Frontiers in Emerging Technology
ISSN *(Online)*: 3104-509X
Volume 2, Issue 5, Sept-Oct 2026, Page 1-22

Research Article Received: 12-09-2026| Accepted: 10-10-2026| Published: 27-10-2026

Evidence to Policy Assurance for Agentic AI: A Verifiable Provenance Framework for Regulated Enterprises

Aditya Krishnan *[1]

Abstract

Conventional execution traces record what an agent did, but do not by themselves establish whether the evidence supporting a consequential action was authoritative, current, and sufficient for a specified policy decision. This study develops the Evidence to Policy Assurance Graph (EPAG), a typed representation linking sources, qualified evidence, claims, policy predicates, compliance determinations, authorizations, executed actions, and observed outcomes. A design-science process drew on 38 selected sources and evaluated a constructed $250,000 funds-transfer fixture. The evaluation comprised one valid baseline (S0), five single-condition perturbations (S1–S5), and six predefined criteria: completeness, qualification, localization, determinacy, reconstruction, and separability. The baseline satisfied all six criteria under the fixture rules. The perturbations exposed separate failures involving stale ledger evidence, a non-authoritative spreadsheet, insufficient funds, a missing second approval, and an execution amount that differed from the authorization. Each failure mapped to a defined primary disposition and, where applicable, a follow-up action. The 629 AgentDojo security cases and 144 ToolEmu cases cited in the paper provide published risk context; none was rerun or evaluated with EPAG. These findings establish representational adequacy for six constructed graph instances, not measured performance in a deployed system or proof of regulatory compliance. Independent studies must test tool-gateway enforcement, decision errors, audit-path reconstruction, latency, throughput, storage costs, and adversarial behavior.

Keywords

Agentic AI, Evidence to Policy Assurance Graph (EPAG), Execution Provenance, Evidence Qualification, Policy Compliance, Continuous Assurance.

1 Independent Researcher, USA

1.INTRODUCTION

Agentic artificial intelligence systems move beyond traditional generative AI, which primarily generates content, to systems that can plan multistep tasks, retrieve information, use memory, invoke tools, and change enterprise state. Such actions can affect accounts, records, communications, and transactions. As published external context, AgentDojo reports 97 realistic user tasks and 629 security test cases across e-banking, e-mail, and travel workflows [18]. ToolEmu evaluates 144 high-stakes cases across 36 high-stakes tools and reports that even the safest agent in its evaluation exhibited identified failures in 23.9% of cases [19]. SafeAgent examines automated risk simulation for multi-tool agent settings [20]. These are results from their respective studies, not EPAG experiments. For a regulated enterprise, the pertinent assurance question is whether a consequential agent action relied on qualifying evidence, satisfied the applicable policy conditions, and remained within delegated authority.

The study is situated in a governance-to-action gap. A conventional trace can show that an agent read a balance, checked a beneficiary, requested approval, and called a payment tool. That sequence alone does not establish whether the balance came from the authoritative ledger, was no more than five minutes old under the fixture policy, covered the proposed transfer, or whether two valid approvals authorized the same amount and beneficiary. Article 12(1) of Regulation (EU) 2024/1689 requires high-risk AI systems to have the technical capability to record events automatically throughout their lifetime; Article 12(2) links those capabilities to traceability and specified monitoring

***Corresponding Author:** Aditya Krishnan
DOI: 10.65796/fet.vol.2.issue.2.1
© The Author(s) 2025, This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC-BY-NC)

purposes [28]. The provision does not certify EPAG or make every enterprise agent a high-risk system. EPAG instead links event records to the evidence, policy, authorization, execution, and outcome relationships needed to inspect an individual action's justification.

The study has four objectives. It first develops a typed graph to capture a justified path from authoritative source to observed result, that is, between support of the evidence, policy satisfaction, authorization, and conformity of execution. Second, it checks evidence validity through traceability, authority, relevance, and temporal validity. Third, it assesses EPAG against six stated criteria using one controlled baseline and five single-condition perturbations. Fourth, it contextualizes the relevance of EPAG by discussing its relation to agent-safety criteria, runtime checks, regulatory record-keeping requirements, and deployment architecture. The paper asks how qualified evidence can be linked to compliance and authorization, what evidence properties are required before a predicate is satisfied, and how EPAG localizes stale evidence, non-authoritative evidence, predicate failure, missing authorization, and execution deviation.

The unit of analysis is one consequential agent action that changes enterprise state. The study does not establish general agent alignment, automated legal interpretation, regulatory certification, or coverage of model-training provenance. It contributes a formal vocabulary for provenance, policy, and authorization; separates four evidence-qualification properties; and provides a reproducible action-level evaluation protocol. It claims representational adequacy in constructed scenarios, not effectiveness in production.

This study is organized into six sections. Section 2 discusses provenance, grounding, runtime governance, continuous assurance, assurance cases, regulation, visibility, and safety testing, among other concepts. Section 3 reports the design-science methodology, the six constructed graphs, and the published benchmarks used only as context. Section 4 outlines the EPAG and funds-transfer fixture. Section 5 discusses the controlled results and makes connections to previous literature. Section 6 discusses contributions, limitations, and conclusions.

2. LITERATURE REVIEW

2.1 Execution Tracing and Provenance

Agentic AI combines planning, retrieval, memory, and tool use across sequences of actions, so accountability cannot stop at inspection of the final output. Sapkota et al. (2026) distinguish AI agents from broader agentic architectures, while Abou Ali et al. (2026) survey the reasoning, memory, perception, and tool-interaction components of such systems [1; 2]. Wang et al. (2026) survey evidence tracing and execution provenance in LLM agents [23]. The W3C PROV ontology provides interoperable concepts for entities, activities, agents, derivation, attribution, association, and delegation [4]. These resources support reconstruction of information lineage. They do not, on their own, require a separate finding that specific evidence qualified to satisfy a stated policy predicate. AgentSpec, by contrast, addresses rule-based runtime enforcement rather than introducing the provenance model attributed to it in an earlier description [3].

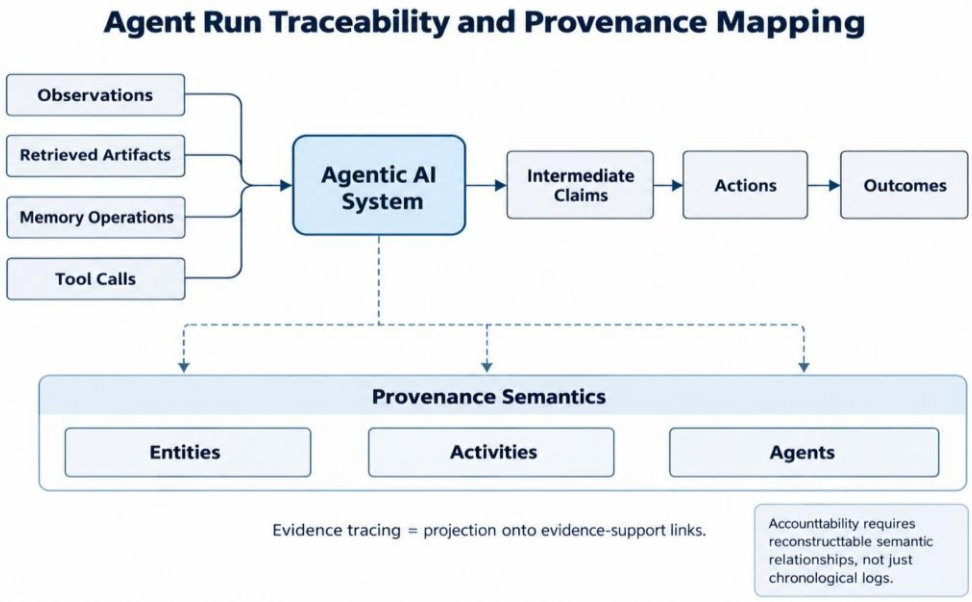

Figure 1: Agent run traceability and provenance mapping across autonomous workflows

Figure 1 illustrates the distinctions among observations, retrieved artifacts and memory operations, and callable tools, which together produce intermediate claims, actions, and outcomes for the agentic AI system. The provenance layer represents these activities as entities, activities, and agents, along with evidence-support relationships. This structure helps reconstruct semantic relationships for accountability, not just chronological logs.

2.2 Evidence Grounding and Action Validation

Evidence grounding tests whether an intended agent action is supported by information available in the agent's execution context. She et al. (2026) propose ProvenanceGuard, a provenance-based safeguard that assesses proposed tool calls before execution [6]. Their published experiments report reductions in misaligned-trace error rates from 42.9% to 1.8% on Agent SafetyBench and from 32.1% to 17.3% on WorkBench. These are results of ProvenanceGuard, not results obtained by EPAG. EPAG separates a supported claim from a satisfied policy predicate: a correct retrieved balance can still be unfit for approval if it comes from an unauthorized source, falls outside a five-minute freshness window, or fails a required funds threshold.

2.3 Runtime Policy Governance and Privilege Control

With runtime governance, the execution path is an active control model. Kaptein et al. (2026) model policy evaluation based on agent identity, a fragment of the execution path, the proposed action, and the organizational state, and they show that prompting unrelated to the organizational state cannot address path-dependent behavior [7]. Joshi et al. (2026) further extend this Control Vocabulary to include permissions, prohibitions, obligations, dispensations, conflict resolution, and ontology-based reasoning, which occurs outside the language model [8].

In practice, this lets enterprise policy manage both tool invocation and agent interactions. EPAG retains the evidentiary basis for each predicate result and the authority under which the resulting action is permitted. Table 1 shows the key runtime governance controls, their policy evaluation inputs, their

permissions, prohibitions, obligations, dispensations, and ontology-based reasoning. It also shows how EPAG enhances these controls by recording predicate outcomes and authorization decisions.

Table 1: Runtime policy governance controls and privilege assurance mechanisms

Aspect	Key Data	Assurance Relevance
Runtime governance focus	Execution path treated as an active control object	Enables controls to respond to changing agent behavior during execution
Policy evaluation inputs	Agent identity, partial execution path, proposed action, organizational state	Supports context-aware and path-dependent policy decisions
Control limitations addressed	Prompt instructions and static access controls are insufficient for dynamic behavior	Justifies runtime rather than purely preconfigured enforcement
Policy vocabulary	Permissions, prohibitions, obligations, dispensations, and conflict resolution	Provides a richer basis for governing agent actions
Reasoning mechanism	Ontology-based policy reasoning outside the language model	Separates policy enforcement from model-generated reasoning
Governed interactions	Tool invocations and agent-to-agent interactions	Extends controls across enterprise agent workflows
EPAG contribution	Preserves evidence behind predicate results and granted authorization scope	Strengthens auditability, traceability, and justification of runtime decisions

2.4 Continuous Assurance and Evidence Synthesis

Continuous assurance channels governance from periodic documentation to evidence that can be evaluated as systems and processes change. In Trustworthy AI Posture, Lupo et al. (2026) introduce a pathway built on ontologies to correlate regulatory obligations, assurance claims, controls, and evidence verifiable across claims, systems, organizations, and ecosystems [9]. Taking this a step further, Muhammad et al. (2026) transform governance requirements into versioned, executable policy checks via Audit-as-Code for reproducible technical decisions [10]. A governance-to-action gap remains: a general permission or policy model does not necessarily document the evidence and authorization supporting an individual state-changing action [11]. EPAG aims to fill that gap by connecting action to policy.

2.5 Regulatory and Assurance Context

The regulatory and assurance literature supplies context for the proposed graph, not evidence that the graph meets an external certification standard. The NIST AI Risk Management Framework organizes risk management around GOVERN, MAP, MEASURE, and MANAGE functions [12], while its Generative AI Profile addresses risks specific to generative systems [25]. ISO/IEC/IEEE 15026-2:2022 addresses assurance cases [13]. Hevner et al. (2004) and Peffers et al. (2007) supply the design-science basis for constructing and evaluating an information-system artifact [14; 15]. For record-keeping, Article 12 of the EU AI Act concerns automatic logging capabilities for systems within its high-risk scope [28]. None

of these sources establishes that the six EPAG fixture scenarios demonstrate legal compliance or production effectiveness. The separate manufacturing-regression study concerns predictive optimization rather than evidence provenance or action authorization [16].

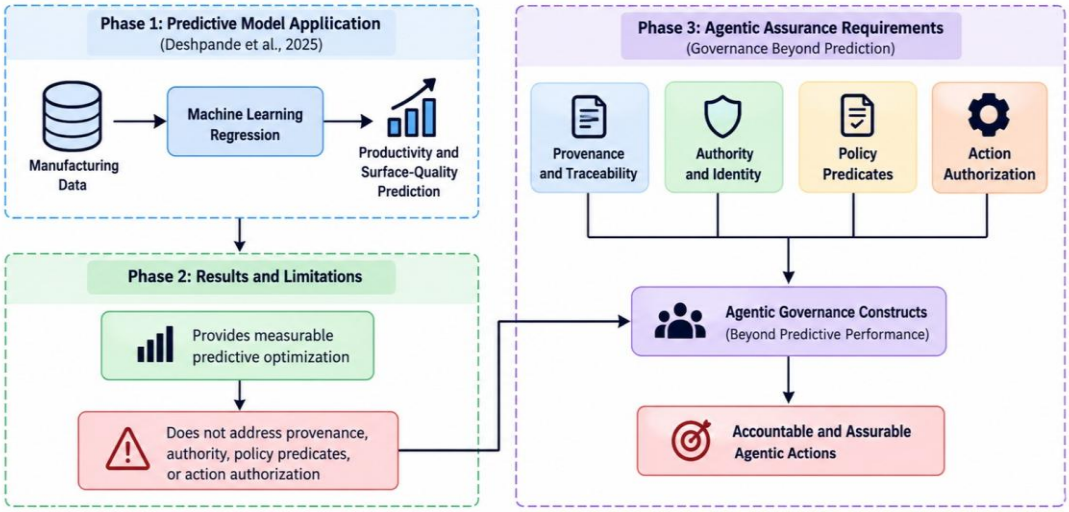

Figure 2: Predictive optimization limitations and governance requirements for agentic assurance

Figure 2 shows a gradient from lower-level predictive machine learning to broader agentic assurance systems. The manufacturing regression example illustrates the optimization of surface-quality and productivity measures, not the assessment of action-level assurance. However, predictive performance alone is insufficient because of provenance, authority, policy predicates, and action authorization. The framework thus incorporates governance mechanisms to support traceable, accountable, and assurable agentic actions.

2.6 Agent Visibility, Risk Testing, and Safety Benchmarks

Published agent-safety studies provide context for the need to inspect consequential tool actions. AgentDojo includes 629 security test cases and 97 user tasks [18]; ToolEmu includes 144 high-stakes cases across 36 high-stakes tools [19]. Together, the 773 published cases are secondary evidence about risk in agent workflows. This study did not access their case-level outputs for EPAG scoring, rerun either benchmark, or treat those 773 cases as EPAG graph instances. SafeAgent supplies additional context on agent risk simulation [20]. The present evaluation consists only of the six constructed EPAG graphs S0–S5.

The unresolved synthesis issue is combining provenance and evidence fitness, policy evaluation, authorization, execution conformity, and backward reconstruction of evidence and its history into a single inspectable assurance representation. For a regulated workflow, the assurance record must identify the evidence source, decision time, policy version, approving identity, action parameters, and observed result. These fields permit review of stale evidence, failed predicates, and execution outside the approved scope.

3. METHODS AND TECHNIQUES

3.1 Data Collection Methods

3.1.1 Research Design

Because the study developed an assurance artifact, the research used a design science

orientation. The artifact comprised the typed Evidence to Policy Assurance Graph, evidence-qualification function, authorization invariant, typed relations, and configured failure dispositions [17]. The research process followed stages of problem identification, objective definition, artifact construction, demonstration, evaluation, and communication to capture the essence of design science for constructing and evaluating information system artifacts. The unit of analysis was the justification chain for a single consequential agent action. This process captured the structured design science methodology outlined by Peffers et al. (2007), which follows a progression from problem definition to demonstration and evaluation [15].

Table 2: Data collection methods, analytical layers, sampling, and study roles

Component	Sample / Unit	Data or Material Collected	Collection / Sampling Method	Analytical Role in the Study
Research design	One consequential agent action and its complete justification chain	Typed EPAG, evidence qualification function, authorization invariant, typed graph relations, and controlled failure dispositions	Design science sequence involving problem identification, objective definition, artifact construction, demonstration, evaluation, and communication	Supported the construction and systematic assessment of an assurance artifact rather than a predictive model
Layer A: Literature corpus	**n = 38 sources**	Agent taxonomies, execution provenance, provenance safeguards, runtime governance, deontic policy, continuous assurance, regulatory guidance, design science, assurance cases, agent visibility, risk simulation, prompt injection evaluation, and privilege control	Criterion based purposive sampling from stable publisher, proceedings, standards, legal, and repository records	Supplied the theoretical, technical, regulatory, and methodological constructs required to develop EPAG
Layer A: Selection criteria	38 retained primary or traceable sources	Sources addressing at least one EPAG construct and having a stable and verifiable publication record	Included traceable primary sources; excluded marketing material, unsourced commentary, duplicate versions,	Improved source relevance and ensured that the literature base addressed assurance, governance, provenance, and

			and studies concerned only with task success	control requirements
Layer B: EPAG graph instances	**n = 6 graphs**	Baseline **S0** and five controlled perturbations **S1-S5**	Controlled scenario construction using one baseline followed by single-condition changes	Enabled structured testing of evidence, policy, authorization, and execution failures within the EPAG representation
Layer C: Security and safety evidence	**n = 773 published cases**	**629 AgentDojo security cases** and **144 ToolEmu test cases**, with SafeAgent providing additional methodological context	Secondary empirical context drawn from published agent security and safety evaluations	Demonstrated the practical relevance of prompt injection testing, tool-use risk simulation, and automated agent safeguarding; the 773 cases were not rerun through EPAG
Search and verification period	Literature collection process	Selected publications and supporting records across the defined source categories	Literature search, source verification, and eligibility screening completed on **19 September 2026**	Established a defined collection cutoff and documented the temporal boundary of the study's evidence base

Table 2 presents the study's research design, analytical layers, sampling procedures, and data sources. It also summarizes the roles of a 38-source literature corpus, six instances of EPAG graphs, and 773 published security and safety cases.

3.1.2 Primary Data Sources and Sampling

The study used three analytical layers. Layer A comprised 38 purposively selected sources covering agent architectures, execution provenance, provenance safeguards, runtime governance, deontic policy, continuous assurance, regulatory guidance, design science, assurance cases, agent visibility, risk simulation, prompt injection evaluation, and privilege control. A source qualified for inclusion when it contributed a relevant EPAG construct and could be traced to a publisher, conference proceeding, standard, legal text, or identifiable research repository. The screening excluded marketing material, unsourced commentary, duplicate versions, and studies concerned only with task-completion outcomes. Literature selection and verification closed on 19 September 2026.

For Layer B, the study constructed six EPAG graph instances: baseline S0 and five single-condition perturbations S1–S5. This is the entire set evaluated against criteria C1–C6. Layer C supplied published context from 773 security or safety cases: 629 AgentDojo security cases and 144 ToolEmu cases [18; 19]. AgentDojo benchmarks agent tasks and prompt-injection attacks; ToolEmu examines

risks using simulated tool environments. SafeAgent contributes additional published methodological context [20]. The 773 cases were neither rerun nor independently scored using EPAG, and they must not be included in the denominator for the six-graph criterion results.

3.2 Data Analysis and Concept Coding

The researcher coded the selected sources for seven analytical questions: what object was involved, which relation was claimed, how evidence quality was established, what policy condition applied, who held authority, which execution constraints applied, and which audit question could be reconstructed. The recording fields were Object, Actor, Input, Relation, Condition, Decision, Output, Timing, Version, and Supported Audit Question [21]. A deductive pass applied these fields to the literature; a second pass captured source-specific constructs such as delegation, dispensation, and privilege scope [22]. The researcher then checked whether the coded concepts could support backward reconstruction of policy version, evidence age, approving identity, and requested versus executed parameters. One researcher performed source selection and coding. No second coder independently annotated the sources; therefore, inter-rater agreement was neither measured nor established. This limits the reliability claims that can be drawn from the coding process.

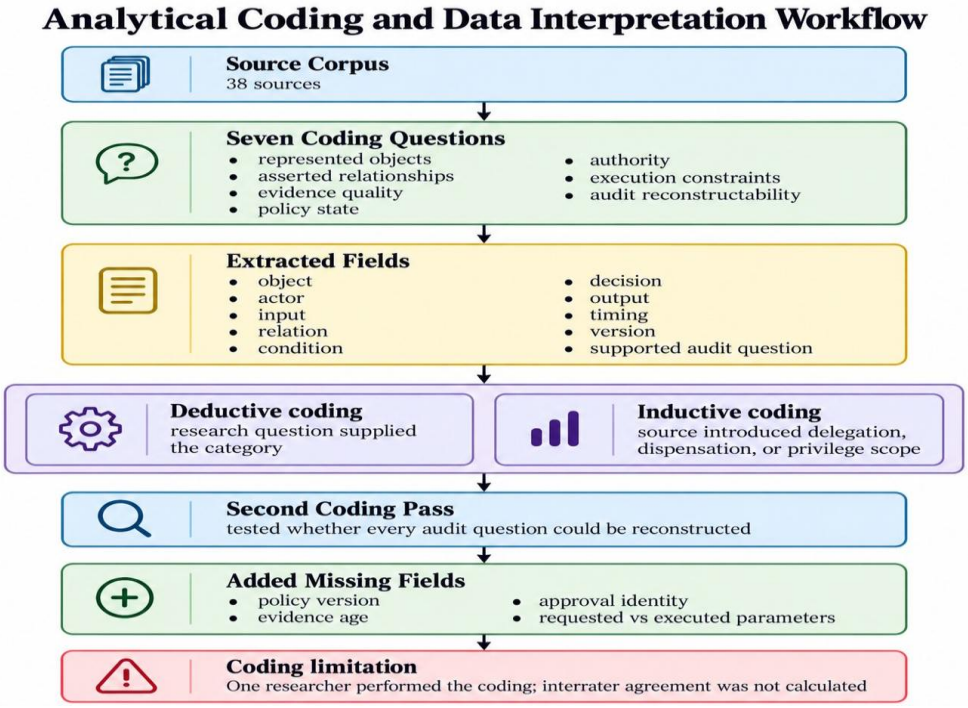

Figure 3: Concept coding workflow for audit reconstruction and data validation

Figure 3 shows the two-stage process of concept coding, normalization, validation, and completion. The first stage involves coding 38 literature sources related to the seven analytical questions, using structured field coding and both deductive and inductive coding. The second stage verifies whether the coding can reconstruct the audit questions and uncover missing information such as policy version, evidence age, approving entity, and differences between requested and executed parameters. Additionally, as shown in the workflow, one researcher conducted the coding, so inter-rater agreement was not computed.

3.3 Scenario Design, Failure Injection, and Analytical Procedure

The study modelled baseline S0 as a $250,000 transfer to an approved beneficiary in a constructed enterprise payment fixture. Its policy required evidence from the authoritative core ledger no older than five minutes, available funds of at least $250,000, an approved beneficiary-registry entry, and two valid approvals for transfers of $100,000 or more. The baseline contained a $400,000 available balance, current ledger and registry evidence, two approvals covering the intended action, and execution parameters matching the authorized amount, account, beneficiary, tool, and validity period. These values define the fixture; they are not observations from a deployed banking system.

The researcher then applied five single-condition perturbations to S0. S1 made the ledger evidence older than the five-minute limit. S2 replaced the authoritative ledger record with a user-maintained spreadsheet. S3 reduced available funds below $250,000. S4 removed the second required approval. S5 changed the executed transfer amount after the authorization had been recorded. Table 3 gives the controlled change and expected break for each graph. The configured responses were HOLD with REFRESH for S1, BLOCK with RETRIEVE for S2, BLOCK for S3, HOLD with ESCALATE for S4, and BLOCK with ALERT and REMEDIATE for S5. In each response pair or group, the first term is the primary disposition; the remaining terms describe follow-up actions. For an already executed S5 deviation, BLOCK cannot reverse the transfer and instead refers to preventing further processing while alert and remediation procedures address the discrepancy.

Table 3: Controlled scenario changes and corresponding expected EPAG assurance effects

Scenario	Controlled Change	Expected Assurance Effect
S0 — Baseline	$250,000 transfer supported by a $400,000 authoritative ledger balance, current ledger and beneficiary-registry evidence, two valid approvals, and execution matching the authorized amount, beneficiary, account, and validity period	Complete assurance chain with **ALLOW** decision
S1 — Stale Evidence	Core-ledger evidence exceeds the required five-minute freshness window	Temporal validity fails, producing **HOLD and REFRESH**
S2 — Non-authoritative Evidence	Authoritative core-ledger evidence is replaced with a spreadsheet	Evidence authority fails, producing **BLOCK and RETRIEVE** from an approved source
S3 — Insufficient Funds	Available funds are reduced below the requested **$250,000** transfer amount	Required funds predicate is unsatisfied, producing **BLOCK**
S4 — Missing Approval	The second required approval for a transfer above **$100,000** is removed	Authorization becomes incomplete, producing **HOLD and ESCALATE**
S5 — Execution Deviation	The executed transfer amount differs from the amount previously authorized	Execution exceeds authorization, producing

	BLOCK, ALERT, and REMEDIATE

3.4 Evaluation Criteria, Decision Rules, Validity, and Reproducibility

The study assessed all six EPAG instances with the same predetermined criteria, C1–C6 (Table 4). For C1, the researcher checked whether the eight object classes, their required typed relations, and an explicit state for the relevant link or failure were represented; S0 supplied the complete valid path, while S1–S5 retained an inspectable location for the injected break. For C2, the researcher inspected separate values or supporting records for traceability, source authority, predicate relevance, and temporal validity. S0 satisfied those four conditions. S1 specifically failed temporal validity, S2 failed source authority, and S3–S5 left evidence qualification intact while introducing later failures. For C3, the expected location was compared with the graph state: no break in S0; the evidence-validity property in S1; source authority in S2; funds predicate in S3; approval/authorization in S4; and the authorization-to-action conformity check in S5.

C4 tested whether the fixture rules yielded one primary disposition per case: ALLOW (S0), HOLD (S1), BLOCK (S2), BLOCK (S3), HOLD (S4), or BLOCK (S5). REFRESH, RETRIEVE, ESCALATE, ALERT, and REMEDIATE were follow-up actions, not additional primary decisions. C5 required a backward traversal from the recorded action, attempted action, or outcome to the applicable authorization state, compliance result, predicate, claim, evidence, and source; an unsuccessful case had to retain the failed link without representing it as satisfied. C6 checked that the five perturbations remained distinguishable as temporal-validity, source-authority, policy-predicate, approval, or execution-conformity failures. The assessment recorded qualitative pass conditions and expected dispositions for these six constructed instances. No independent adjudicators, production transaction logs, or quantitative error measurements were used.

Table 4: EPAG evaluation criteria, pass conditions, and decision rule mappings

Criterion	Evaluation Focus	Pass Condition
C1 — Completeness	Verifies that all required EPAG objects and typed relations are explicitly represented	No required object, relation, or assurance link is missing or left implicit
C2 — Qualification	Tests whether evidence qualification properties are independently inspectable	Traceability, authority, relevance, and temporal validity are each recorded together with their evaluation basis
C3 — Localization	Determines whether an injected failure can be located at the appropriate graph element	Each perturbation maps to its expected node, property, or typed relation
C4 — Determinacy	Tests whether the assurance state produces an unambiguous operational response	Each scenario produces exactly one primary disposition under the predefined fixture rules

C5 — Reconstruction	Tests whether the assurance chain can be traversed backward from action or outcome	Reverse traversal reconstructs the complete decision path to policy, evidence, and source while preserving the relevant historical context
C6 — Separability	Determines whether different classes of assurance failure remain distinguishable	Evidence, policy, approval, authorization, and execution failures are represented separately rather than collapsed into one generic failure state

Table 4 defines the pass conditions for C1–C6; Table 3 identifies the controlled input change, and Table 6 records the corresponding observed graph break and disposition. Read together, the tables provide a scenario-to-criterion audit trail: S0 is the no-failure control, S1–S2 exercise evidence qualification, S3 tests predicate satisfaction, S4 tests the approval requirement, and S5 tests execution conformity. Completeness, deterministic disposition, reconstruction, and failure separability were inspected on every graph, including the invalid ones. Figure 4 shows the eight-stage assessment workflow. Because the same researcher defined the framework and judged the constructed fixtures against its design criteria, this test can be partly circular. The six cases demonstrate internal representational consistency, not independent validation of EPAG's effectiveness.

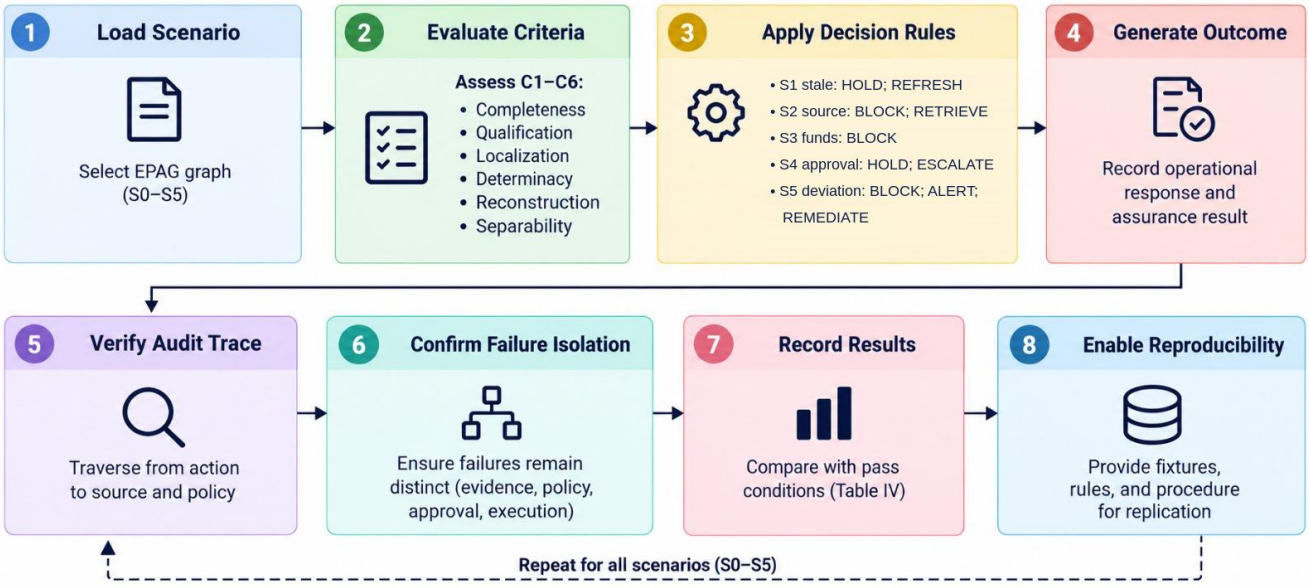

Figure 4: Eight-stage EPAG evaluation workflow supporting validity and reproducibility

4. EVIDENCE TO POLICY ASSURANCE FRAMEWORK

4.1 Problem Formulation and EPAG Objects

The Evidence to Policy Assurance Graph was formulated as a typed graph,

$$G_{EPAG} = (V, R)$$

for representing the justification of one agent action. Its vertex set has eight classes of

objects: Source, Evidence, Claim, Policy Predicate, Compliance Determination, Authorization, Action, and Outcome. Typed relationships included creating evidence from a source, supporting evidence with a claim, a claim satisfying or failing a policy predicate, predicates contributing to a compliance determination, a compliance result authorizing an action, an authorization constraining an action, and an action producing an outcome.

Each object carried an identifier, relevant event time and version, responsible actor, and correlation fields so that a later audit could reconstruct the policy and evidence state used for the earlier decision [24]. This design is consistent with the NIST AI Risk Management Framework's GOVERN, MAP, MEASURE, and MANAGE risk-management functions [12], but it does not establish conformity with that framework. Table 5 identifies the eight EPAG object classes and the minimum information recorded for traceability, authorization checks, and backward audit reconstruction.

Table 5: EPAG objects and required assurance information for audit reconstruction

EPAG Object	Assurance Role	Required Assurance Information
Source	Origin of evidence	System owner, source type, trust designation, version
Evidence	Observed artifact or value	Value, timestamp, hash, retrieval event, source reference
Claim	Policy relevant proposition	Statement, confidence or proof status, supporting evidence
Policy Predicate	Machine evaluable condition	Rule ID, version, operator, threshold, scope
Compliance Determination	Combined predicate result	Satisfied or failed predicates, evaluator, time
Authorization	Permission to execute	Approver or policy authority, scope, expiry, constraints
Action	Requested or executed operation	Tool, target, parameters, actor, correlation ID
Outcome	Observed state change	Result, status, side effects, remediation state

4.2 Evidence Qualification Gate

Evidence qualification was represented as

$$Q(e,p,t) = T(e) \wedge A(e) \wedge R(e,p) \wedge V(e,t)$$

T(e) required traceability to the originating source. A(e) required that the source had authority for the policy purpose. R(e,p) tested whether evidence e was relevant to predicate p, while V(e,t) tested whether it remained valid at decision time t. These four checks assessed evidence qualification, not predicate truth. For example, a core-ledger balance may be accurate when retrieved yet fail V(e,t) once its age exceeds the fixture's five-minute window. Qualified evidence can also support a claim that fails a funds-available predicate, as in S3. The NIST Generative AI Profile addresses AI risk management but does not prescribe or validate this EPAG gate [25].

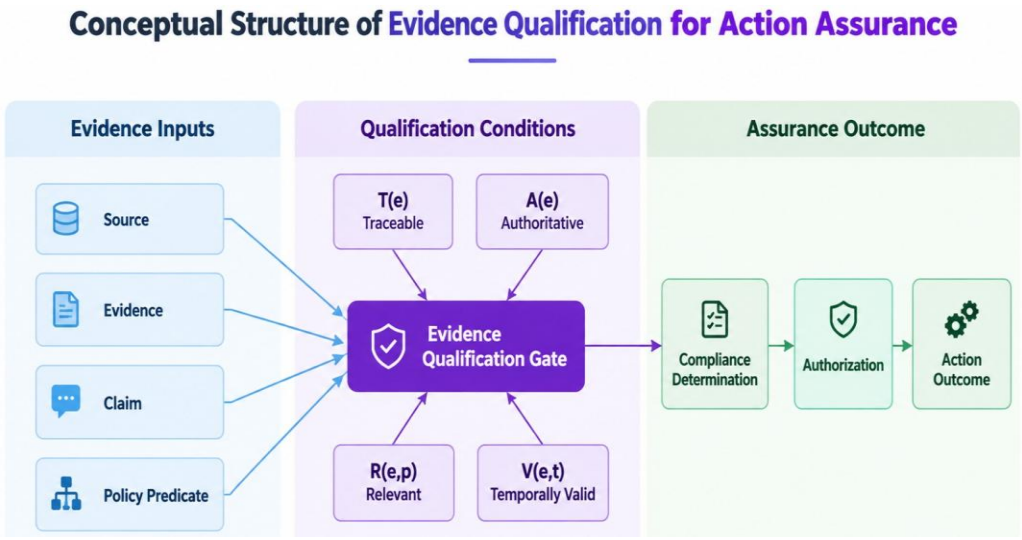

Figure 5: Evidence qualification gate linking inputs to assurance outcomes

Figure 5 depicts the evidence-qualification gate linking source records, claims, and policy predicates. It tests traceability, source authority, predicate relevance, and temporal validity before evidence can support downstream predicate evaluation. Passing the gate does not authorize an action by itself; policy predicates and authorization constraints must also be satisfied.

4.3 Authorization Invariant and Four Design Principles

Authorization was represented by

$$Allow(a,t) = CompleteAuth(z,t) \wedge (\forall p \in Req(a): Sat(p,t)) \wedge ScopeMatch(a,z)$$

Here, a denoted the action, z represented the authorization object, Req(a) contained the required policy predicates, and ScopeMatch(a,z) checked the tool, target, amount, beneficiary, timing, and other approved parameters. The invariant is conjunctive: every required predicate must be satisfied, the authorization must be valid for the action, and the execution must stay within that scope. A satisfied funds predicate cannot substitute for a missing second approval; a valid approval cannot authorize a different executed amount. P1 required explicit links between policy claims, evidence, and sources. P2 separated predicate satisfaction from evidence qualification.

P3 required the authorization record to preserve approving authority, action scope, expiry, and constraints. P4 associated the executed action and observed outcome with that authorization and its decision context. This is similar to programmable privilege control, where tool permissions are restricted by stated policies and by intent expressed in natural language [26]. ISO/IEC/IEEE assurance-case guidance relates claims, arguments, evidence, and context. EPAG supplies an action-level evidentiary record; it does not constitute a complete assurance case.

4.4 Enterprise Deployment Architecture and Human Accountability

A proposed deployment would connect the agent runtime to an evidence-qualification gateway, policy and predicate engine, authorization service, tool gateway, and enterprise system. Each component would send correlated records to an EPAG recorder and an integrity-protected graph or audit store. Required implementation controls include stable identifiers, a versioned policy registry, an authoritative-source catalog,

evidence freshness rules, identity and approval rules, scoped authorization, and protected event storage [27]. Article 12 of the EU AI Act requires automatic event-logging capability for systems classified as high-risk under that regulation; it does not establish that EPAG itself complies with the Act [28]. Figure 6 depicts the proposed architecture. This study did not deploy the components, test their enforcement behavior, or measure their operational performance.

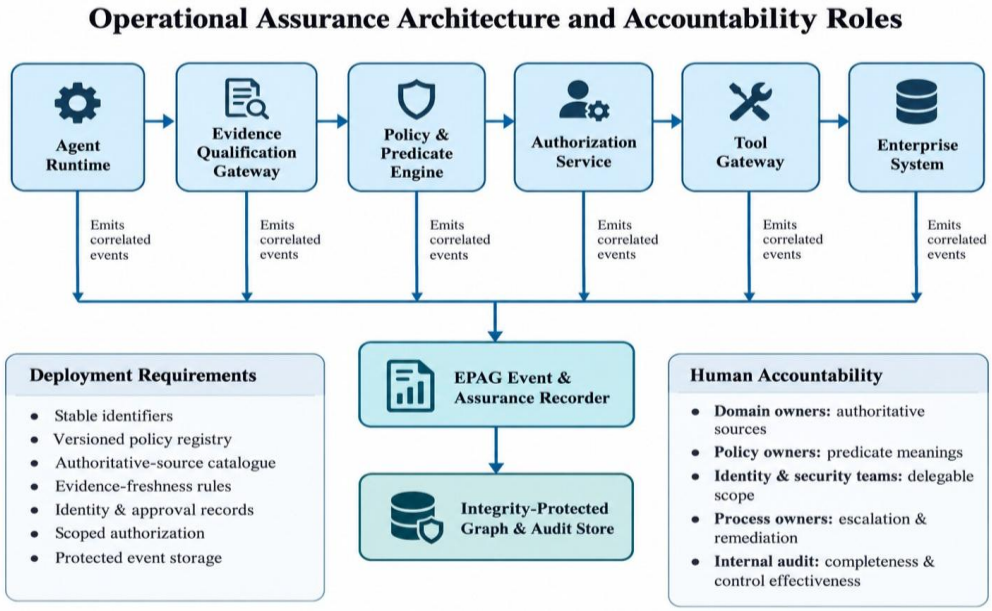

Figure 6: Operational assurance architecture and accountability across enterprise systems

Figure 6 shows these controls along the runtime path. Under the proposed architecture, identity and security teams would define delegable scope; internal audit would assess record completeness and controls; process owners would assign escalation and remediation procedures; and domain owners would identify authoritative sources. These responsibilities were not evaluated in a deployment.

5. RESULTS AND DISCUSSION

5.1 Baseline Construction and Criterion Results

Baseline S0 represented a $250,000 transfer backed by a $400,000 available balance, current records from the authoritative ledger and beneficiary registry, and two approvals covering the authorized action. This constructed case is situated in a regulated banking domain discussed in [29]. The graph linked those inputs to qualified evidence, funds and beneficiary claims, policy predicates, a compliance determination, authorization, execution, and outcome. The evidence met the five-minute freshness rule, and the executed account, beneficiary, amount, tool, and validity period matched the approval. C1 and C2 passed on required graph structure and qualification fields; C3 found no injected failure; C4 yielded the primary disposition ALLOW; C5 preserved the backward path; and C6 did not conflate any control category. The baseline thus met 6/6 predefined representational criteria. This score reports internal fixture consistency. It is not an estimate of operational accuracy, risk reduction, or regulatory conformity.

5.2 Failure Localization across Scenarios

Each negative graph exposed its injected break at a different stage. S1 failed V(e,t) because ledger evidence exceeded the

freshness window; its primary disposition was HOLD, followed by REFRESH. S2 failed A(e) because the available-balance record came from a user-maintained spreadsheet rather than the approved ledger; its primary disposition was BLOCK, followed by RETRIEVE from an authorized source. S3 retained qualified ledger evidence but failed the funds predicate and produced BLOCK. S4 removed one required approval and produced HOLD, followed by ESCALATE. S5 recorded an execution amount that differed from its authorization and produced BLOCK with ALERT and REMEDIATE. These are configured fixture decisions, not measurements of a live enforcement mechanism. Table 6 summarizes the localized breaks and their stated responses; Figure 7 places each break in the graph.

Table 6: Qualitative Scenario Results

Scenario	Localized Break	Disposition	Audit Conclusion
S0	None	**ALLOW**	Complete assurance chain with no introduced break
S1	\(V(e,t)=false\)	**HOLD AND REFRESH**	Evidence failed temporal validity because the ledger value was stale
S2	\(A(e)=false\)	**BLOCK AND RETRIEVE**	Evidence failed source authority and required retrieval from an approved source
S3	Predicate unsatisfied	**BLOCK**	Qualified evidence was present, but the required funds predicate failed
S4	Authorization incomplete	**HOLD AND ESCALATE**	Required approval was missing, preventing complete authorization
S5	Action exceeds authorization	**BLOCK, ALERT, AND REMEDIATE**	Executed parameters deviated from the previously authorized action

Figure 7 locates the five controlled assurance breaks in the EPAG action chain. S1 and S2 affect evidence qualification, S3 affects a funds policy predicate, S4 affects the authorization condition, and S5 affects conformity between authorization and execution. For S1–S5 the figure shows a primary response plus any prescribed follow-up. In the post-authorization S5 fixture, the graph documents a deviation and the requirement to block further processing, alert the responsible function, and begin remediation; the figure does not demonstrate that EPAG prevented an already completed transfer.

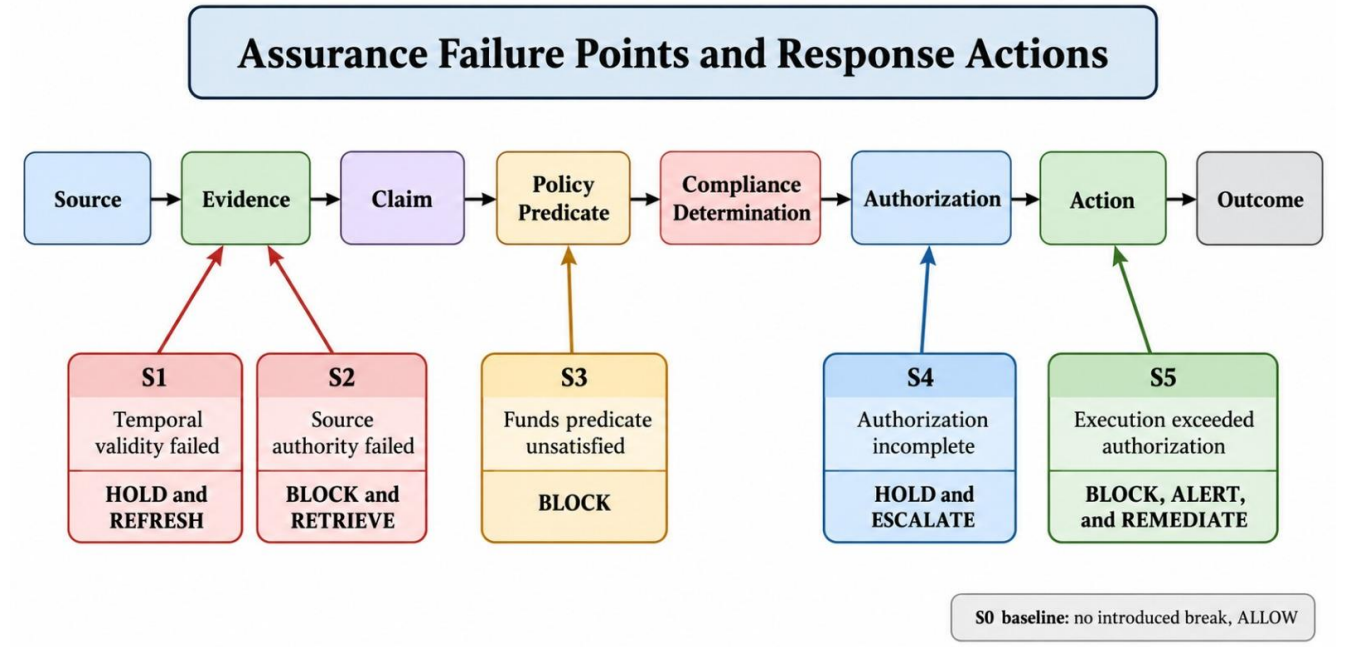

Figure 7: Failure localization and corresponding operational disposition

5.3 Cross-Scenario Synthesis and Backward Reconstruction

Across S0–S5, C1 checked the presence of the required object and relation structure, including an explicit representation of each failed condition. C2 showed why S1 and S2 failed evidence qualification while S3–S5 retained qualified evidence. C3 localized each injected condition to its expected graph property or link. C4 produced the predefined primary response for every fixture. C5 preserved a backward path from the attempted or recorded action to the available decision history, and C6 distinguished evidence, predicate, approval, and execution-conformity failures. These results follow from the constructed fixtures and the chosen rules. Work on bounded autonomy in enterprise agent workflows provides related policy context [30], but EPAG has not been tested against independently labeled transactions or a live audit system.

Visibility of an agent's identity, capabilities, inputs, outputs, actions, and consequences supports oversight [31]. Figure 8 depicts the backward reconstruction route from a recorded outcome or attempted action through the authorization state, compliance determination, predicate, claim, evidence, and source. An invalid case retains the location and reason for the failed check rather than presenting a satisfied chain. The figure establishes that these relationships can be represented and traversed within the fixture. It does not show that deployed event collectors always capture complete, authentic records.

Figure 8: Backward audit reconstruction from outcomes to authoritative sources

5.4 Comparison of Findings with the Existing Literature

The constructed scenarios distinguish execution provenance from action-level assurance. A trace records information lineage; EPAG also records whether the evidence qualifies for a specific policy purpose and whether the executed parameters conform to authorization. Provenance-based validation tests support for a proposed action. Runtime governance evaluates permissions during execution. EPAG retains the evidentiary basis and outcome of those checks for a single consequential action. Ikhwantri et al. (2026) likewise examine claims, evidence, and evaluation in assurance-case generation for Cyber Resilience Act requirements [32]; that work does not validate EPAG.

Research on fintech data architecture emphasizes secure records for real-time decisions [33], while work on physical identity and access management examines authorization boundaries in critical infrastructure [34]. These studies provide application context, not independent EPAG test results. Table 7 compares the primary representational focus of related approaches across trace lineage, evidence qualification, policy predicates, and authorization-to-execution linkage. EPAG makes all four relationships explicit in the proposed graph.

Table 7: Primary Representational Emphasis of Related Approaches

Approach	Trace	Evidence Qualification	Policy Predicate Link	Authorization and Execution Link
Execution provenance	Primary	Variable	Not central	Variable
Provenance-based validation	Supported	Partial	Not central	Pre-execution focus
Runtime policy governance	Supported	Conditional	Primary	Primary for enforcement
Continuous assurance	Supported	Primary	Supported	Variable or lifecycle-level
EPAG	**Explicit**	**Explicit**	**Explicit**	**Explicit**

Frontiers in Emerging Technology
ISSN *(Online)*: 3104-509X
Volume 2, Issue 5, Sept-Oct 2026, Page 1-22
Research Article
Received: 12-09-2026| Accepted: 10-10-2026| Published: 27-10-2026

5.5 Future Research Recommendation

Future evaluation should implement EPAG at a tool gateway and policy-enforcement point with policies written independently of the assessment criteria, alongside a trace-only comparison. The experiment should include externally labeled decisions and measure false-positive and false-negative dispositions, failure-localization accuracy, backward audit-path completeness, decision latency, throughput, storage overhead, and scalability under load [35]. A second stage should test prompt injection, compromised or replayed evidence, forged approval identifiers, policy-version conflicts, multi-agent delegation, privilege escalation, and incident recovery. These are proposed measurements and test conditions; the present study reports no values for them.

Future studies should adapt the predicates, authority rules, and evidence requirements to finance, healthcare, insurance, public services, and industrial control [36]. Enterprise decision governance involves technical controls and organizational responsibility [37; 38]. To support replication, evaluations should release scenario fixtures, independently assigned ground-truth labels, policy versions, decision logs, and error analyses where confidentiality permits. They should report structural representation separately from operational performance and legal or organizational compliance.

6. CONCLUSIONS

This paper proposes the Evidence to Policy Assurance Graph (EPAG), an action-level representation for consequential agent operations in regulated enterprise settings. EPAG connects authoritative sources, qualified evidence, claims, policy predicates, compliance determinations, authorizations, executed actions, and observed outcomes. It distinguishes evidence support from predicate satisfaction and authorization from execution conformity. The distinction answers the study's governance-to-action question: a trace that records an action does not, by itself, show that the action met the governing conditions. EPAG therefore records source authority, evidence relevance and freshness, required policy decisions, approval scope, and the relationship between the approved and executed parameters.

The controlled $250,000 transfer fixture contained a $400,000 authoritative ledger balance, approved beneficiary information, two valid approvals, and matching execution parameters. Baseline S0 met all six predefined representational criteria. The five single-condition perturbations produced different configured responses: S1, stale ledger evidence, yielded HOLD and REFRESH; S2, a non-authoritative spreadsheet, yielded BLOCK and RETRIEVE; S3, insufficient funds, yielded BLOCK; S4, a missing second approval, yielded HOLD and ESCALATE; and S5, an executed amount inconsistent with authorization, yielded BLOCK, ALERT, and REMEDIATE. In each case, the first response was the primary disposition. The graph retained a path to the recorded evidence and policy state, including the reason for a failed check. These are qualitative findings from six constructed graph instances, not evidence of successful enforcement, transaction recovery, or regulatory certification.

Several constraints limit interpretation. The researcher defined both the EPAG construct and the six evaluation criteria, creating a risk of circularity in the positive assessment. One researcher also selected and coded the sources; there was no independent recoding or inter-rater agreement estimate. The funds-transfer fixture covered one policy configuration and one-condition-at-a-time perturbations, not a representative sample of enterprise agent behavior. EPAG was not

***Corresponding Author:** Aditya Krishnan
DOI: 10.65796/fet.vol.2.issue.2.1
© The Author(s) 2025, This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC-BY-NC)

deployed at a live tool gateway, and the study did not measure latency, throughput, scalability, false-positive or false-negative dispositions, storage overhead, adversarial resistance, or recovery performance. The model also assumes trustworthy identifiers, timestamps, policy versions, source records, and approval records. It does not establish resistance to record tampering, compromised authoritative sources, or collusion among approvers. Consequently, representational adequacy in the six fixtures cannot be generalized to operational reliability or legal compliance.

EPAG's contribution is a single action-level representation of evidence qualification, policy satisfaction, authorization scope, and execution conformity. A future implementation should compare EPAG against trace-only logs using independently specified policy rules, labeled case outcomes, and reproducible decision records. Evaluations at tool gateways and policy-enforcement points should include adversarial and recovery scenarios, quantitative latency and storage measures, localization accuracy, audit-path completeness, and testing in multiple regulated domains. Until those tests are completed, EPAG remains a conceptual assurance artifact demonstrated through controlled scenarios.

REFERENCES;

1. R. Sapkota, K. I. Roumeliotis, and M. Karkee, "AI agents vs. agentic AI: A conceptual taxonomy, applications and challenges," *Information Fusion*, vol. 126, Art. no. 103599, 2026, doi: 10.1016/j.inffus.2025.103599.
2. M. Abou Ali, F. Dornaika, and J. Charafeddine, "Agentic AI: A comprehensive survey of architectures, applications, and future directions," *Artificial Intelligence Review*, vol. 59, Art. no. 11, 2026, doi: 10.1007/s10462-025-11422-4.
3. H. Wang, C. M. Poskitt, and J. Sun, "AgentSpec: Customizable runtime enforcement for safe and reliable LLM agents," in *Proc. 2026 IEEE/ACM 48th Int. Conf. Software Engineering*, Apr. 2026, pp. 2938–2950. https://dl.acm.org/doi/pdf/10.1145/3744916.3764546
4. W3C Provenance Working Group, "PROV-O: The PROV ontology," W3C Recommendation, Apr. 30, 2013. https://www.w3.org/TR/prov-o/
5. L. Waltersdorfer and M. Sabou, "Leveraging knowledge graphs for AI system auditing and transparency," *Journal of Web Semantics*, vol. 84, Art. no. 100849, 2025. https://www.sciencedirect.com/science/article/pii/S1570826824000350
6. Y. She, Y. Liang, and E. Kang, "Safeguarding LLM agents from misalignment through provenance analysis," in *Proc. 41st IEEE/ACM Int. Conf. Automated Software Engineering*, 2026, doi: 10.1145/3832783.3837534.
7. M. Kaptein, V.-J. Khan, and A. Podstavnychy, "Runtime governance for AI agents: Policies on paths," *arXiv preprint arXiv:2603.16586*, 2026, doi: 10.48550/arXiv.2603.16586.
8. Joshi, T. Finin, K. P. Joshi, and L. Kagal, "Deontic policies for runtime governance of agentic AI systems," in *Proc. IEEE Int. Conf. Web Services*, 2026, pp. 1255–1264, doi: 10.1109/ICWS72778.2026.00159.
9. G. Lupo, B. Q. Vo, and N. Locke, "Trustworthy AI posture: A framework for continuous AI assurance of agentic systems at horizontal and vertical scale," *arXiv preprint arXiv:2603.03340*, 2026.
10. E. Muhammad, K.-C. Yow, and S. Alsenan, "Audit-as-code: A policy-as-code framework for continuous AI assurance,"

Frontiers in Artificial Intelligence, vol. 9, 2026, doi: 10.3389/frai.2026.1759211.
11. Koch and J. A. Wellbrock, "Beyond task success: An evidence-synthesis framework for evaluating, governing, and orchestrating agentic AI," *arXiv preprint arXiv:2604.19818*, 2026, doi: 10.48550/arXiv.2604.19818.
12. National Institute of Standards and Technology, *Artificial Intelligence Risk Management Framework (AI RMF 1.0)*, NIST AI 100-1, Jan. 2023, doi: 10.6028/NIST.AI.100-1.
13. ISO/IEC/IEEE 15026-2:2022, *Systems and Software Engineering—Systems and Software Assurance—Part 2: Assurance Case*, 2nd ed., 2022.
14. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004, doi: 10.2307/25148625.
15. K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007, doi: 10.2753/MIS0742-1222240302.
16. V. Sharma, J. Singh, R. C. Sharma, and A. Singh, "Machine learning-based prediction and optimization of surface roughness in FDM-printed PLA using ANN and Box-Behnken design," *Materials Science and Technology*, vol. 41, no. 15, pp. 1153–1173, 2025. https://journals.sagepub.com/doi/abs/10.1177/02670836251370811
17. H. Weigand and P. Johannesson, "How to identify your design science research artifact," in *Proc. 2023 IEEE 25th Conf. Business Informatics (CBI)*, Jun. 2023, pp. 1–10. https://ieeexplore.ieee.org/abstract/document/10187511/
18. E. Debenedetti, J. Zhang, M. Balunovic, L. Beurer-Kellner, M. Fischer, and F. Tramer, "AgentDojo: A dynamic environment to evaluate prompt injection attacks and defenses for LLM agents," in *Advances in Neural Information Processing Systems*, vol. 37, 2024, arXiv:2406.13352.
19. Y. Ruan, H. Dong, A. Wang, S. Pitis, Y. Zhou, J. Ba, Y. Dubois, C. J. Maddison, and T. Hashimoto, "Identifying the risks of LM agents with an LM-emulated sandbox," in *Proc. Int. Conf. Learning Representations*, 2024, arXiv:2309.15817.
20. X. Zhou *et al.*, "SafeAgent: Safeguarding LLM agents via an automated risk simulator," in *Proc. 64th Annual Meeting of the Association for Computational Linguistics*, 2026, pp. 32516–32543, doi: 10.18653/v1/2026.acl-long.1501.
21. V. Kulothungan, "Using blockchain ledgers to record AI decisions in IoT," *IoT*, vol. 6, no. 3, Art. no. 37, 2025. https://www.mdpi.com/2624-831X/6/3/37
22. S. T. Fife and J. D. Gossner, "Deductive qualitative analysis: Evaluating, expanding, and refining theory," *International Journal of Qualitative Methods*, vol. 23, Art. no. 16094069241244856, 2024. https://journals.sagepub.com/doi/pdf/10.1177/16094069241244856
23. Y. Wang, J. Zhang, T. Cai, *et al.*, "From agent traces to trust: A survey of evidence tracing and execution provenance in LLM agents," *arXiv preprint arXiv:2606.04990*, 2026, doi: 10.48550/arXiv.2606.04990.
24. H. Dibowski, "Full traceability and provenance for knowledge graphs," in *Formal Ontology in Information Systems*, pp. 223–237, IOS Press, 2024. https://journals.sagepub.com/doi/pdf/10.3233/FAIA241309
25. National Institute of Standards and Technology, *Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile*, NIST AI 600-1, Jul. 2024, doi: 10.6028/NIST.AI.600-1.

26. T. Shi, J. He, Z. Wang, L. Wu, H. Li, W. Guo, and D. Song, "Progent: Securing AI agents with privilege control," *arXiv preprint arXiv:2504.11703*, 2025.
27. G. Zhu, C. Wang, Z. Wang, Z. Li, and Q. Li, "OpenPort Protocol: A security governance specification for AI agent tool access," *arXiv preprint arXiv:2602.20196*, 2026. https://arxiv.org/pdf/2602.20196
28. European Parliament and Council, "Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence," *Official Journal of the European Union*, Jul. 12, 2024.
29. N. V. Konduri, "Cloud-Ledger Railways for regulated banking microservices," *Journal of Electrical Systems and Information Technology*, vol. 13, no. 1, Art. no. 91, 2026. https://link.springer.com/content/pdf/10.1186/s43067-026-00393-3.pdf
30. B. N. Jørgensen and Z. G. Ma, "Governing agentic AI in enterprise workflows: A bounded-autonomy framework for delegated authority and controlled execution," *Information*, vol. 17, no. 9, Art. no. 923, 2026. https://www.mdpi.com/2078-2489/17/9/923
31. Chan *et al*., "Visibility into AI agents," in *Proc. ACM Conf. Fairness, Accountability, and Transparency*, 2024, pp. 958–973, doi: 10.1145/3630106.3658948.
32. F. Ikhwantri, I. L. Ojanguren, D. Marijan, M. I. Maslioukova, J. A. Marin, and P. Kosmides, "An agentic RAG and evaluation framework for assurance case generation: Industrial use case for the EU Cyber Resilience Act compliance," *arXiv preprint arXiv:2608.19509*, 2026.
33. C. Bonthu, G. Malik, and M. R. Dhanagari, "AI-driven secure and scalable data architecture with multi-domain MDM and cybersecurity for real-time fintech decision-making," in *Proc. 2025 Tenth Int. Conf. Science Technology Engineering and Mathematics (ICONSTEM)*, Nov. 2025, pp. 1–7. https://ieeexplore.ieee.org/document/11374416
34. P. Ananthula, S. Sepuri, M. Sethi, and H. Goel, "The strategic value of AI-driven physical identity and access management: Establishing cyber physical zero trust in critical infrastructure," in *Proc. 2026 4th Int. Conf. Self Sustainable Artificial Intelligence Systems (ICSSAS)*, May 2026, pp. 292–300, doi: 10.1109/ICSSAS68835.2026.11559385.
35. H. Hou, Y. Ma, and J. Miao, "CapAgent: Semantic data-flow governance for LLM agents in big-data cognitive computing," *Big Data and Cognitive Computing*, vol. 10, no. 9, Art. no. 293, 2026. https://www.mdpi.com/2504-2289/10/9/293
36. Z. Orhan, M. Orhan, B. D. Lund, N. R. Mannuru, R. V. K. Bevara, and B. Porter, "Artificial intelligence standards in conflict: Local challenges and global ambitions," *Standards*, vol. 5, no. 4, Art. no. 27, 2025. https://www.mdpi.com/2305-6703/5/4/27
37. S. Choudhary and S. Sepuri, "AI-augmented decision governance in compliance-critical enterprise workflow systems: Implications for leadership and decision-making models in mid-sized enterprises," in *Proc. 2026 4th Int. Conf. Self Sustainable Artificial Intelligence Systems (ICSSAS)*, May 2026, pp. 1183–1191, doi: 10.1109/ICSSAS68835.2026.11559573.
38. S. Padmanabhan Poti and C. J. Stanton, "Enabling affordances for AI governance," Journal of Responsible Technology, vol. 18, Art. no. 100086, 2024. https://www.sciencedirect.com/science/article/pii/S266665962400012X

Conflict of Interest: No Conflict of Interest
Source of Funding: Author(s) Funded the Research
How to Cite: Krishnan, A. (2026). Evidence to Policy Assurance for Agentic AI: A Verifiable Provenance Framework for Regulated Enterprises. *Frontiers in Emerging Technology, 2*(5), 1-22.