidence capture, and consistent scoring across multiple sessions and users. The architecture is also transparent enough for domain experts to audit thoroughly, enabling traceability of every decision point and outcome. This transparency is especially critical in high-stakes training environments, where assessment validity and defensibility are essential requirements; an opaque learned score—lacking clear causal linkage to observable actions—would undermine trust, hinder instructional feedback, and complicate validation against established competency standards [7]. Furthermore, the system's deterministic nature ensures reproducibility under identical inputs, supporting rigorous pedagogical evaluation and iterative refinement of both scenarios and performance metrics.

The design is best understood as a scenario execution substrate—a foundational framework that supports diverse interaction modalities without compromising architectural coherence. It can accommodate discrete choices, temporal ordering of actions, drag-and-drop resource allocation, map-based exploration with geospatial context, structured data-entry reporting, or continuous controls such as a frequency-stabilization slider. Critically, all these interactions emit the same normalized action event format and pass through a unified transition logic, logging infrastructure, and attribution pipeline. As a result, extending the interaction vocabulary—whether to incorporate new sensor inputs, collaborative multi-user triggers, or adaptive difficulty modulation—does not necessitate replacing or overhauling the core state model. Instead, enhancements remain modular and interoperable, preserving system integrity while enabling scalable pedagogical innovation aligned with evolving operational requirements.

8.2. Limitations

Five principal limitations warrant careful consideration in the interpretation and future development of this system. First, the current technical evaluation focuses exclusively on functional fidelity and interface responsiveness, without empirically assessing pedagogical effectiveness—specifically, learning gain over time, long-term knowledge retention, transfer of acquired competencies to real-world operational tasks, or the impact on instructor preparation and facilitation effort. A rigorous follow-up investigation should therefore adopt a pre-registered, randomized controlled design that systematically contrasts the digital sand table against established instructional modalities, including traditional lecture delivery, facilitated group discussion, and physical paper-based sand table exercises. Second, the underlying editor schema enforces a rigid structure by fixing exactly two decision options and three quantitative metrics per node, thereby constraining support for multi-layered branching logic, conditional feedback loops, and interactive elements that do not rely on discrete choice selection. Third, the error-scoring mechanism applies a uniform five-point decrement across all error types, yet domain-specific requirements suggest that safety-critical misjudgments and communication breakdowns may necessitate differentiated weighting schemes to reflect their distinct operational consequences. Fourth, browser-based speech synthesis exhibits notable variability due to differences in operating system architecture, availability and quality of installed text-to-speech voices, device mute settings, and platform-specific user-activation policies governing audio initiation. Fifth, runtime trace data are currently retained solely in volatile page memory, which precludes persistent cross-session analysis; scalable, multi-user analytics would require integration with a formally governed, access-controlled storage infrastructure or secure, digitally signed local export functionality.

Furthermore, the offline-first deployment model introduces non-trivial governance challenges related to version control, integrity assurance, and accountability. Once distributed, a scenario file may continue executing indefinitely—even after its content has become obsolete, deprecated, or superseded by updated safety protocols or pedagogical standards. To mitigate these risks, every distributed scenario must be accompanied by unambiguous version identifiers, cryptographically verifiable content hashes, prominent expiry warnings with configurable thresholds, clearly designated review ownership responsibilities, and a digitally signed release manifest that attests to compliance with current institutional and regulatory expectations. Equally important is comprehensive accessibility validation aligned with internationally recognized web standards: this includes verification of logical focus navigation order, full keyboard operability without reliance on pointing devices, sufficient color contrast ratios for text and interface elements, responsive text scaling behavior across viewport sizes, provision of synchronized captions or verbatim transcripts for all spoken audio content, and implementation of alternative interaction pathways for users who cannot perform drag-and-drop operations due to motor or perceptual constraints.

8.3. Future Work

The next engineering stage will replace the fixed editor model with a versioned, extensible schema that supports strongly typed interaction nodes, conditional branching graphs with configurable exit logic, severity-weighted behavioral penalties calibrated to domain-specific risk profiles, modular and reusable rule libraries enabling cross-application consistency, and full internationalization support including bidirectional text handling and locale-aware formatting. A dedicated graph validator will perform static analysis to detect unreachable states, cycles lacking well-defined termination conditions, missing or ambiguous terminal states, and inconsistent references to performance metrics across workflow definitions. A portable, deterministic trace format—designed for reproducibility—will facilitate exact replay of user-system interactions and robust regression testing across environment updates. For secure deployment, the runtime package must embed a cryptographic digest and provide an offline-capable verification interface to ensure integrity without network dependency. Evaluation protocols should prioritize objective, multi-dimensional metrics: decision accuracy against ground-truth benchmarks, end-to-end response latency under varying load conditions, explanatory fidelity and coherence in post-event justifications, delayed retention of learned patterns after extended intervals, and generalization performance on structurally novel, previously unencountered scenarios—moving decisively beyond subjective satisfaction surveys as the sole indicator of efficacy [9].

9. Conclusion

This paper presented and technically validated an offline-first, configurable digital sand table designed for scenario-based operational readiness training in power-system contexts. Its core architectural principle is the formalization of training exercises as finite-state systems, where behavior is fully determined by declaratively encoded scenario data. The instructor-facing editing environment explicitly separates narrative progression, actionable decision points, conditional consequences, domain-specific rule bases, and quantitative metric transformations—thereby enabling transparent authoring and reproducible execution. During runtime, state transitions are strictly deterministic and locally executed without external dependencies; the assessment pipeline captures granular, timestamped action sequences to support evidence-based evaluation. A built-in irreversible error ceiling enforces fidelity to real-world consequence logic: once a critical misstep occurs, attainment of a perfect final score is precluded by design. Static analysis of the anti-icing runtime—totaling 190,728 bytes—confirmed the absence of external scripts, third-party stylesheets, or programmatic network calls; boundary testing further verified the integrity of the scoring invariant across all defined failure modes. As such, the prototype delivers a lightweight, self-contained, and auditable technical foundation.

Future work centers on methodical enhancement: implementing richer multimodal interactions (e.g., drag-and-drop topology adjustments, voice-assisted command entry), introducing formal schema validation for scenario definitions using JSON Schema or equivalent, enabling cryptographically signed offline distribution to ensure content integrity, conducting systematic cross-device compatibility testing (including tablet and desktop form factors), and executing controlled empirical studies to measure learning retention and transfer to field practice.

References

1. P. Hevia-Koch, B. Wanner, and R. Kuwahata, *Electricity Grids and Secure Energy Transitions: Enhancing the Foundations of Resilient, Sustainable and Affordable Power Systems*, 2023.
2. R. Radvanovsky and A. McDougall, *Critical Infrastructure: Homeland Security and Emergency Preparedness*. CRC Press, 2023.
3. D. A. Kolb, R. E. Boyatzis, and C. Mainemelis, "Experiential learning theory: Previous research and new directions," in *Perspectives on Thinking, Learning, and Cognitive Styles*, vol. 1, no. 8, 2001, pp. 227–247.
4. O. Chernikova, N. Heitzmann, M. Stadler, D. Holzberger, T. Seidel, and F. Fischer, "Simulation-based learning in higher education: A meta-analysis," *Rev. Educ. Res.*, vol. 90, no. 4, pp. 499–541, 2020.
5. S. M. Halpin, "Does team training improve team performance. A meta," 2008.
6. D. Harel, "Statecharts: A visual formalism," *Commun. ACM*, vol. 31, no. 5, pp. 514–530, 1988.
7. T. Bray, "The JavaScript Object Notation (JSON) Data Interchange Format," RFC 7159, 2014.
8. D. R. Sanchez, A. Rueda, K. Kawasaki, S. Van Lysebetten, and D. Diaz, "Reviewing simulation technology: Implications for workplace training," *Multimodal Technol. Interact.*, vol. 7, no. 5, p. 50, 2023.
9. M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?' Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2016, pp. 1135–1144.
10. A. Hickson, *HTML, The Living Standard*. Lulu.com, 2014.
11. D. Gunning and D. W. Aha, "DARPA's explainable artificial intelligence program," *AI Mag.*, vol. 40, no. 2, pp. 44–58, 2019.
12. S. G. Chaparro, *Efectos Digitales de Audio con Web Audio API*, Ph.D. dissertation, Universitat Politècnica de València, 2015.
13. P. Nso-Mangue and S. Luján-Mora, "Global web accessibility assessment of national libraries' websites: A 2025 snapshot," *J. Librariansh. Inf. Sci.*, p. 09610006261468030, 2026.
14. Z. M. Alshamaa and N. N. Saleem, "Black box software testing techniques: A literature review," *Passer J. Basic Appl. Sci.*, vol. 7, no. 2, pp. 1001–1011, 2025.