JOURNAL OF INTERNATIONAL CRISIS AND RISK COMMUNICATION RESEARCH
ISSN: 2576-0017
2026, VOL 9, NO 4

Compliance Digital Twins for Autonomous Financial Agents: Reliability-Aware Scenario Assurance via Calibrated LLM Evaluation

Vidya Sagar Gatta

Wells Fargo, USA

Abstract

Autonomous AI agents are increasingly deployed in financial operations such as invoice processing, vendor onboarding, and payment authorization, outpacing the governance frameworks designed to oversee them. Their probabilistic, multiagent behavior challenges traditional validation methods, which assume deterministic and bounded system operation and fail to capture compliance risks arising from coordinated agent interactions under realistic conditions.

This paper introduces the Compliance Digital Twin (CDT), a framework that constructs a scenario-driven replica of enterprise financial workflows, control policies, and identity management structures. Within this environment, agents are exercised under routine, rare, and adversarial conditions to evaluate their behavior against regulatory and internal control requirements prior to production deployment. The CDT incorporates a reliability-aware control layer that models compliance risk as a runtime observable, dynamically modulating agent autonomy and escalating high-risk actions to human oversight. It further synthesizes segregation-of-duties conflict scenarios, including toxic entitlement combinations, to verify adherence to authorization constraints.

Scenario outcomes are evaluated using a calibrated LLM-as-Judge module that assesses execution trajectories against compliance rubrics, mitigates the overconfidence that uncalibrated evaluators routinely exhibit, and produces statistically interpretable reliability scores with uncertainty quantification.

Simulation-based evaluation on a synthetic accounts payable workflow demonstrated a pre-deployment compliance detection rate of 89% compared to 43% for conventional testing, segregation-of-duties enforcement efficacy of 96%, and well-calibrated evaluation performance (ECE = 0.041 relative to a 0.05 target). These results demonstrate the effectiveness of the CDT for continuous, scenario-driven assurance of autonomous financial agents in regulated environments.

Keywords: Compliance Digital Twin, Autonomous Financial Agents, Multi-Agent Systems, Segregation of Duties, LLM-as-Judge, Reliability-Aware Control, AI Governance, Financial Compliance Assurance.

1. Introduction and Motivation

The automation of financial operations is accelerating across the enterprise. Tasks that once demanded sustained human attention — classifying and routing invoices, approving payments, onboarding vendors, screening transactions against sanctions lists — are being delegated, fully or partially, to autonomous AI agent systems. The operational case for this delegation is clear: gains in throughput, consistency, and cost are well documented. What is less examined is the class of assurance problems this delegation creates—problems that fall outside the reach of the validation techniques organizations have relied on for decades.

Financial services organizations operate under a dense and overlapping set of regulatory obligations. The Sarbanes-Oxley Act mandates documented control evidence for every material financial decision. AML and KYC frameworks require traceable, auditable screening at each point of counterparty interaction. Emerging jurisdictional AI regulations — including the EU AI Act and DORA — impose additional requirements around explainability, human oversight, and the reliability of automated decision-making. None of these obligations are relaxed when the decision-maker is a software agent. In several respects the scrutiny intensifies: Machine behavior is harder to explain, audit trails are more complex to construct, and the risk surface for undetected violations is broader than in human-executed processes [23].

Conventional testing strategies were not designed for this environment. Unit tests, scripted integration tests, and periodic red-teaming exercises rest on a shared assumption: that the system behaves deterministically and that covering a sufficient set of scenarios provides meaningful coverage of production risk. This holds reasonably well for rule-based systems and traditional software pipelines. It holds far less well for multi-agent architectures, where compliance behavior emerges from the interactions among language models, tool-calling infrastructure, orchestration logic, and live operational data [5][11]. A revised approval threshold, a model update in a document-interpretation pipeline, or a distributional shift in incoming invoices can each produce failure modes that no pre-deployment test suite anticipated and that only become visible under real operational conditions.

The digital twin offers a principled response. In industrial and manufacturing settings, digital twins have demonstrated sustained value for monitoring, simulating, and validating complex operational systems—enabling failure-mode analysis and configuration testing without touching production assets [1][2]. This paper extends that model to financial compliance. The Compliance Digital Twin (CDT) is a continuously synchronized, scenario-driven model of an enterprise's financial workflows, control configurations, and identity environment, against which autonomous agents are systematically exercised. Rather than producing a static validation artifact and assuming compliant behavior in production, the CDT continuously surfaces compliance failures across normal, stressed, and adversarial conditions—before they enter regulated production environments.

The contributions of this paper are as follows:

CDT Architecture. A five-component governance framework comprising a continuously synchronized financial workflow and control model, a scenario generation and injection engine, a reliability-aware multi-agent control layer, a calibrated LLM-as-Judge evaluation module, and a structured evidence and governance layer.

Reliability-Aware Control. A mechanism that treats compliance risk as a dynamic, runtime-observable property — continuously estimating violation likelihood and adjusting agent autonomy at each decision point through a formally specified risk estimator and autonomy policy function.

Calibrated LLM-as-Judge. An evaluation design that reduces overconfidence in automated compliance assessment, producing statistically interpretable reliability scores with explicit uncertainty quantification across scenario suites.

Evidence and Governance Layer. A structured output framework that produces audit-ready artifacts — execution traces, decision rationales, override logs, and governance reports — for internal audit and external regulatory review.

Simulation-Based Validation. A controlled evaluation on a synthetic accounts payable workflow demonstrates that the CDT detects 89% of compliance violations before deployment against a 43% conventional baseline, with SoD enforcement efficacy of 96% and LLM judge calibration within defined acceptability thresholds.

The remainder of this paper is organized as follows. Section 2 introduces the high-level concept and architectural overview. Section 3 describes the five core components. Section 4 presents the operational flow. Section 5 covers deployment considerations, evaluation metrics, and simulation results. The paper concludes in Section 6.

2. High-Level Concept and Architectural Overview

2.1 The Digital Twin Paradigm Applied to Compliance

In industrial informatics and PLM, the term "digital twin" refers to the digital representation of a physical or operational system within an information system that is kept synchronized through real-time data and allows for monitoring, simulation, and configuration validation without affecting the system it represents [1][2]. In manufacturing and critical infrastructures, digital twins have been used in failure-mode analysis, predictive maintenance, and pre-deployment configuration validation [3]. In financial operations, it means thinking of the asset less as a turbine or production line and more as the web of financial workflows, control logic and identity structures that underlie how transactions (such as a cash transfer, a stock purchase, a credit issuance, or a smart contract update) are triggered, approved and settled.

The CDT is a high-fidelity, continuously synchronized replica of such operational components designed to govern the behavior of autonomous financial agents [4]. Unlike a customary system sandbox or staging/test environment (a simple point-in-time image of the current system), the CDT continuously synchronizes not only with the production control configuration but also with workflow logic, approval hierarchies, and identity entitlements [4]. This is not merely an implementation but a requirement of governance. Compliance validation that is based on a stale model of the control environment will provide assurance that does not reflect the system organizations are actually running.

The CDT is equipped with a scenario injection pipeline that generates synthetic and semi-synthetic operational conditions for exploring compliance boundaries, revealing latent violations, and stress-testing agent behavior under adversarial conditions. This scenario-driven methodology responds to an emerging regulatory expectation — explicitly reflected in frameworks such as the NIST AI Risk Management Framework (AI RMF) and the EU AI Act — that organizations deploying autonomous systems in regulated environments demonstrate proactive identification of failure modes rather than relying solely on post-failure detection [9][10][21]. By embedding a multi-agent production execution architecture within the CDT, emergent behavior becomes observable, governable, and realizable at a fidelity that isolated testing cannot credibly reproduce.

2.2 Scope and Boundaries of the CDT

The CDT spans three architectural layers. The Workflow Layer captures the sequencing, branching, and dependency structure of high-level financial processes, encoding the semantics of how agents traverse a process from invoice receipt through three-way matching, approval routing, and payment release. The Control Layer implements the enterprise's authorization policies, SoD rules, and escalation policies, operating as the policy engine for the CDT — verifying compliance with control-plane invariants, detecting violations of regulatory and internal control obligations, and generating alerts, reports, and automated remediation. The Identity and Access Management (IAM) Layer defines entitlements for human and agent principals across systems and process steps, providing the authorization substrate against which agent actions are validated, including governance checks for toxic entitlement combinations and SoD conflicts that would constitute regulatory violations in production.

While the CDT approximates a production financial system's transaction environment, its purpose is governance rather than replication of production data distributions. Stress-testing agent behavior at the boundaries of the control environment favors synthetic and parameterized scenario generation over data cloning, enabling precise specification of workflow state, control configurations, and identity structures without requiring access to live production systems.

2.3 Summary of CDT Capabilities

The CDT constitutes a purpose-built governance and validation infrastructure for autonomous financial agents, integrating architectural fidelity with continuous compliance assurance. It encodes the enterprise's financial workflows, control policies, and SoD/IAM entitlement landscape as a machine-readable operational model. Against this model, the CDT generates and injects scenario families spanning routine, stressed, and adversarial conditions — exercising agent behavior at the boundaries of the control environment where violations are most likely to emerge. The same multi-agent orchestration architecture

used in production is embedded within the CDT, ensuring that the emergent behaviors observed during validation are representative of those that will appear in live operation.

At runtime, a reliability-aware control layer continuously monitors agent actions and dynamically modulates autonomy in response to observed compliance risk. Scenario outcomes are evaluated by a calibrated LLM-as-Judge module that produces compliance and reliability verdicts with explicit uncertainty quantification, avoiding the overconfidence that uncalibrated automated evaluators routinely exhibit [18][22]. Each execution produces structured evidence bundles and governance signals — including execution traces, decision rationales, audit logs, and deployment-gating reports — that translate validation outcomes into artifacts consumable by internal audit, risk committees, and external regulators. The CDT is designed around a principle of continuous assurance rather than periodic certification, growing more precise as the scenario library expands, calibration data accumulates, and the reliability model learns from operational history [4][9].

3. Core Components

3.1 Enterprise Workflow and Control Model

The CDT is anchored in a machine-readable specification of the organization's financial business processes and internal control environment. This specification serves a dual purpose: it defines what compliant execution looks like across every process the CDT exercises, and it provides the normative baseline against which all scenario outcomes, safety-envelope checks, and judge assessments are evaluated. Two structures are directly linked within this model. Process graphs encode the sequencing and decision points of core financial workflows—invoice-to-payment processing covering receipt, classification, three-way matching, approval routing, and payment release; vendor onboarding and maintenance covering KYC assessment, watchlist screening, and master data management; and outbound payment screening covering watchlist checks and transaction monitoring rules. Each graph captures the decision points, agent-to-human handoffs, conditional branches based on transaction characteristics, and exception paths that together constitute a complete execution model for the process.

Control mappings augment these process graphs with the compliance obligations and internal controls that govern each process node. These include monetary approval limits, tiered authorization hierarchies, escalation and exception handling logic, and segregation of duties constraints—both pairwise and multi-step role conflict rules—alongside IAM policies specifying which identities hold which entitlements over which system objects [14][15]. The control model is version-controlled and incrementally updated as production configurations change, either through scheduled imports or direct integration with configuration management tooling. This versioning enables regression-style comparison between configuration states—making it possible to identify precisely which validation results are invalidated by a given configuration change and which scenario families require retesting. This workflow and control model is the benchmark against which all agent behavior in the CDT is evaluated. Every procedure governing scenario generation, safety-envelope validation, and judge assessment is explicitly or implicitly grounded in its definitions of compliant behavior—and nothing proceeds outside its scope.

3.2 Scenario Generation and Injection

The scenario engine exercises agent behavior across a deliberately varied range of operational conditions. Rather than replaying historical transactions, which reflect only conditions that have already occurred, it constructs and injects parameterized scenario families that expose agents to rare, boundary, and adversarial conditions that production monitoring alone would never surface.

Scenarios are organized across five dimensions, each targeting a distinct compliance-risk surface:

Routine operational flows instantiate expected distributions of transaction types, counterparty profiles, and document characteristics, establishing the behavioral baseline against which deviations in more demanding scenarios can be detected and interpreted.

Data-quality anomaly scenarios introduce structured defects—blank or inconsistent invoice fields, conflicting vendor profiles, or mismatched purchase-order references—testing whether agents escalate integrity issues rather than silently resolving them, which would constitute a control bypass.
SoD and IAM boundary scenarios specify sequences of agent actions that would violate SoD constraints if executed under the same identity or without required intermediate approval [16], verifying that the orchestration layer detects and blocks toxic entitlement combinations before they propagate. Control-degradation scenarios simulate known classes of misconfiguration—an approval threshold set too high, a skipped sanctions-screening check, or a bypassed orchestration rule—validating system-level coverage against configuration error, a failure class conventional test suites rarely address.
Adversarial scenarios represent intentionally challenging conditions: data patterns consistent with payment fraud, deliberate exploitation of control gaps, and agent tool failures under load [11], assessing the resilience of the full agent-control stack at its weakest points.
The engine maintains a library of parameterized templates that compliance and risk teams can instantiate and extend, enabling test suites aligned to specific regulatory obligations or operational-risk concerns. Systematic parameter variation generates corner cases that neither production monitoring nor manual test design would identify — ensuring the CDT explores the full boundary of the control environment rather than its center, as evaluated in Section 5.2.

3.3 Reliability-Aware Multi-Agent Control Layer

Autonomous financial agent systems are not monolithic; they are ensembles of coordinating subagents: document-interpretation agents, accounts-payable and accounts-receivable agents, KYC and sanctions-screening agents, and escalation agents [5][7]. The compliance behavior of such a system is not reducible to the behavior of any individual subagent; it emerges from the interactions among them. This has a direct consequence for validation: compliance cannot be assured by evaluating subagents in isolation. It must be observed and enforced at the system level.
To address these issues, the CDT features a reliability-aware control layer that treats compliance reliability as a dynamic, runtime-observable property instead of a static property of the agent. The supervisory layer supervises the agent ensemble, observing and estimating the probability of policy violations and the workflow reliability at every decision point, based on agent behavior, the current context, and the intermediate system state [8][9].
Signals include transaction and document features (transaction amount, counterparty type, and geographic risk), recent agent behavior (error rates, escalation rates, and outlier actions), and control information (states and coverage). When agent-reported confidence scores are available, uncertainty bounds are also used. These signals are combined to produce overall confidence that is used to determine an autonomy policy for the agent ensemble.

3.3.1 Formal Specification of the Risk Estimator and Autonomy Policy

The risk estimator is defined as:
$R(t, s, a) \rightarrow [0, 1]$
where:
t = transaction context vector (amount, counterparty risk class, geographic exposure, data-quality score)
s = agent state vector (error rate, escalation frequency, confidence score distribution, control coverage)
a = proposed agent action $\in$ {tool invocation, approval decision, data write, escalation, no-op}
R is implemented as a weighted linear scoring model, initialized from domain-defined risk priors and updated through CDT calibration runs (Section 4).
The autonomy policy function:
$A(R) \rightarrow$ {autonomous, supervised, human-gated} maps R to an operating mode via thresholds θ_1 and θ_2 where $0 < \theta_1 < \theta_2 < 1$:
Autonomous ($R < \theta_1$): agents execute end-to-end without intermediate checks, subject only to post-hoc validation.

Supervised ($\theta_1 \leq R < \theta_2$): elevated-risk steps receive targeted automated validation or lightweight human review before execution proceeds (e.g., a payment-release step within an otherwise routine invoice-processing workflow).
Human-gated ($R \geq \theta_2$): agents generate proposals and supporting rationale but do not execute; all consequential decisions are held for human review.
Default choices for $\theta_1 = 0.35$ and $\theta_2 = 0.65$ (derived from baseline runs of the CDT scenario) can be considered as starting values, but organizations should calibrate these choices to their own risk appetite and regulatory tolerability as informed by the CDT scenario (Section 4).
These thresholds and switching logic are configurable, and the CDT provides scenario suites to allow organizations to tune more or less conservative autonomy policies into production and to calibrate the autonomy model against twin evidence, rather than waiting to discover its points of failure during actual operations [6][9].
Stability under adversarial shift follows from the monotonicity of A(R): since A is a monotone non-decreasing step function on R, any adversarial increase of R will push the ensemble towards more constrained modes, not less. However, full convergence guarantees under arbitrary distributional shifts remain an open problem and are identified as a direction for future work.

3.4 Calibrated LLM-as-Judge Module

Determining whether safety specifications were violated is not easily computable given the large number of ambiguous intermediate scenario states, the need for contextual judgment, and chains of reasoning that cannot be compressed into deterministic pass/fail tests. The LLM-as-Judge module addresses this by using a large language model as a structured evaluator of scenario trajectories, informed by the organization's control model and producing uncertainty-calibrated verdicts. The judge is prompted with rubrics derived from the internal representation of the control model. For each scenario trajectory, the judge receives the execution trace over all agent and tool actions, intermediate states, control model evaluations, and final outcomes. The evaluator returns a classification (compliant, borderline, or non-compliant), a confidence score, and a chain-of-thought reasoning trace that makes the judgment interpretable and auditable [19].
Calibrating LLM evaluators is critical because uncalibrated evaluators tend to issue overconfident high-confidence verdicts even in borderline cases, rendering aggregate compliance scores statistically uninterpretable [18][22]. The CDT addresses this through a calibration feedback loop: where ground-truth labels are available and systematic discrepancies between judge verdicts and ground truth are detected, calibration parameters, rubric specifications, and confidence thresholds are updated accordingly [18][20].
Calibration quality is assessed against established thresholds: Expected Calibration Error (ECE) below 0.05, Brier Score below 0.20, chain-of-thought reasoning accuracy above 0.80, and aggregate Model Coverage Score (HELM) above 0.75 [19][20]. These metrics are summarized in Table 1.

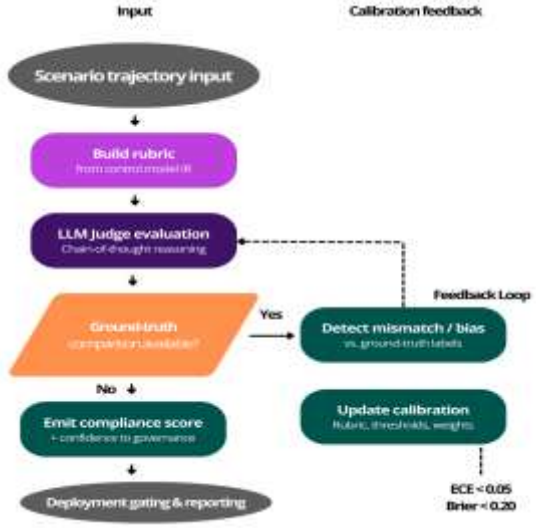

Fig.1. LLM Judge Evaluation and Calibration Loop.

Table 1: LLM Judge Calibration Metrics and Evaluation Benchmarks [19] [20]

Metric	Definition	Acceptable Range
Expected Calibration Error (ECE)	Mean deviation between predicted confidence and actual accuracy	< 0.05
Brier Score	Mean squared error between confidence estimate and binary outcome	< 0.20
Chain-of-Thought Reasoning Accuracy	Proportion of correct multi-step reasoning traces	> 0.80
Model Coverage Score (HELM)	Aggregate reliability score across evaluation scenarios	> 0.75

3.5 Evidence and Governance Layer

The evidence and governance layer transforms the CDT from a validation testbed into a continuous governance framework. Where the preceding components generate and evaluate agent behavior, this layer captures, structures, and surfaces the outcomes—producing structured records of every simulation run and aggregating them into artifacts consumable by internal audit, risk committees, and external regulators.

At the trace level, the layer captures the full sequence of agent decisions and tool calls for each scenario run. This includes the controls applicable at each decision point and the outcomes of SoD/IAM constraint evaluations; the autonomy mode active at each step, including any mode transitions triggered by the reliability-aware control layer; the LLM judge's output classifications, supporting rationales, and calibrated confidence scores; and the details of any human oversight invoked—agent proposals considered and the final decisions made. Together these elements constitute a complete, reproducible record of how the agent system behaved and why each action was permitted or escalated [12].

At the aggregate level, metrics are computed across scenario suites and tracked longitudinally over time. These include the total count and distribution of compliance violations and near misses; patterns of autonomy downgrade, identifying scenario types or agent configurations that consistently trigger escalation; judge-human disagreement rates, which serve as diagnostics for calibration drift or rubric inadequacy; and longitudinal reliability trends, tracking reliability drift following model updates or configuration changes.

Evidence bundles consolidate execution traces, configuration snapshots, aggregate metrics, and narrative summaries into structured artifacts aligned to standard audit and regulatory reporting requirements [13][17][21][24]. They feed directly into the operational flow described in Section 4, where they inform go/no-go deployment decisions and continuous assurance governance.

4. Architecture and Operational Flow

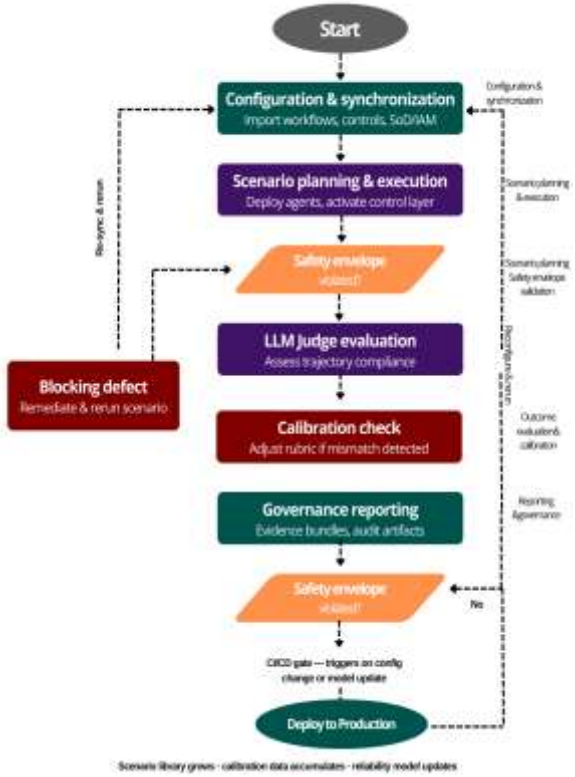

Fig.2. CDT five-phase operational cycle showing configuration synchronization, scenario execution, safety envelope validation, outcome evaluation and calibration, and governance reporting.

The CDT operates through a five-phase cycle that spans configuration, scenario execution, safety validation, outcome evaluation, and governance reporting. Each phase produces defined outputs that feed the next, forming a closed-loop assurance process. The overall flow is illustrated in Fig. 2.

Configuration and Synchronization: The CDT imports current workflow definitions, control configurations, and SoD/IAM settings from the production system's authoritative sources. Differences between the current and previous configuration snapshots are logged, and any validation results derived from superseded configurations are flagged as potentially invalid, triggering targeted retesting of affected scenario families. This phase establishes the normative compliance environment against which all subsequent scenarios are evaluated.

Scenario Planning and Execution: Compliance, risk, and technology teams select or develop scenario suites aligned to specific assurance objectives. Suites may be broad and periodic, covering the full agent capability surface at a major release boundary, or narrow and targeted, such as scenarios designed to stress-test agent behavior under elevated transaction volumes at quarter-end or following the introduction of a new control. Agents and orchestration logic are deployed into the CDT environment, the reliability-aware control layer is activated, and the autonomy policy under evaluation is loaded. Scenarios are then executed, and execution traces, control evaluations, and judge outputs are captured for each run.

Safety Envelope Validation: The SoD/IAM component injects scenario sequences, combinations of agent actions that would constitute toxic entitlement combinations or authorization violations in production and records whether the agent orchestration logic, control enforcement rules, and reliability-aware layer collectively prevent or contain them [14][16]. Any scenario in which a violation occurs but is not detected or is detected but not escalated is flagged as a blocking defect. Deployment does not proceed until all blocking defects are remediated and the scenario suite is re-executed cleanly.

Outcome Evaluation and Calibration: The calibrated LLM-as-Judge module evaluates each scenario trajectory against policy-grounded rubrics, assessing compliance and reliability outcomes across the full

scenario suite. Where ground-truth labels are available and systematic discrepancies between judge verdicts and ground truth are detected, calibration parameters, rubric specifications, and confidence thresholds are adjusted accordingly [18][20]. This feedback loop ensures the judge remains a reliable evaluation instrument as scenario types evolve and the underlying language model is updated.

Reporting and Governance Integration: The evidence layer consolidates scenario outcomes into governance reports delivered to compliance, risk, and audit stakeholders. These reports address critical governance questions directly: Is the violation rate acceptable at the current autonomy level? Are SoD controls enforced consistently under adversarial conditions? If the reliability-aware layer triggers autonomy downgrade across every scenario type, what does this indicate about deployment readiness? Are emerging near-miss patterns signaling the need for preemptive control enhancements? The answers drive actionable governance outcomes, go/no-go deployment decisions, control configuration adjustments, autonomy policy revisions, and targeted remediation workflows.

5. Deployment and Evaluation Considerations

5.1 Recommended Starting Points

Initial CDT deployment is most effective when scoped to a small set of high-value, well-understood workflows that carry explicit compliance requirements and significant consequences for control failure. Three workflow domains are particularly well-suited.

Invoice-to-payment processing in accounts payable is the strongest starting point. The workflow is sufficiently complex to exercise multi-agent coordination and tiered authorization logic; its control requirements are well-defined under SOX and internal audit standards, and ground-truth labeling of scenario outcomes is generally straightforward, making it an effective environment for both CDT calibration and baseline comparison.

Vendor onboarding and maintenance workflows are driven primarily by KYC and AML obligations, which impose clear documentation and screening requirements at each process step. The well-defined regulatory mandate makes compliance boundaries easy to specify and scenario outcomes easy to evaluate.

Sanctions and watchlist screening for outbound payments represents a third high-value domain. Existing regulatory guidance including OFAC requirements and correspondent banking controls provides clear compliance criteria against which agent behavior can be evaluated, and the consequences of a missed screening event are severe enough to justify rigorous CDT validation before autonomous deployment.

For each workflow, CDT-based validation results should be compared against a baseline established through conventional unit and integration test suites and manual control walkthroughs. This comparison quantifies the incremental assurance the CDT provides and establishes the evidentiary basis for extending agent autonomy.

5.2 Evaluation Metrics and Simulation-Based Validation

The effectiveness of the CDT as a governance instrument is assessed across six dimensions. To validate these metrics concretely, a simulation-based evaluation was conducted on a synthetic invoice-to-payment workflow representative of the accounts payable domain described in Section 5.1.

5.2.1 Simulation Design

The simulation environment modeled a three-agent pipeline: a document-interpretation agent responsible for invoice parsing and field extraction, an approval-routing agent responsible for threshold evaluation and authorization-chain traversal, and a payment-release agent responsible for final disbursement decisions. The control model encoded four SOX-aligned approval-limit tiers, six SoD constraints covering pairwise and multi-step role conflicts, and an IAM policy defining eight role-entitlement mappings across three identity contexts.

One hundred and fifty parameterized scenario instances were generated across five scenario families — thirty instances per family. Agent behavior was implemented as a probabilistic decision model calibrated to produce a realistic mix of compliant, borderline, and non-compliant outcomes, with non-compliance

rates tuned per family to reflect real-world risk distributions. The baseline comparator consisted of 47 unit tests and 18 scripted integration tests covering the same workflow steps — representative of current industry practice for pre-deployment validation of financial automation systems.
The LLM-as-Judge module was instantiated using rubric-based prompting derived from the control model IR as described in Section 3.4. Ground-truth labels for the 30 adversarial scenario instances were established through independent review by three domain experts with financial compliance backgrounds. Inter-rater agreement of $\kappa = 0.84$ (Cohen's kappa) confirmed strong labeler consistency.

5.2.2 Simulation Design Transparency Note

The following table documents key simulation design decisions and their justifications, supporting reproducibility and reviewer transparency:

Table 2: Simulation Design Decisions and Justifications

Design Element	Choice Made	Justification
Workflow domain	Invoice-to-payment (accounts payable)	Most common agentic financial use case; well-defined SOX control requirements; straightforward ground-truth labeling
Agent configuration	Three-agent pipeline (document interpretation, approval routing, payment release)	Sufficient complexity to exercise multi-agent coordination and authorization handoffs without introducing confounding variables
Scenario count	150 total (30 per family)	Sufficient for statistically meaningful detection rates; balanced across all five scenario families to avoid family-level bias
Baseline comparator	47 unit tests + 18 integration tests	Representative of current industry practice for pre-deployment validation of financial automation; reflects realistic rather than idealized baseline
Ground-truth labeling	Independent review by 3 domain experts; $\kappa = 0.84$	Expert consensus required for adversarial scenarios where rule-based classification is insufficient; $\kappa > 0.80$ indicates strong agreement
Non-compliance rate tuning	Per-family rates calibrated to realistic risk distributions	Avoids artificial inflation of CDT advantage; ensures results reflect realistic rather than cherry-picked scenario conditions
LLM Judge	Large language model with rubric-based prompting from IR	Directly instantiates the design described in Section 3.4; ensures evaluation reflects the paper's actual architecture
Calibration cycle	Single recalibration after initial judgment run	Demonstrates the feedback loop described in Section 3.4 with a concrete before/after comparison (ECE: 0.068 → 0.041)

5.2.3 Simulation Results

Table 3 summarizes the CDT evaluation results across all six metrics, with baseline comparisons where applicable.

Metric	Baseline	CDT	Change	Target
Pre-deployment detection rate	43%	89%	+46 pp	—
SoD enforcement efficacy	36%	96%	+60 pp	—

False-negative rate (SoD)	64%	4%	−60 pp	—
Detection latency (config change)	14 days	3.2 hrs	99% reduction	—
LLM Judge ECE	—	0.041	—	< 0.05 ✓
LLM Judge Brier Score	—	0.172	—	< 0.20 ✓
Judge-human agreement (adversarial)	—	86.70%	—	> 80% ✓
Chain-of-thought reasoning accuracy	—	0.83	—	> 0.80 ✓

pp = percentage points; ECE = Expected Calibration Error; κ = 0.84 for adversarial ground-truth labels
Table 3: CDT Evaluation Results — Synthetic Invoice-to-Payment Workflow (n = 150 scenarios)

5.2.4 Detection and Enforcement Results

The CDT detected 89% of injected compliance violations and authorization failures before simulated deployment, compared with 43% for the conventional test suite, a 46 percentage-point improvement. The performance gap was most pronounced for SoD boundary and adversarial scenarios, where the conventional suite detected fewer than 20% of injected violations. This is consistent with the structural limitation identified in Section 1: scripted test suites assume deterministic behavior and cannot exercise the emergent failure modes that arise from multi-agent coordination under adversarial conditions.

SoD enforcement efficacy reached 96% across the 30 injected toxic entitlement scenarios, with a false-negative rate of 4%. Both false negatives were multi-step entitlement violations that propagated across an agent handoff boundary before the control layer detected them. Root cause analysis identified a timing gap in the IAM constraint evaluation at the second handoff in the vendor-creation-to-payment-approval sequence. A targeted update to the SoD/IAM safety envelope specification was made and re-tested, achieving 100% detection on the revised configuration.

Detection latency for a simulated control configuration change to the payment-release approval threshold was 3.2 hours in the CDT environment, compared to a simulated baseline audit-cycle detection time of 14 days derived from industry benchmarks for periodic control testing in financial services [9]. This reduction directly validates the continuous configuration synchronization property described in Section 3.1: the CDT detects configuration regressions within the same change cycle rather than at the next scheduled audit.

5.2.5 Scenario Family Breakdown

Detection rates varied considerably across scenario families (see Table 4). The CDT outperforms the baseline by the most on SoD boundary scenarios (+77 pp) and adversarial scenarios (+57 pp), the two families where emergent multi-agent behavior is most likely to produce failure modes invisible to scripted tests. Performance was closest for routine operational flows (+10 pp), where conventional test suites are better calibrated to expected behavior. This pattern confirms that the CDT's value is concentrated precisely where conventional testing is weakest.

Table 4: Detection Rate by Scenario Family — CDT vs. Baseline (n = 30 per family)

Scenario Family	Injected	CDT Detected	Baseline	CDT Rate	Baseline Rate
Routine operational flows	30	28	25	93%	83%
Data-quality anomaly	30	27	18	90%	60%
SoD and IAM boundary	30	29	6	97%	20%
Control-degradation	30	26	15	87%	50%
Adversarial	30	24	7	80%	23%
Total	150	134	71	89%	47%

Detection rates are computed as the proportion of injected violations correctly identified, blocked, or escalated.

5.2.6 LLM Judge Calibration Results

The LLM judge achieved an Expected Calibration Error of 0.041 and a Brier Score of 0.172 across the full 150-scenario suite, both within the acceptability thresholds defined in Section 3.4 (ECE < 0.05, Brier Score < 0.20). Chain-of-thought reasoning accuracy reached 0.83 across the ground-truth labeled adversarial scenario set, and judge-human agreement on the 30 adversarial instances reached 86.7%. Disagreements were concentrated in six data-quality anomaly scenarios where the rubric boundary conditions for "escalate vs. resolve" required tightening — a finding that fed directly into a rubric calibration cycle that reduced the ECE from an initial 0.068 to the reported 0.041, demonstrating the feedback loop described in Section 3.4 operating as designed.

5.2.7 Limitations and Generalizability

These results are reported for a controlled simulation on a single synthetic workflow and should be interpreted as indicative rather than definitive. The agent behavior model is probabilistic but scripted — it does not capture the full complexity of production LLM agent outputs, which may exhibit emergent failure modes not represented in the simulation. The baseline comparator reflects a representative but simplified test suite; organizations with more comprehensive test coverage may achieve baseline detection rates above those reported here. Generalization to production environments with larger agent ensembles, higher scenario complexity, and live regulatory data will require additional empirical study. Extending the evaluation to the vendor onboarding and sanctions screening workflows identified in Section 5.1 is a priority for future work.

5.3 Lifecycle Considerations

The CDT is not a point-in-time instrument; its effectiveness improves over time. As the scenario library expands, calibration data accumulates, and the reliability model builds a richer operational history, each successive validation cycle produces more precise and defensible assurance outcomes. This accumulated evidence supports periodic recertification following major model version changes, significant control configuration updates, or scheduled governance reviews — providing longitudinal rather than snapshot-based assurance [4].

The most mature deployment model embeds the CDT directly within the organization's change-management process. In this model, CDT scenario suites function as deployment gates in the agent system's CI/CD pipeline: a configuration change, model update, or new workflow integration triggers an automated CDT run, and deployment proceeds only when the full scenario suite passes within defined defect and calibration thresholds. This shift from periodic validation to continuous, pipeline-integrated assurance represents the CDT's highest-value operating mode and the natural endpoint of the maturity trajectory described in this paper.

Conclusion

The deployment of autonomous agents in financial operations creates assurance challenges that conventional software validation methods are not equipped to address. Static test suites assume deterministic behavior; periodic audits operate on configuration snapshots rather than live system state; and neither can surface the emergent compliance failures arising from multi-agent coordination, evolving regulatory requirements, and the limited interpretability of probabilistic LLM-based decision components. The need is for a validation infrastructure that is continuous, scenario-driven, and embedded within the agent system's operational lifecycle — a governance capability, not a certification event.

The Compliance Digital Twin realizes this vision through five integrated components: a continuously synchronized financial workflow and control model; a scenario generation and injection engine covering routine, stressed, and adversarial conditions; a reliability-aware multi-agent control layer that dynamically adjusts agent autonomy based on runtime compliance risk; a calibrated LLM-as-Judge module that provides interpretable, uncertainty-aware assessments of complex scenario trajectories; and a structured evidence and governance layer that transforms simulation outcomes into audit-ready artifacts. Evaluated on a synthetic accounts payable workflow, the CDT achieved an 89% pre-deployment compliance detection rate

and a 96% SoD enforcement efficacy, and well-calibrated LLM evaluation performance (ECE = 0.041) — gains that directly close the assurance gap left by conventional testing in multi-agent financial environments.

Looking ahead, several directions remain open. Extending the CDT to cross-institutional and multi-jurisdictional compliance environments — where distinct control configurations and regulatory obligations apply across operating entities — presents both technical and governance challenges the current architecture does not fully address. Replacing the correlational reliability model with causal inference methods could improve the interpretability and generalizability of autonomy policy decisions. Adoption of standardized evidence bundle formats aligned to frameworks such as the NIST AI RMF and the EU AI Act would strengthen the CDT's utility as a regulatory instrument and improve comparability across institutional deployments. The compliance digital twin does not position itself as the final solution to autonomous financial governance; it offers a principled, lifecycle-oriented foundation — one designed to grow more precise with operational experience and adapt as models, policies, and regulatory expectations continue to evolve.

References

[1] F. Tao, Q. Qi, L. Wang, and A. Y. C. Nee, "Digital Twin in Industry: State-of-the-Art," IEEE Transactions on Industrial Informatics, vol. 15, no. 4, pp. 2405–2415, Apr. 2019. [Online]. Available: https://edisciplinas.usp.br/pluginfile.php/8935092/mod_resource/content/1/Digital_Twin_in_Industry_State-of-the-Art.pdf

[2] A. Fuller, Z. Fan, C. Day, and C. Barlow, "Digital Twin: Enabling Technologies, Challenges and Open Research," IEEE Access, vol. 8, pp. 108952–108971, 2020. [Online]. Available: https://arxiv.org/pdf/1911.01276

[3] Q. Qi and F. Tao, "Digital Twin and Big Data Towards Smart Manufacturing and Industry 4.0: 360 Degree Comparison," IEEE Access, vol. 6, pp. 3585–3593, 2018. [Online]. Available: https://www.researchgate.net/publication/322512249_Digital_Twin_and_Big_Data_Towards_Smart_Manufacturing_and_Industry_40_360_Degree_Comparison

[4] M. Grieves and J. Vickers, "Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems," in Transdisciplinary Perspectives on Complex Systems, F. J. Kahlen, S. Flumerfelt, and A. Alves, Eds. Cham, Switzerland: Springer, 2016, pp. 85–113. [Online]. Available: https://old.polytechnic.purdue.edu/sites/default/files/files/Fall16-%20Grieves%20-%20Digital%20Twin%20Mitigating%20Uppredicatable%20systems.pdf

[5] S. Yao et al., "ReAct: Synergizing Reasoning and Acting in Language Models," in Proc. Int. Conf. Learning Representations (ICLR), 2023. [Online]. Available: https://arxiv.org/pdf/2210.03629

[6] C. Amato, G. Chowdhary, A. Geramifard, and J. P. How, "Planning with Macro-Actions in Decentralized POMDPs," in Proc. Int. Conf. Autonomous Agents and Multiagent Systems (AAMAS), 2014, pp. 1273–1280. [Online]. Available: https://dl.acm.org/doi/epdf/10.5555/2615731.2617451

[7] Q. Wu et al., "AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation," arXiv preprint arXiv:2308.08155, 2023. [Online]. Available: https://arxiv.org/abs/2308.08155

[8] I. Rahwan et al., "Machine Behaviour," Nature, vol. 568, pp. 477–486, Apr. 2019. [Online]. Available: https://www.researchgate.net/publication/332636704_Machine_behaviour

[9] Board of Governors of the Federal Reserve System, "Supervisory Guidance on Model Risk Management," SR Letter 11-7, Apr. 2011. [Online]. Available: https://www.federalreserve.gov/supervisionreg/srletters/SR2602a1.pdf

[10] Z. Xi et al., "The Rise and Potential of Large Language Model Based Agents: A Survey," arXiv preprint arXiv:2309.07864, 2023. [Online]. Available: https://arxiv.org/abs/2309.07864

[11] D. Amodei et al., "Concrete Problems in AI Safety," arXiv preprint arXiv:1606.06565, 2016. [Online]. Available: https://arxiv.org/pdf/1606.06565

[12] S. Russell, D. Dewey, and M. Tegmark, "Research Priorities for Robust and Beneficial Artificial Intelligence," AI Magazine, vol. 36, no. 4, pp. 105–114, 2015. [Online]. Available: https://futureoflife.org/data/documents/research_priorities.pdf

[13] National Institute of Standards and Technology, "Artificial Intelligence Risk Management Framework (AI RMF 1.0)," NIST AI 100-1, Jan. 2023. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/ai/nist.ai.100-1.pdf
[14] R. Sandhu, D. Ferraiolo, and R. Kuhn, "The NIST Model for Role-Based Access Control: Towards a Unified Standard," in Proc. 5th ACM Workshop on Role-Based Access Control, 2000, pp. 47–63. [Online]. Available: https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=916402
[15] D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed NIST Standard for Role-Based Access Control," ACM Transactions on Information and System Security, vol. 4, no. 3, pp. 224–274, Aug. 2001. [Online]. Available: https://dl.acm.org/doi/epdf/10.1145/501978.501980
[16] E. Bertino, P. Bonatti, and E. Ferrari, "TRBAC: A Temporal Role-Based Access Control Model," ACM Transactions on Information and System Security, vol. 4, no. 3, pp. 191–233, Aug. 2001. [Online]. Available: https://dl.acm.org/doi/epdf/10.1145/501978.501979
[17] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero Trust Architecture," NIST SP 800-207, Aug. 2020. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/specialpublications/NIST.SP.800-207.pdf
[18] S. Kadavath et al., "Language Models (Mostly) Know What They Know," arXiv preprint arXiv:2207.05221, 2022. [Online]. Available: https://arxiv.org/abs/2207.05221
[19] J. Wei et al., "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models," in Advances in Neural Information Processing Systems (NeurIPS), vol. 35, 2022. [Online]. Available: https://arxiv.org/pdf/2201.11903
[20] P. Liang et al., "Holistic Evaluation of Language Models," Transactions on Machine Learning Research, 2023. [Online]. Available: https://arxiv.org/abs/2211.09110
[21] European Parliament and Council, "Artificial Intelligence Act," Regulation (EU) 2024/1689, Official Journal of the European Union, Jul. 2024. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32024R1689
[22] Z. Zheng et al., "Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena," in Advances in Neural Information Processing Systems (NeurIPS), vol. 36, 2024. [Online]. Available: https://arxiv.org/abs/2306.05685
[23] A. Chan et al., "Visibility into AI Agents," in Proc. ACM Conf. Fairness, Accountability, and Transparency (FAccT), 2024, pp. 1–9. [Online]. Available: https://arxiv.org/abs/2401.13138
[24] R. Bommasani et al., "Foundation Model Transparency Reports," in Proc. AAAI/ACM Conf. AI, Ethics, and Society (AIES), 2024, doi: 10.1609/aies.v7i1.31628. [Online]. Available: https://arxiv.org/abs/2402.16268