TUSIRO

Financial Systems Integrity Platform

Technical Dossier

Execution Correctness, Shared Semantics,
and Evidence Architecture

Architecture, semantic model, evidence system,
and correctness verification framework for
market-connected execution infrastructure.

CONFIDENTIAL | FOR TECHNICAL AND PILOT EVALUATION

This document contains proprietary architectural and design information.
Distribution without authorization is prohibited.

Contents

1 Execution Correctness as an Engineering Discipline

2 The Shared Semantic Core

3 Canonical Event Model and Source Discipline

4 Order Lifecycle Contracts

5 Evidence Ledger and Deterministic Replay

6 Correctness Audit Reports

7 Delivery Governance and Evidence Bundles

8 Commercial Boundaries

9 Implementation Status and Acceptance Evidence

10 Current Commercial Entry Point

This dossier describes the architecture, semantic model, evidence system, and correctness verification framework of the Tusiro platform. It is intended for engineering leads, system architects, and technical decision-makers evaluating execution integrity solutions for market-connected infrastructure.

1 Execution Correctness as an Engineering Discipline

Financial execution systems accumulate risk not primarily through catastrophic failure, but through the slow erosion of behavioral guarantees. Orders are acknowledged but not tracked. Fills exceed authorized quantities without detection. Cancel requests vanish into protocol gaps between systems. Lifecycle states become inconsistent across subsystems without anyone noticing until the damage surfaces in reconciliation, or worse, in production.

Tusiro treats execution correctness as an engineering discipline, not a monitoring exercise. The platform evaluates whether an execution system behaved according to declared rules, documents what the evidence supports and what it contradicts, and produces audit-grade evidence artifacts that survive external review.

What Tusiro evaluates

- Whether fill events were preceded by order acknowledgement evidence
- Whether cumulative fill quantities stayed within authorized baselines
- Whether cancel acknowledgements were preceded by cancel requests
- Whether event timestamps per order lifecycle were monotonically ordered
- Whether every initiated lifecycle reached a documented terminal state or was explicitly flagged as unresolved

These are not heuristics or anomaly scores. Each evaluation has a formal contract with defined inputs, exhaustive result conditions (PASS, FAIL, UNKNOWN, SKIPPED), and explicit evidence requirements. When evidence is insufficient, the system reports UNKNOWN rather than inferring safety from absence.

Three product surfaces

Tusiro is designed around three surfaces that share a single semantic core:

Surface	Purpose	Status
A	Execution Correctness Verification — evaluates historical evidence offline against formal contracts	**Available**
B	Adversarial Resilience Testing — tests whether systems survive semantically hostile but protocol-valid conditions	Planned
C	Runtime Protection — bounded containment during trust degradation events	Planned

All three surfaces consume the same canonical event model, the same contract families, and the same evidence ledger.

2 The Shared Semantic Core

Tusiro's architecture is built around one permanent nucleus that all surfaces, sources, and contract families must share. This prevents the platform from fragmenting into disconnected tools as it scales.

Layer	Responsibility
Canonical Event Model	Shared vocabulary between raw sources and semantic evaluation. Source-agnostic, typed, deterministically sequenced.
Contract Layer	Formal correctness rules defining expected lifecycle behavior. Source-independent specifications with explicit applicability gates.
Evaluation Engine	Processes canonical events against contract rules. Manages state mutation, contradiction detection, phase transitions, and verdict emission.
Evidence Ledger	Append-only, typed persistence of every fact, finding, and decision. The single source of truth for all downstream consumers.
Dispatcher	Narrow routing layer that separates evaluator truth from downstream actionability. Tags outputs without reinterpreting them.

Anti-fragmentation law

Every surface, every source, and every customer engagement must use the same canonical event model, the same contradiction taxonomy, the same phase model, the same evidence ledger schema, and the same report contract. The platform does not permit separate schemas, separate contradiction logic, separate identity models, or one-off semantics hidden inside connectors. If that discipline breaks, the platform stops being a platform.

3 Canonical Event Model and Source Discipline

Raw execution data is messy, inconsistent across venues and brokers, and frequently missing fields that downstream analysis assumes are present. Tusiro addresses this through a three-stage canonical event lifecycle and a strict source honesty doctrine.

Three-stage event lifecycle

Stage	Type	Properties
1	**Draft**	Output of connector mapping. Kernel fields populated from source. Not validated, not persisted, not sequenced.
2	**Validated**	Passed schema validation and pre-sort filtering. Structurally valid and sortable. Not yet in the ledger.
3	**Record**	Persisted to the evidence ledger. Carries a monotonic sequence number and run identifier. Immutable.

Source honesty doctrine

A source honesty violation is defined as any mapping, inference, or field population that asserts evidence the raw source does not contain. This is a testable property enforced at the connector level, not a cultural aspiration. When evidence is missing, Tusiro documents the gap. It does not fabricate substitutes.

Source profiles

Every data source requires a versioned source profile that declares its capabilities, identity fields, timestamp semantics, data quality thresholds, and event-family mapping. The profile is validated before every pipeline run and becomes part of the run manifest. Source-conditioned evaluation rules ensure that the platform adapts its expectations to what the source can actually provide, rather than assuming capabilities that do not exist.

- Capability flags: explicit boolean declarations for ACK, reject, cancel-request, replace, snapshot
- Identity profile: which order identifiers are available, which are nullable, which are never present
- Adapter payload schema: typed, validated, with semantic-smuggling-risk annotations per field
- Event-family mapping: how raw source types map to canonical families, with unmappable types producing explicit rejections

4 Order Lifecycle Contracts

A contract family is an abstract, source-independent set of correctness rules defining expected behavior for a category of financial execution lifecycle. Contracts are formal specifications, not heuristics. Each rule has defined inputs, exhaustive result conditions, explicit applicability gates, and documented interactions with the contradiction system.

Order Lifecycle Consistency (OLC)

Rule	Evaluates	Timing
ACK Before Fill	Fill events preceded by opening evidence	Immediate
No Overfill	Cumulative fill quantity within authorized baseline	Immediate
No Orphan Cancel	Cancel acknowledgements preceded by cancel requests	Immediate
Monotonic Ordering	Timestamps per order lifecycle do not run backward	Immediate
Terminal State Integrity	Every initiated lifecycle reaches terminal state or is documented as unresolved	Deferred

Four verdict outcomes

Every rule evaluation produces exactly one of four results. **PASS** means sufficient evidence was present and no violation was detected. **FAIL** means sufficient evidence was present and a violation was detected. **UNKNOWN** means the rule applies but evidence is insufficient to determine compliance. **SKIPPED** means the rule did not apply to this context due to source capability gaps or lifecycle classification exemptions.

UNKNOWN never collapses into PASS. This is a non-negotiable architectural constraint. When the platform cannot determine compliance, it says so rather than inferring safety from absence of evidence.

Contradiction and phase model

Contradictions are structural or semantic impossibilities detected in the evidence stream. They are separate from rule verdicts and serve a different purpose: contradictions affect trust phase, while verdicts do not. The three-layer phase model tracks readiness (bootstrap vs active observation), shared-scope trust, and per-instance trust independently. Trust can only degrade within a run, never recover. This monotonicity ensures that downstream consumers can rely on trust assessments without fear of retroactive revision.

5 Evidence Ledger and Deterministic Replay

The evidence ledger is not a log. It is the typed, append-only, immutable memory of everything Tusiro observed, concluded, and decided during a pipeline run. Every canonical event, every contradiction, every phase transition, every verdict, every derived claim, and every dispatch decision is persisted as a typed record. Both accepted and rejected verdicts are preserved, keeping the causal chain complete.

Six record families

- **Canonical events** — the normalized, sequenced, validated evidence stream
- **Contradiction records** — structural and semantic impossibilities with phase impact
- **Phase transitions** — trust degradation events across three layers
- **Verdict records** — formal rule evaluation outcomes with full evidence payload
- **Derived claims** — typed pipeline assertions (rejections, coverage summaries, bootstrap status)
- **Dispatch decisions** — routing and actionability tags for every evaluator output

Run manifests and replay attestation

Every pipeline run produces a manifest containing input digests (SHA-256), configuration digests (source profile, contract bundle), environment metadata, and a complete pipeline output summary. Replay attestation compares two manifests to verify deterministic reproduction: same inputs, same configuration, same code, same outputs. When all match, the attestation is REPLAY_CONFIRMED.

The golden replay suite verifies determinism across three layers: event ordering (sequence assignments), identity binding (lifecycle chain resolution), and ledger output (content hashes over canonical serialization). Adversarial test traces with oracle-grade expected outcomes verify correct behavior on known-difficult edge cases.

Canonical serialization

For content hashing and comparison, Tusiro uses a frozen serialization format: JSON UTF-8, alphabetical key ordering at all levels, explicit nulls, integers without leading zeros, decimals as string representation, booleans lowercase. Per-record SHA-256 hashes are aggregated per type, then into a single ledger content hash. This makes bitwise-identical comparison possible across runs and environments.

6 Correctness Audit Reports

The audit report is a governed consumer of the evidence ledger. It reads from ledger queries only, produces human-readable output, and never recomputes or reinterprets evaluator truth. The report contract specifies ten mandatory sections, twelve hard bans, and explicit disclaimers.

Report structure

#	Section	Content
1	**Run Metadata**	Pipeline identity, timestamps, record counts, run status
2	**Source Profile Summary**	Source capabilities, honesty notes, declared limitations
3	**Coverage and Capability Limits**	Per-rule applicability, evaluated counts, skip reasons
4	**Bootstrap and Scope Status**	Observation window, preexisting order classification
5	**Contradiction Summary**	Structural and semantic impossibilities, phase impact
6	**Verdict Summary**	Per-rule PASS/FAIL/UNKNOWN/SKIPPED with routing status
7	**Suppressed and Superseded Detail**	Non-reportable verdicts with documented reasons
8	**Highest Finding Density Instances**	Neutral-language concentration analysis
9	**Known Limitations**	Source gaps, exemption impact, data quality
10	**Claim Boundary Warnings**	What the report does and does not prove

Executive summary governance

Every report includes a governed executive summary with fixed fields: run outcome, computed coverage confidence, material findings count, inconclusive count, trust impact, and a recommended next action selected from a fixed menu. The executive summary does not contain custom narrative, does not use the words "safe" or "compliant," and does not recommend specific actions.

7 Delivery Governance and Evidence Bundles

Tusiro's delivery layer ensures that evidence artifacts reach external recipients in governed, integrity-verified packages. Bundle assembly and validation are automated and enforced through a ten-item delivery checklist.

Three delivery tiers

Tier	Bundle Type	Contains	Use Case
1	**Audit Summary**	Executive summary, capability table, claim boundary, disclaimers	Overview evaluation
2	**Evidence-Backed**	Full report, redacted manifest, replay attestation, disclaimers	Standard commercial audit
3	**Forensic Review**	Everything in Tier 2 plus full manifest, ledger dump, diagnostics	Scoped investigation under SOW

Bundle integrity

Every bundle includes a manifest with per-artifact SHA-256 digests, a bundle-level integrity hash, and explicit inclusion/exclusion records documenting what is present and what was deliberately omitted. The ten-item delivery checklist verifies tier alignment, run validity, quarantine status, attestation presence, disclaimer completeness, and SHA-256 integrity before any bundle is released.

Customer-safe mode

When activated, customer-safe mode anonymizes order lifecycle identifiers in the report while preserving all analytical content. The anonymization mapping table is never included in any delivered bundle. This is an enforced boundary, not a configuration guideline.

8 Commercial Boundaries

Tusiro maintains explicit boundaries to prevent the platform from drifting into a consulting practice. These boundaries are architectural constraints, not business preferences.

What Tusiro delivers

- Execution correctness audits backed by typed evidence and deterministic replay
- Governed reports with explicit source-capability disclosure
- Evidence bundles with cryptographic integrity verification
- Reproducible pipeline runs with manifest and attestation artifacts

What Tusiro does not deliver

- Custom rule development per customer
- Manual interpretation of findings as a service
- Regulatory certification or compliance sign-off
- Ongoing monitoring or real-time alerting
- Data cleaning or correction
- Blame attribution or fault assignment

Claim boundary

Every Tusiro report carries explicit claim boundary language. The report does not prove that raw source data is complete, accurate, or tamper-free. It does not prove that the evaluated system is correct in any absolute sense. It does not prove that all possible violations were detected. UNKNOWN results do not imply safety. PASS results do not imply guaranteed correctness. Coverage is bounded by declared source capabilities.

9 Implementation Status and Acceptance Evidence

Tusiro is not a specification-only project. The architecture described in this dossier is implemented, tested, and exercised through a complete delivery workflow. The following capabilities are operational:

- **CI enforcement gates operational** — automated import boundary enforcement, frozen semantic identifier drift detection, and report boundary verification run on every code change
- **Report generator implemented and semantically validated** — produces governed Markdown audit reports per the report contract, with numerical reconciliation verified against ledger truth across all report sections
- **Golden replay suite built and passing** — three-layer determinism verification (event ordering, identity binding, ledger output) across full production datasets and adversarial test traces with oracle-grade expected outcomes
- **Evidence bundle assembly and validation implemented** — all three delivery tiers assemble correctly, ten-item delivery checklist automated, bundle SHA-256 integrity verified, customer-safe anonymization operational
- **Internal dry delivery completed** — full end-to-end pipeline-to-bundle workflow exercised, including replay attestation, side-by-side standard and customer-safe comparison, and sanity observations on operational characteristics
- **Source profile discipline operational** — explicit YAML source profiles validated against frozen schema with automated test coverage
- **264 automated tests passing** — covering schema validation, evaluator logic, dispatcher routing, golden replay, bundle assembly, delivery workflow, and source profile validation

The platform has been validated against a production-grade dataset of 581,030 exchange-side order book events, producing 354,865 verdict records, 13,522 contradiction records, and a complete evidence ledger of 1,317,818 typed records. Coverage confidence on the reference run is HIGH (99.999% of applicable rules evaluated, 2.59% inconclusive).

10 Current Commercial Entry Point

Available today

Tusiro offers Tier 2 evidence-backed execution correctness audits as the primary commercial entry point. A Tier 2 engagement produces a governed audit report, a redacted run manifest, a replay attestation, and a cryptographically verified evidence bundle. No custom rule development, no manual interpretation, and no ongoing monitoring commitment.

First pilot structure

Element	Detail
Scope	One source, one contract family (OLC), one evaluation run
Deliverable	Tier 2 evidence-backed audit bundle with replay attestation
Duration	Fixed-term, typically 2–4 weeks from data receipt to delivery
Customer provides	Historical execution data export (CSV or equivalent) with order lifecycle events
Tusiro provides	Source profiling, pipeline run, governed report, evidence bundle, technical briefing
What it proves	Whether the evaluation framework produces actionable findings on the customer's actual execution data

Ideal pilot profile

- Broker, trading desk, or venue operator with execution infrastructure they need to verify
- Access to historical order lifecycle data (submit, acknowledge, fill, cancel events) in exportable format
- Technical counterpart who can evaluate audit findings and evidence artifacts
- Interest in repeatable, governed audit capability rather than one-off consulting engagement

What a pilot is not

A pilot is not an open-ended consulting engagement, a custom rule development project, a regulatory certification process, or a runtime monitoring setup. It is a fixed-scope evaluation proving that Tusiro produces governed, reproducible findings on the customer's actual execution data.

To evaluate Tusiro for your execution infrastructure, request a technical briefing or pilot engagement.

contact@tusiro.com

This document is provided for technical and pilot evaluation purposes. It does not constitute a commercial offer, regulatory certification, or guarantee of system capability. Tusiro is a product of ongoing engineering. Capabilities described herein reflect the platform's architectural design and verified implementation.