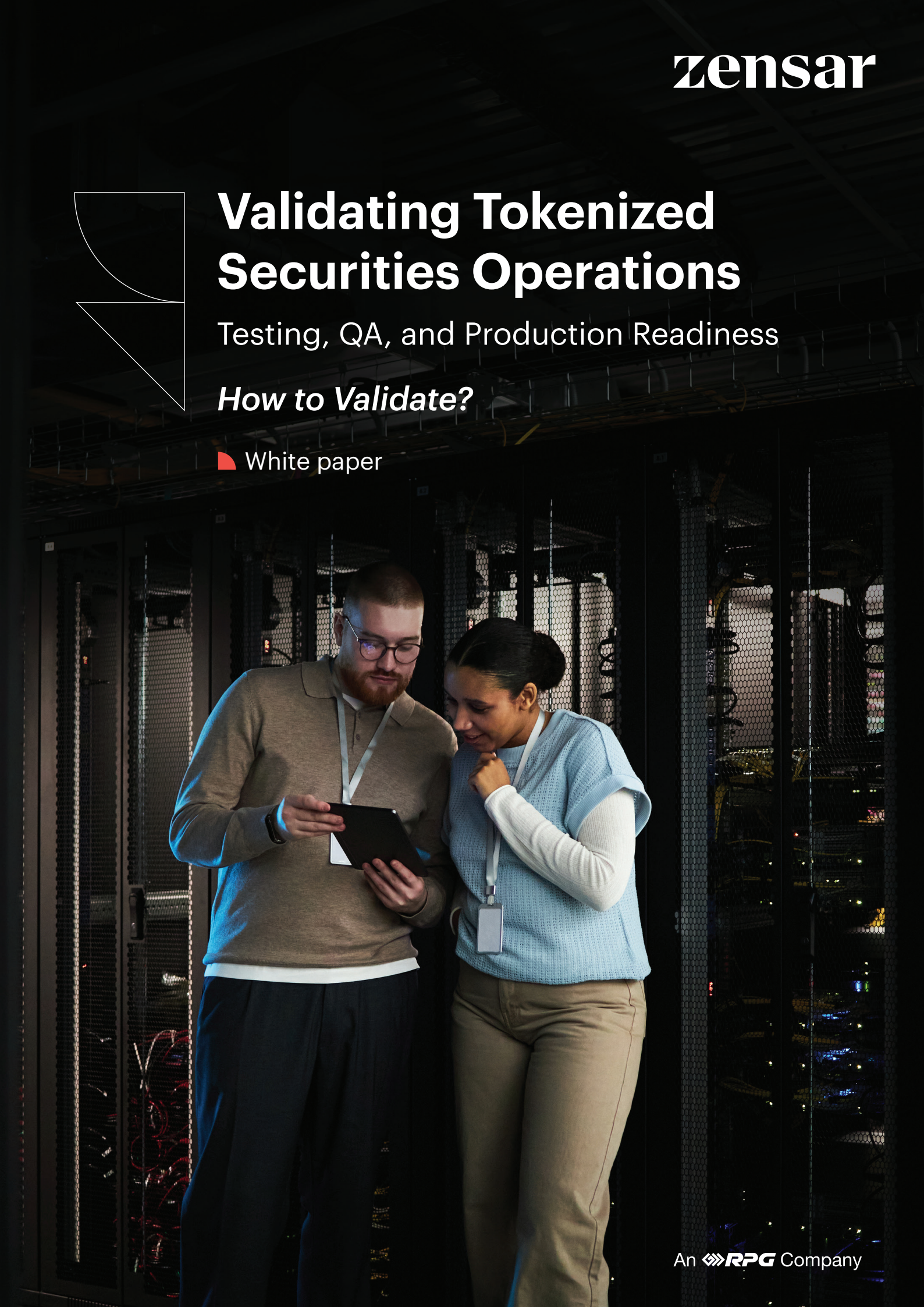
zensar

Validating Tokenized Securities Operations

Testing, QA, and Production Readiness

How to Validate?

White paper



An RPG Company

Editorial note

This paper provides a pre-production validation framework for DTC participant firms to evaluate their operational readiness before entering production. It draws on publicly available regulatory documents, DTCC publications, and Zensar's consulting work with capital markets participants.

Qualifications: (1) The DTC tokenization service is an announced pilot — timelines and specifications are subject to change. (2) Regulatory obligations are evolving; specific compliance obligations should be confirmed with legal counsel. (3) This paper is not legal advice and not independent research — Zensar has commercial services in this space. (4) Statements are distinguished as Confirmed Facts, Communicated Roadmaps, Analytical Inferences, Reasonable Assumptions, or Areas Still Evolving.

Series navigation | Paper 1 (Why it Matters): Executive framing, structural analysis, and decision framework. Paper 2 (How to Prepare): Full six-domain operational playbook, maturity model, implementation approaches, and risk considerations. Paper 3 (How to Validate): Pre-production validation framework, domain-by-domain test scenarios, and validation maturity gates.

This paper assumes familiarity with the six-domain framework introduced in Section 3 of paper 2 - Operational Readiness Framework for Tokenized Securities.

01 Why validation is distinct from implementation

Implementation and validation are distinct workstreams. This distinction is the conceptual anchor of this paper. A firm may correctly architect all six operational capability domains — event ingestion, dual-state position management, break detection, corporate action processing, override detection, and audit trail — and still encounter material failures when those capabilities are first exposed to real production conditions. Architecture correctness does not guarantee operational resilience.

Three characteristics of the DTC tokenization program make pre-production validation particularly important, and they are worth stating precisely.

1. No historical break-pattern data exists. As of August 2026 the service has completed production demonstrations (July 2026, with Citadel Securities, J.P. Morgan, Vanguard and Societe Generale) but has not yet entered live trading, which is targeted for Q4 2026. Participants therefore still cannot calibrate detection thresholds, routing logic, or break-resolution workflows against real operating experience; every threshold set before production remains a hypothesis, not a calibration
2. ComposerX API specifications remain incomplete. Integration behavior under edge conditions — malformed events, out-of-sequence delivery, rate-limited responses, and failover scenarios — cannot yet be fully anticipated from documentation alone. Systems that pass functional testing under well-formed inputs may fail silently under realistic production event streams.
3. Early-phase failures carry disproportionate consequences. Operational failures in the initial production phase carry reputational and operational consequences that are disproportionate to their volume. The participant community will closely observe initial entrants. A failure that would be routine in a mature operational environment, including a missed event, a misclassified break, or an undetected override, is far more consequential when it occurs in the first weeks of production.

These three characteristics, when combined, create a validation problem that cannot be solved by functional testing alone. Readiness validation addresses it by generating synthetic versions of the conditions that production will create before production begins. The goal is not to replicate production perfectly; it is to surface failure modes that architecture review and unit testing cannot surface.

The output of a well-executed validation program is not a pass/fail verdict. It is a set of evidence artifacts, including ingestion logs, break classification records, override detection latency measurements, and audit trail queries, that demonstrate control effectiveness to operational sign-off and internal audit review. This evidence serves two purposes: it confirms readiness before day one, and it establishes the baseline against which subsequent production performance can be measured.

02 The six domains and their validation requirements

The six operational capability domains are defined and analyzed in full in paper 2 (sections 3.1-3.6). This section does not repeat that analysis. It provides a validation-focused summary of each domain — what could fail in production that implementation testing alone cannot surface — and identifies the validation test that addresses it. The full test scenarios appear in section 3.

Domain	What could fail in production that testing alone cannot surface	Primary validation test
D1: Event ingestion	Malformed, duplicate, or out-of-sequence events from ComposerX may be silently dropped or incorrectly processed. Rate-limited responses and failover behavior under load may differ from those during functional testing.	Synthetic event injection: well-formed, malformed, duplicate, and out-of-sequence events are injected to confirm ingestion handles each condition correctly. See section 3.
D2: Dual-state position management	Rapid sequential updates to both record streams, such as on-chain and off-chain arriving close together, may produce timestamp attribution errors or momentary position inconsistencies that resolve incorrectly.	Concurrent update stress test: rapid sequential updates applied to both streams simultaneously; position view confirmed consistent and timestamps correctly attributed. See section 3.
D3: Break detection and classification	Classification logic trained on expected break types may misroute novel break patterns. Escalation triggers calibrated against historical thresholds may fire too early or too late against a new dual-record environment.	Break injection testing: known break types (timing break, root-wallet divergence, CA misapplication, identifier mismatch) are injected to confirm correct classification and routing in each case. See section 3.
D4: Corporate action processing	CA events applied to the traditional record may not propagate correctly to the tokenized record, or vice versa, under production timing conditions. Complex CA types (mergers, rights issuances) carry higher misapplication risk than routine income events.	CA simulation by type: dividend, coupon, merger, stock split, and rights issuance events simulated against dual-record positions. See section 3.
D5: Override Detection	DTC root-wallet interventions cannot be obtained from a sandbox environment. Detection logic and evidence generation can only be tested against synthetic overrides before production. Detection latency is a critical metric that cannot be estimated solely from design.	Root-wallet override simulation: synthetic override events injected into the LedgerScan event stream; detection latency, evidence generation, and audit entry confirmed within expected parameters. See section 3.
D6: Audit trail and regulatory evidence	Audit trail completeness cannot be assumed from the fact that logging is active. Point-in-time position reconstruction may fail for positions updated by root-wallet events, CA misapplications, or rapid concurrent updates. Gaps in the audit record may only be discovered during examination.	Audit evidence completeness test: audit trail queried across a defined event sequence and time range; all events confirmed present, correctly attributed, and supporting point-in-time position reconstruction. See section 3.

03 The Production Readiness Validation Framework.

Exhibit 1 maps the six validation activities to their domains, objectives, and example test scenarios. It is the centerpiece of this paper and the primary planning tool for a pre-production validation program. It is offered as a planning tool, not a compliance checklist.

The framework should be read in conjunction with the domain-by-domain validation detail in section 4, which explains the production-risk rationale for each test and the specific conditions each test must cover to be considered complete.

Exhibit 1. Production readiness validation framework

The following framework maps validation activities to the six operational capability domains. It is a planning tool, not a compliance checklist.

D1 Event ingestion **L3 gate**	**Validation objective:** Confirm reliable, authenticated receipt of all ComposerX event types. **Example test scenario:** Inject well-formed, malformed, duplicate, and out-of-sequence events; confirm ingestion behavior in each case.
D2 Dual-State Position **L3 gate**	**Validation objective:** Confirm that the simultaneous on-chain/off-chain position view is accurate under concurrent updates. **Example test scenario:** Apply rapid, sequential updates to both record streams; confirm the position view remains consistent and that timestamps are correctly attributed.
D3 Break detection **L3 gate**	**Validation objective:** Confirm break types are correctly detected, classified, and routed. **Example test scenario:** Inject known break types (timing break, root-wallet divergence, CA misapplication, identifier mismatch); confirm correct classification and routing in each case.
D4 Corporate action **L4 gate**	**Validation objective:** Confirm CA events are applied consistently across both records for each CA type in the tokenized portfolio. **Example test scenario:** Simulate dividend, coupon, merger, stock split, and rights issuance events against dual-record positions; confirm consistent application and the absence of one-sided CA processing.
D5 Override detection **L4 gate**	**Validation objective:** Confirm DTC root-wallet interventions are detected, documented, and evidence is generated within expected latency. **Example test scenario:** Simulate root-wallet override events in the LedgerScan event stream; confirm detection, evidence generation, and audit entry within expected latency parameters.
D6 Audit trail and evidence **L4 gate**	**Validation objective:** Confirm the audit trail is complete, queryable, and supports point-in-time position reconstruction. **Example test scenario:** Query the audit trail for a specified position across a defined time range; confirm all events are present, correctly attributed, and the reconstructed position matches the expected state.
REG Regression **All domains**	**Validation objective:** Confirm that integration behavior has not regressed following API specification updates or ComposerX version changes. **Example test scenario:** Rerun full validation suite after each API version change; confirm all six domain test results match baseline. Flag any deviation for root cause analysis before production promotion.

Gate legend: L3 gate = required for pilot-ready (Level 3) status. L4 gate = required for production-ready (Level 4) status. Regression = required after any API specification change, regardless of maturity level. See section 5 for the full maturity gate definitions.

04 Domain-by-domain validation detail

This section expands on each domain entry in Exhibit 1, including the production-risk rationale and the specific conditions each test must cover to be considered complete. Implementation design details, including approaches, vendor options, and OMS architecture, are covered in paper 2, sections 3.1-3.6. This section is focused exclusively on what validation must confirm.

D1: Event ingestion validation

What validation must confirm: The ingestion layer correctly handles the full range of ComposerX event conditions that will occur in production — not just the well-formed events used in functional testing.

The production risk specific to D1 is silent failure: malformed or edge-case events that are dropped without error, processed incorrectly, or trigger unexpected ingestion-layer behavior. Silent failure in D1 propagates to all downstream domains — a dropped event becomes a missed break, a misattributed position, or an audit gap. The validation test must therefore cover not only positive ingestion (well-formed events) but also negative conditions:

Test condition	What it confirms	Pass criterion
Well-formed events (all event types)	Baseline ingestion capability across all LedgerScan event types, including token transfer, settlement confirmation, wallet permission change, and lifecycle events.	All event types were correctly received, normalized, and delivered to downstream position management within expected latency.
Malformed events (schema violations, missing fields)	The ingestion layer rejects or quarantines malformed events without a silent failure; errors are logged and flagged.	Malformed events do not propagate to the position state. Error logged with sufficient detail for root cause analysis.
Duplicate events (same event ID, multiple deliveries)	Idempotency: duplicate events do not produce duplicate position updates.	Duplicate detected and suppressed; position state unchanged; deduplication log entry created.
Out-of-sequence events (later event arriving before earlier)	Sequence handling: late-arriving events are correctly ordered before the position update is applied.	Position state reflects the correct final state regardless of delivery order. Sequence resolution log entry created.
Rate limit responses	The ingestion layer handles ComposerX rate limit responses without data loss.	Rate limit encountered; ingestion paused and retried; no events lost during rate limit window.
Failover/connection loss	The Ingestion layer recovers from connection loss and replays missed events from the correct sequence point.	Connection loss detected; reconnection initiated; missed events replayed from correct sequence point without duplication.

D2: Dual-state position management validation

What validation must confirm: The dual-state position view is accurate and consistent under concurrent update conditions — not just under the sequential, well-spaced updates used in functional testing.

The production risk specific to D2 is timestamp-attribute errors under rapid concurrent updates. When an on-chain event and an off-chain batch update arrive close together, the position management layer may apply them out of order, resulting in a momentary or persistent position inconsistency. The validation test must cover:

Test condition	What it confirms	Pass criterion
Rapid sequential updates — on-chain then off-chain	The position state correctly reflects both updates in the correct order.	Final position matches expected state. No intermediate state persisted incorrectly.
Rapid sequential updates — off-chain then on-chain	Position state correctly reflects both updates when the off-chain batch arrives before on-chain confirmation.	Final position matches expected state. Batch updates are held in a pending state until on-chain confirmation is received or reconciled correctly after the fact.
Concurrent updates — on-chain and off-chain at the same timestamp	Tie-break logic resolves concurrent updates consistently and in accordance with DTC's record authority (LedgerScan is definitively authoritative).	Tie-break resolved in favor of LedgerScan record. Resolution logic is documented and repeatable.
Identifier translation accuracy	Network-native token IDs (Canton, DTCC AppChain and Stellar) are correctly mapped to CUSIP/ISIN for all eligible securities in the test portfolio.	Zero identifier translation errors across the test portfolio — resolution table complete and up to date.
Point-in-time position query	The historical position state can be reconstructed at any point in the event sequence.	Point-in-time query returns the correct position for all test scenarios. Reconstruction matches the expected state.

D3: Break detection and classification validation

What validation must confirm: The break classification framework correctly identifies and routes each of the break types that will occur in production. This is the most analytically demanding validation activity because it requires an explicit break taxonomy — a defined inventory of break types — before testing can begin.

Participants using the ZTAOP Break Taxonomy. (available as a companion document that defines 16 break types across four families) can use that inventory directly as the test input set. Participants developing their own taxonomy should ensure it covers at minimum the four break families below, as these represent the structurally distinct failure modes of a dual-record environment:

Break family	Representative break type	Validation test — inject and confirm
Timing breaks	On-chain settlement confirmed; off-chain OMS batch not yet updated. Position temporarily diverges.	Inject: on-chain confirmation without corresponding OMS update. Confirm: classified as a timing break; auto-monitored rather than escalated; resolves correctly when batch processes.
Authority breaks	DTC root-wallet override modifies a tokenized position without the participant's instruction. No corresponding OMS event.	Inject: synthetic root-wallet event via D5 simulation. Confirm: classified as an authority break (not timing break); routed to override handling workflow; evidence generated.
CA misapplication breaks	CA event applied to traditional record; tokenized record not updated within expected window.	Inject: CA event on traditional record only. Confirm: classified as a CA misapplication break (not a timing break); routed to the CA processing team with CA type and position size context.
Identifier breaks	A network-native token ID (Canton, DTCC AppChain or Stellar) for a newly issued tokenized security is not yet in the identifier resolution table.	Inject: event with unmapped token ID. Confirm: classified as identifier break; quarantined for manual identifier resolution; not miscategorized as timing break.

The classification test passes only when each injected break type is routed to the correct workflow — not merely detected. A break detector that raises every exception for manual review has not passed D3 validation. The test must confirm correct classification and correct routing.

D4: Corporate action processing validation

What validation must confirm: CA events are applied consistently across both the traditional and tokenized records for each CA type in the participant's tokenized position portfolio. The specific CA types to be tested depend on the portfolio — but the minimum set for any participant includes at least two CA types: one income event (dividend or coupon) and one corporate event (merger or rights issuance).

The production risk specific to D4 is one-sided CA processing: a CA event applied to the traditional record but not the tokenized record — or vice versa — produces a discrepancy that is more complex to resolve than a timing break because the event has already been financially processed on one side. The validation test must cover:

CA type	Validation test	Pass criterion
Dividend/coupon payment	Simulate dividend or coupon declaration against a dual-record position. Confirm credit applied to both records within the expected window.	Both records reflect dividend credit at the same position quantity. No one-sided processing. D3 does not flag a break.
Stock split	Simulate a 2-for-1 split against a dual-record equity position. Confirm the position quantity is doubled on both records.	Both records reflect a doubled quantity after the split effective date. Identifier mapping is updated if the token ID changes during a split.
Merger/acquisition	Simulate converting the acquiree position to the acquirer position on both records. Confirm that the old position is retired and the new position is created consistently.	Both records reflect the correct post-merger position. Old token ID retired. New token ID (if applicable) correctly mapped. No residual balance on old position.
Rights issuance	Simulate rights distribution against a dual-record position. Confirm the rights position created on both records.	Rights position created on both records at the correct ratio. Subscription elections are consistently recorded across records.

CA validation should be sequenced after D1 and D3 validation is complete — CA misapplication breaks cannot be classified correctly if break detection is not functioning. Defer D4 validation until DTCC publishes its CA processing framework; the validation test design should align with the published framework.

D5: Override detection validation

What validation must confirm: DTC root-wallet interventions are detected within expected latency, evidence is generated in a form appropriate for regulatory examination, and the audit trail correctly records the intervention as a root-wallet event — not as a participant-initiated event or a timing break.

Root-wallet override simulation is the only practical pre-production test for D5. Actual DTC root-wallet interventions are not available in a sandbox environment. The simulation must therefore replicate the LedgerScan event structure of a genuine root-wallet override — specifically, the event type, the position modification without a corresponding participant instruction, and the 'Conditions Requiring Reversal' designation from the NAL structure. The validation test must confirm:

Test dimension	What it confirms	Pass criterion
Detection latency	The time between a root-wallet event entering the LedgerScan event stream and its detection by the participant's monitoring layer.	Detection within the participant's defined latency threshold (to be determined based on position management cycle; typically measured in seconds to minutes, not hours).
Event classification	The override event is classified as a root-wallet intervention — not as a participant instruction, a timing break, or an unclassified exception.	Override event type correctly identified. Routed to override handling workflow, not standard break resolution workflow.
Evidence generation	The evidence artifact generated for the override event contains: event timestamp, DTC authority identifier, position before override, position after override, and the designation 'Conditions Requiring Reversal'.	Evidence artifact complete. All required fields are populated. Artifact stored in the audit trail within the expected latency of detection.
Internal position update	The participant's internal position record is updated to reflect the override within the expected reconciliation window.	Internal position updated. Prior position state is preserved in the audit trail for point-in-time reconstruction — no downstream settlement instruction is generated based on the pre-override position.

D6: Audit trail and regulatory evidence validation

What validation must confirm: The audit trail is complete, queryable, and supports point-in-time position reconstruction for any position and any time range within the production period. This test is the final gate before production entry and the one most directly relevant to regulatory examination readiness.

The audit evidence completeness test should be run after all other domain validation activities are complete — because the audit trail must capture events from all six domains, and it can only be confirmed complete after those events have been generated. The test sequence:

Test step	Action	Pass criterion
1. Generate a complete event sequence	Run a defined test sequence covering: token transfer (D1), position update (D2), timing break and resolution (D3), CA event and confirmation (D4), root-wallet override (D5).	All events are generated and delivered to the ingestion layer. Event count confirmed against expected total.
2. Query audit trail completeness	Query the audit trail for the test position across the full event sequence time range.	All events are present in the audit trail. No gaps in the event sequence. Each event is correctly attributed to its source (participant instruction, DTC action, CA event, or root-wallet override).
3. Point-in-time reconstruction	Reconstruct the position at five points in the test sequence: before the CA event, after the CA event, before the root-wallet override, after the root-wallet override, and at the end of the sequence.	Reconstructed position matches the expected state at each of the five time points. Reconstruction relies solely on the audit trail, not on live position data.
4. Regulatory evidence export	Export the audit trail for the test position in the format intended for regulatory examination response.	Export complete. All events are present. Format consistent with what would be provided in response to an SEC examination request. Export producible within the participant's defined regulatory response time.

05 Validation journey — from implementation to production

The validation journey is sequential. Domains 1-3 establish pilot-ready status (Level 3); Domains 4-6 establish production-ready status (Level 4). Each gate is not a checkpoint to be ticked, but an evidence milestone: the objective is not test execution alone, but the generation of artifacts that demonstrate operational control effectiveness for sign-off and audit review. A firm that has completed testing without producing reviewable evidence has not completed validation.

Exhibit 2. Validation journey from implementation to production

06 Validation maturity milestones: The production readiness gates

The maturity model for the series is defined in full in Paper 2, Section 4 (Levels 1–5). This section reproduces only Levels 3 and 4 — the two levels that define the validation gates separating Pilot Ready from Production Ready. Level 3 is achieved through implementation; Level 4 requires successful completion of a structured validation program.

Level 3 Pilot ready	The organization has implemented basic capabilities in Domains 1, 2, and 3 in a test or UAT environment, using the LedgerScan sandbox (or equivalent synthetic environment) as the event source. It can ingest token events, maintain a dual-state position view, and detect and classify position discrepancies at a basic level. **Validation gates required for Level 3:** D1 event ingestion test (all six conditions in Section 4.1) · D2 concurrent update test · D3 break injection test (all four break families in Section 4.3). Documented break resolution workflow tested with synthetic data. **Not yet required at Level 3:** D4 CA simulation · D5 override simulation · D6 audit evidence completeness test. Domain 6 audit trail may be at a basic level only.

Level 4 Production ready Pilot ready	The organization has implemented production-grade capabilities across Domains 1–3 and Domain 5, with documented operational runbooks and trained staff. Domain 4 is addressed at least for the CA types in the tokenized portfolio. Domain 6 provides queryable records of all tokenized position events. **Validation gates required for Level 4:** All Level 3 gates · D4 CA simulation (for all CA types in the tokenized portfolio) · D5 root-wallet override simulation (detection latency, evidence generation, and audit entry confirmed) · D6 audit evidence completeness test (point-in-time reconstruction confirmed across a full event sequence including CA and override events). **Additional requirement:** The organization can demonstrate control effectiveness to internal and external auditors using the evidence artifacts generated by the validation program. Auditor briefing completed before production entry.

Regression gate (all levels): After any API specification update or ComposerX version change, rerun the full validation suite for all domains at the current maturity level. A regression that reduces classification accuracy or audit-trail completeness requires root cause resolution before production promotion or continuation.

07 Sourcing validation capability

Validation capabilities may be sourced through three routes. The choice depends on the participant's timeline, the availability of DTCC's sandbox environment, and the internal QA capacity available alongside the implementation program.

Source	What it provides	Limitation	Best for
DTCC's own sandbox environment	A limited synthetic LedgerScan event stream for integration testing. The most reliable source of well-formed ComposerX event data is because it reflects the actual API implementation.	Provides well-formed events only. Does not support malformed event injection, break injection, CA simulation, root-wallet override simulation, or regression testing. Scope is limited to D1 and D2 basic validation.	All participants as the baseline for D1 event ingestion testing. Should be the first validation environment established, not the only one.
Internal QA capability	Full control over test design, scenario coverage, and regression cadence. It can be extended to cover all six domains if built with sufficient domain knowledge.	Requires post-trade operations and DLT expertise to build correctly. Building a break injection framework, CA simulator, and override simulator alongside the operational implementation is a significant parallel workstream. Risk: QA built by the same team as the implementation may share the same assumptions and blind spots.	Tier-1 participants with established digital assets, QA functions, and sufficient resourcing to treat validation as a parallel workstream rather than an afterthought.
Specialist validation accelerators	Pre-built, reusable components for synthetic event generation, break injection, CA simulation, override simulation, and audit evidence validation. Can cover all six domains and produce evidence artifacts in the format required for operational sign-off and audit review.	The market for purpose-built DTC tokenization validation tooling is nascent. Apply the same due diligence as for any emerging vendor: assess reference implementations, coverage scope, and alignment to current DTCC API specifications.	Mid-tier participants that cannot resource a full internal QA build, and participants that need to achieve Level 4 maturity on a compressed timeline before the first production wave.

On combining sources: Most participants will use a combination. DTCC's sandbox provides the well-formed event baseline (D1). Internal QA handles D2 concurrent update testing and D3 break injection for the participant's specific break taxonomy. A specialist accelerator covers D5 override simulation (where internal build requires the most domain knowledge) and D6 audit completeness testing. The modular structure of Exhibit 1 is designed to support this combination — each domain test can be sourced independently.

08 Recommended validation actions

The following validation roadmap runs in parallel with implementation (Paper 2, Phases 3-4) rather than sequentially after it. Beginning validation design before implementation is complete is not premature but necessary, because the break taxonomy (needed for D3 validation design) and the CA type inventory (needed for D4 validation design) must be finalized before implementation is complete.

Stage	When	Actions
Stage 1 Validation design	Parallel with Paper 2 Phase 3 (during implementation)	1. Finalize break taxonomy — at minimum four break families (Section 4.3). This is a prerequisite for D3 validation design and break injection tooling configuration. 2. Finalize CA-type inventory for the tokenized portfolio. This determines which CA simulation scenarios are required for D4 validation. 3. Define the detection latency threshold for D5 override detection. This threshold must be set before the D5 simulation test can be designed. 4. Establish DTCC sandbox access — credentials, event stream configuration, and event type coverage confirmation. 5. Select validation sourcing approach for each domain (DTCC sandbox / internal QA / specialist accelerator). Document sourcing decisions and rationale.
Stage 2 Pilot-ready validation (L3)	Before Level 3 maturity claim/UAT sign-off	6. Execute D1 ingestion test — all six conditions (Section 4.1). Document results. Execute 7. D2 concurrent update test (Section 4.2). Document results. 8. Execute D3 break injection test — all four break families (Section 4.3). Document classification accuracy and routing correctness. 9. Document break resolution runbooks for each break type confirmed in D3 testing. 10. Brief internal audit on Level 3 validation evidence. Confirm that evidence artifacts meet the audit's documentation requirements.
Stage 3 Production-ready validation (L4)	Before production onboarding	11. Execute D4 CA simulation for all CA types in the tokenized portfolio (Section 4.4). Document the consistency of application across both records. 12. Execute D5 root-wallet override simulation (Section 4.5). Confirm detection latency, evidence generation completeness, and audit trail entry. 13. Execute D6 audit evidence completeness test — full event sequence including CA and override events (Section 4.6). Confirm point-in-time reconstruction accuracy. 14. Run regression suite after any API specification update received between Stage 2 and production entry. 15. Brief internal and external auditors on Level 4 validation evidence. Confirm the audit trail is in the form required for regulatory examination response. 16. 16. Document residual gaps — any test condition that could not be fully validated due to API specification incompleteness — and confirm these are monitored as production begins.
Stage 4 Production regression and monitoring	Continuously post-production	17. Run the regression suite after each ComposerX version change or API specification update. 18. Rerun D3 break injection testing if the break taxonomy is updated based on production experience. 19. Rerun D4 CA simulation if new CA types are added to the tokenized portfolio. 20. Rerun D5 override simulation if DTCC's published quarterly reporting indicates root-wallet intervention frequency has changed materially from the pre-production assumption. 21. Extend D6 audit completeness testing to cover any new event types introduced by DTCC program evolution (new eligible securities, new blockchain networks).

09 What to monitor: Two validation-specific areas

The full monitoring framework for the series — covering ComposerX API specifications, CA processing framework, regulatory evidence expectations, vendor platform readiness, and root-wallet intervention frequency — is provided in paper 2, Appendix C. Two of those five areas have direct and immediate implications for validation scope and should be monitored by the validation team as well as the implementation program team.

ComposerX API specifications

Authentication mechanisms, event schemas, rate limits and failover behaviour for LedgerScan, Factory and CMP were not fully published as of August 2026; with launch targeted for Q4 2026, participants should re-confirm current availability on DTCC's developer portal, as specifications may now be partially or fully released. This directly affects the scope of D1 ingestion validation: test conditions for malformed events, rate-limited responses, and failover behavior cannot be fully defined until the API specification is published. Validation design in these areas should be treated as provisional until complete API documentation is available.

Action: Designate a validation owner who tracks DTCC's developer portal for API documentation updates. When updates are published, review for impact on D1 test conditions and trigger a regression run before production promotion

Root-wallet intervention frequency

The anticipated frequency of DTC root-wallet interventions during the pilot is unknown. High frequency changes the D5 validation requirements substantially — a participant that expects rare manual overrides may need to revisit its override-detection architecture if early production data shows a materially higher frequency. arly production data on override frequency will become available through DTCC's quarterly SEC reporting once live trading begins (targeted Q4 2026); the first such report is expected after the initial production quarter.

Action: After DTCC's first quarterly SEC report covering root-wallet interventions, review the observed frequency against the pre-production assumption used in the D5 validation design. If the frequency is materially higher than assumed, rerun the D5 override simulation with updated latency thresholds and rebrief auditors on the evidence artifacts.

10 Conclusion

The fundamental distinction this paper rests on is simple: implementation and validation are not the same activity. A participant who completes implementation across all six operational capability domains has built what is needed to operate in production. A participant who completes validation has confirmed that what was built will actually work when exposed to production conditions for the first time.

The validation activities in this paper — ingestion testing, break injection, CA simulation, override simulation, and audit completeness testing — are not a compliance burden. They are the mechanism by which implementation confidence becomes operational confidence. The evidence artifacts they produce are the difference between entering production knowing you are ready and entering production hoping you are ready.

The DTC tokenization program is the first instance in US capital markets where security entitlements will be maintained simultaneously in traditional book-entry form and as tokenized records on a distributed ledger. There is no institutional memory of what breaks look like in this environment, how overrides appear in event streams, or what a complete audit trail for a dual-record position looks like. The validation program is how participants build that knowledge before production begins — rather than discovering it through production failures.

For the implementation framework that precedes this validation guide, see paper 2 — Operational Readiness Framework for Tokenized Securities. For the executive case for readiness, see paper 1 — Tokenized Securities Readiness: What DTCC Participants Need to Know.

A system that passes functional testing demonstrates that the capability has been built.
A system that passes validation demonstrates that the capability works.
Production readiness is not declared at implementation completion. It is evidenced at validation complete.

This is paper 3 of a three-paper series — How to Validate.

Paper 1 (Why it Matters): Executive framing, structural analysis, heatmap, and first actions.

Paper 2 (How to Prepare): Six-domain operational framework, maturity model, implementation approaches, and risk analysis.

To discuss validation program design: Zensar BFS Capital Markets offers pre-production validation workshops and ZTAOP QA accelerator engagements for DTC participant firms. Contact info@zensar.com. Participants are also encouraged to engage directly with DTCC's working group program and developer portal for current API specifications and sandbox access.

Appendix A — Zensar ZTAOP QA: One example of specialist validation capability

This appendix describes the readiness validation and QA components of Zensar's Tokenized Asset Operations Platform (ZTAOP) as one example of the specialist validation accelerator approach discussed in Section 7. It is presented as an appendix to preserve the analytical independence of the core validation framework. Evaluate ZTAOP QA alongside the sourcing options in Section 7 and conduct independent due diligence.

ZTAOP QA is distinct from the operational components of ZTAOP (described in Paper 2, Appendix A, Section A). It can be used independently of the ZTAOP operational platform — participants who have implemented operational capability through internal build or an established vendor platform can still use ZTAOP QA for pre-production validation.

ZTAOP QA components

Component	What it validates	Domain	Evidence artifact
Synthetic ComposerX event generation	Produces well-formed, malformed, duplicate, out-of-sequence, and edge-case LedgerScan events to confirm that event ingestion handles all anticipated conditions. Covers all six test conditions in Section 4.1.	**D1**	Ingestion log confirming event type, condition, processing result, and latency for each test event. Pass/fail by condition.
Break injection framework	Injects known break types — timing breaks, root-wallet divergences, CA misapplication, identifier mismatches — to validate that detection, classification, and routing logic handles each type correctly. Configurable to the ZTAOP Break Taxonomy (16 types, 4 families) or the participant's own taxonomy.	**D3**	Classification accuracy report: break type injected vs. break type detected vs. routing destination. Misclassification log for root cause analysis.
Corporate action simulation	Generates CA events — dividend, coupon, merger, split, rights issuance — against dual-record positions to confirm consistent application across both records and absence of one-sided processing. Configurable to the participant's CA type inventory.	**D4**	CA consistency report: CA type, position before, position after on each record, consistency confirmed or discrepancy flagged — one-sided processing log.
Root-wallet override simulation	Generates synthetic DTC root-wallet intervention events replicating the LedgerScan event structure of a genuine override, including 'Conditions Requiring Reversal' designation. The only practical pre-production test for D5.	**D5**	Override detection report: event timestamp, detection latency, evidence artifact completeness, audit trail entry confirmation. Latency against the participant's defined threshold.
Audit evidence validation	Queries the audit trail across a defined event sequence and time range, confirming completeness, correct event attribution, and point-in-time position reconstruction accuracy. Runs the four-step test sequence in Section 4.6.	**D6**	Audit completeness report: events expected vs. events present, attribution accuracy, point-in-time reconstruction results at five specified time points. Regulatory evidence export sample.
Regression harness	Reruns all validation scenarios following API specification updates or ComposerX version changes, confirming that integration behavior has not regressed. Produces a delta report comparing current results with the prior-validated baseline.	**All**	Regression delta report: scenario-by-scenario comparison against prior baseline. New failures are flagged for root cause analysis before promotion to production.

Two ZTAOP capabilities introduced in Paper 2, Appendix A support these QA components directly. ZenseAI.data - the context store holding the ISO 20022 token-event taxonomy, identifier-resolution tables and corporate-action enrichment data - supplies the reference context that D2 identifier-translation and D4 corporate-action validation are tested against; a resolution table that is incomplete or stale surfaces as D2 and D4

failures. ZenseAI.Engg - the domain-specific agent that triages position discrepancies - is itself an object of validation: break-injection testing (D3) confirms the agent classifies and routes each injected break correctly, not merely that rule logic fires. Validating these layers confirms that intelligence added on top of the participant's existing OMS and audit systems behaves correctly, without replacing them.

ZTAOP QA is positioned as a pre-production validation tool, not a monitoring or surveillance product. Its outputs are evidence artifacts intended to support operational sign-off and internal audit review. As with all ZTAOP components, it is not a regulated financial service and does not constitute compliance certification.

For operational accelerator components (LedgerScan ingestion, dual-state position management, break detection, override detection, audit trail): see paper 2, Appendix A, Section A.
For the commercial model, indicative ranges are provided in paper 2, Appendix A.

ZTAOP QA is distinct from the operational components of ZTAOP (described in Paper 2, Appendix A, Section A). It can be used independently of the ZTAOP operational platform — participants who have implemented operational capability through internal build or an established vendor platform can still use ZTAOP QA for pre-production validation.

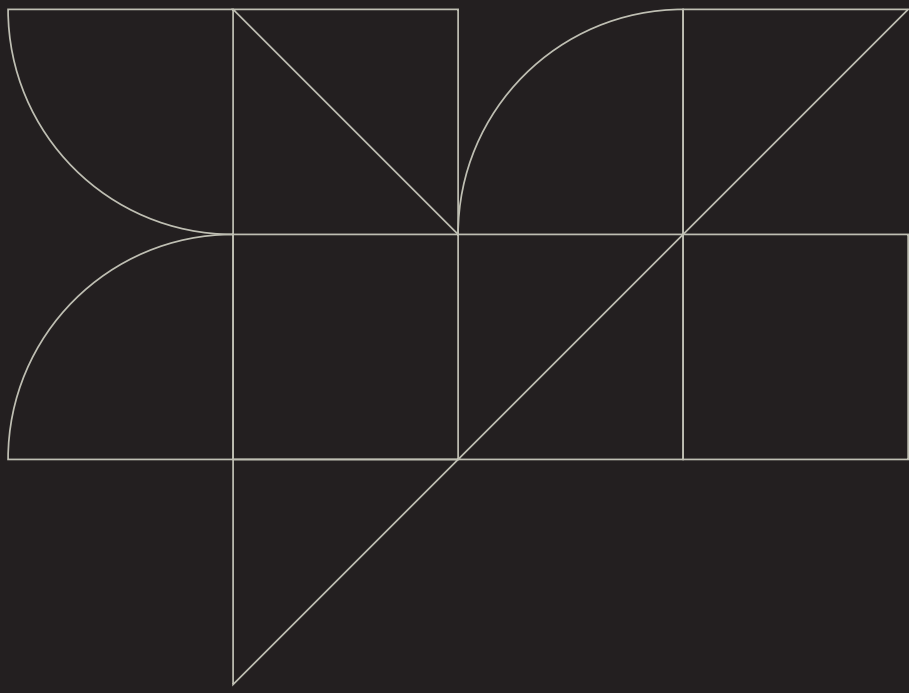
References

[1] SEC Division of Trading and Markets, No-Action Letter to The Depository Trust Company, December 11, 2025. Available at: SEC.gov. Primary regulatory document governing the DTC tokenization pilot.
[2] DTCC Press Release, 'DTCC Advances Development of New Tokenization Service, Convenes 50+ Firms to Drive Digital Assets Adoption,' May 4, 2026.
[3] DTC ComposerX platform documentation, as described in the SEC NAL request letter and DTCC public communications.
[4] World Economic Forum / Accenture, 'Tokenization in Financial Services: Unlocking the Next Wave of Innovation,' 2024–2025.
[5] International Monetary Fund, Working Paper on Tokenized Finance, April 2026.
[6] Bank for International Settlements, various publications on DLT in financial market infrastructure.
[7] ZTAOP Break Taxonomy.

© 2026 Zensar Technologies. Published for industry discussion. Not investment, legal, or regulatory advice.

Author:

Kamlesh Kosare
Global Head of Business Consulting,
Banking & Financial Services - Capital Markets,
Industry Solutions Group (ISG)



zensar
An RPG Company

At Zensar, we're 'experience-led everything.' We are committed to conceptualizing, designing, engineering, marketing, and managing digital solutions and experiences for over 145+ leading enterprises. Using our 3Es of experience, engineering, and engagement, we harness the power of technology, creativity, and insight to deliver impact.

Part of the $4.8 billion RPG Group, we are headquartered in Pune, India. Our 10,000+ employees work across 30+ locations worldwide, including Milpitas, Seattle, Princeton, Cape Town, London, Zurich, Singapore, and Mexico City.

For more information, please contact: **info@zensar.com | www.zensar.com**