CRIITS

A Transaction-Agnostic State Machine for Reliable End-to-End Processing

Author: Ankit Malhotra

Version 1.0 — Date: 16 September 2025

Abstract

Modern transactional systems span multiple services and networks. Failures, retries, and out-of-order delivery make it hard to guarantee that no transaction is missed. CRIITS defines four canonical phases—CR (Created), I (Initiated), I (0..n Intermediary), and TS (Terminal State)—plus operating invariants, a reference architecture, and a polling & reconciliation strategy to deliver observability and finality across domains.

License: CC BY 4.0 | **Contact**: ankit@example.com | **DOI**: (to be minted)

Contents

–Introduction

Distributed systems push business logic across APIs, queues, databases, and third-party gateways. Each hop introduces places to lose context or double-act on messages. Patterns like Sagas, Event Sourcing, and CQRS help, but teams still reinvent status semantics and pollers per use case. CRIITS proposes a small, standard state machine and operating convention to unify tracking, bound failure handling, and make “missed transaction” a measurable, alertable condition—remaining agnostic to domain (payments, KYC, logistics, content moderation, etc.).

–The CRIITS Model

–Canonical States

- **CR — Created**: Transaction accepted by the system of record (SOR). ID minted, minimal validation passed, persistence guaranteed.
- **I — Initiated**: Execution has started (request sent to downstream or workflow engaged).
- **I — Intermediary (0..n)**: Checkpoints representing externally verifiable progress (e.g., 3-DS required, vendor acknowledged, shipment picked).
- **TS — Terminal State**: Finality with mutually exclusive outcomes: TS.SUCCEEDED, TS.FAILED, TS.CANCELED, TS.EXPIRED.

–Invariants

1. **Monotonicity**: forward-only transitions CR → I → I* → TS.
2. **Completeness**: every transaction eventually reaches a single TS.
3. **Idempotency**: replaying the same transition is safe.
4. **Auditability**: transitions are append-only with causality metadata.

–State Machine Diagram

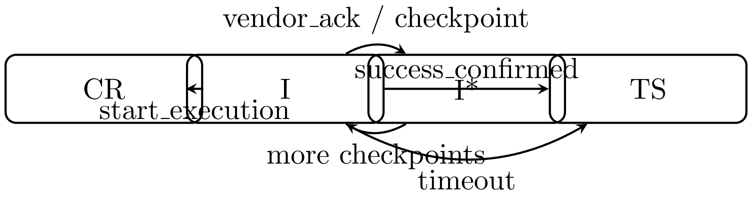

Figure 1: CRIITS canonical state progression with example events.

–Reference Architecture

Core components: Transaction Gateway, Orchestrator/Workers, Append-only Event Log, State Projection, Dispatchers (webhook emitter + outbox), Poller/Reconciler, Dead-Letter & Triage.

–Component Diagram

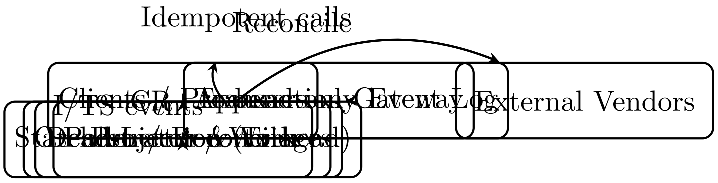

Figure 2: Reference architecture for CRIITS deployment.

–Implementation Guide

–Minimal Data Model (SQL)

```
-- CRIITS Minimal Data Model (DDL)
CREATE TABLE tx_events (
  seq BIGSERIAL PRIMARY KEY,
  tid UUID NOT NULL,
  from_state TEXT,
  to_state TEXT NOT NULL,
  event_type TEXT NOT NULL,
  idempotency_key TEXT NOT NULL,
  producer TEXT NOT NULL,
  observed_at TIMESTAMPTZ NOT NULL DEFAULT now(),
  evidence JSONB,
  causation_id UUID,
  correlation_id UUID
);
CREATE INDEX IF NOT EXISTS tx_events_tid_idx ON tx_events (tid);
CREATE UNIQUE INDEX IF NOT EXISTS tx_events_idem_uk ON tx_events (
   idempotency_key);

CREATE TABLE tx_head (
  tid UUID PRIMARY KEY,
  state TEXT NOT NULL,
  last_event_seq BIGINT NOT NULL,
  last_changed_at TIMESTAMPTZ NOT NULL,
  reason_code TEXT,
  metadata JSONB
);
```

–REST Interface (Sketch)

```
POST /transactions            -> returns tid (CR recorded)
POST /transactions/{tid}/events  -> advances state (idempotency_key
   required)
GET  /transactions/{tid}      -> current state + timeline
GET  /transactions?state!=TS.*&stalled_gt=5m  -> for poller
```

–Polling & Reconciliation

Adaptive backoff: immediately after I, poll at 10s cadence; after 5 minutes, widen to 30s; after 30 minutes, 2m; cap at 10m. Reset cadence upon any new event. Reconciliation loop periodically ingests authoritative lists from vendors (e.g., settlement files, KYC batches) and writes missing I/TS events retroactively with observed timestamps from evidence.

–Idempotency & Concurrency

Producer rule: reuse the same idempotency key for a logical effect. Consumer rule: de-dupe on idempotency key. Use optimistic concurrency on projection head.

–Timeouts, Retries, Escalation

Per-state TTL; retry budgets; on exhaustion, TS.EXPIRED with reason code. Automatic ticketing for SLO breaches.

–Observability & SLOs

–Core Metrics

Time-to-Finality (CR→TS), Stalled Rate (I-state dwell beyond TTL), Missed-by-Push vs Rescued-by-Poll, Duplicate Event Rate, Transition Error Rate.

–Dashboards

Funnel CR→I→I*→TS by cohort; dwell heatmaps; top failure codes.

–Security, Compliance, Audit

Data minimization and encryption; append-only log with checksums; PII/PCI segregation; retention and archiving.

–Case Studies (Domain-Agnostic)

–Payment Authorization & Capture

CR: order created; I: auth initiated; I: 3-DS; I: AUTHORIZED; I: CAPTURE_PENDING; TS: SUCCEEDED/FAILED/EXPIRED.

–KYC Verification

CR: KYC submitted; I: vendor hit; I: MATCHED/MISMATCH/MANUAL_REVIEW; TS: APPROVED/REJECTED/EXPIRED.

–Logistics Fulfilment

CR: shipment created; I: pickup; I: in-transit; TS: DELIVERED/RTO/LOST.

–Adoption Playbook & Maturity

Level 0: map existing statuses; Level 1: emit events; Level 2: outbox + poller; Level 3: SLOs & automated reconciliation; Level 4: vendor contracts reference CRIITS states.

–Limitations & Future Work

CRIITS standardizes observation, not business workflows; some ecosystems lack reliable reconciliation APIs. Future: reference DSL, open schemas, conformance tests.

–Conclusion

Confining lifecycles to CR → I → I* → TS with strict invariants, idempotency, and push+poll recovery yields uniform observability and measurable finality with minimal disruption.

–Appendix A: State & Reason Codes

- CR
- I.* (namespaced): I.AUTH_REQUIRED, I.AUTHORIZED, I.CAPTURE_PENDING
- TS.SUCCEEDED | TS.FAILED | TS.CANCELED | TS.EXPIRED

Reason codes (examples): PG_DECLINED, TIMEOUT, INVALID_INPUT, RETRY_BUDGET_EXCEEDED, DOWNSTREAM_5XX, HUMAN_REJECTED.

–Appendix B: Sample Event (JSON)

```
{
  "tid": "8c2e8b3c-2e3d-4c7d-9c1a-8f07c5f2c901",
  "from_state": "I.AUTHORIZED",
  "to_state": "TS.SUCCEEDED",
  "event_type": "success_confirmed",
  "idempotency_key": "auth-8c2e8b3c-...-try-1",
  "producer": "capture-worker-v3",
  "observed_at": "2025-09-12T07:10:12Z",
  "evidence": {
    "pg_ref": "PG12345",
    "amount_minor": 129900,
    "currency": "INR",
    "files": [{"type":"settlement","uri":"s3://.../2025-09-12/settlement.
      csv"}]
  },
  "causation_id": "a2ff7e84-...-42",
  "correlation_id": "order-5b7..."
}
```

–Appendix C: Querying Stalled Transactions (SQL)

```
SELECT tid, state, last_changed_at
FROM tx_head
WHERE state LIKE 'I.%'
  AND last_changed_at < now() - interval '15 minutes';
```

–Appendix D: Operator Playbook (RMG / Poker Example)

Scope

Applies CRIITS to deposits, withdrawals, KYC, table buy-ins, and game-result settlements in a real-money gaming (RMG) poker platform.

KPIs

TTF (CR→TS) per flow; Stalled Rate in I.AUTH_REQUIRED, I.KYC_REVIEW, I.CAPTURE_PENDING; Duplicate Events per producer; Missed-by-Push vs Rescued-by-Poll.

Alerting

- High: Stalled Rate ¿ 2% for ¿ 10 min in any I-state.
- High: TTF P95 breach vs SLA (e.g., deposits ¿ 2 min, withdrawals ¿ 2 hours).
- Medium: Duplicate Event Rate ¿ 0.1% over 5 min window.
- Medium: Vendor webhook silence ¿ 15 min; switch to aggressive polling policy.

Runbooks

Deposits: If in I.CAPTURE_PENDING beyond 15 min, query settlement file; on match, emit TS.SUCCEEDED with evidence; else, escalate and auto-create ticket.
Withdrawals: If I.MANUAL_REVIEW beyond 24h, batch notify Risk; auto-expire to TS.EXPIRED with reason if KYC outdated.
KYC: If vendor mismatch in evidence, route to HUMAN_REVIEW and enforce TS on resolution; ensure new transactions reference parent_tid for reversals.

Dashboards

Funnel per flow; Dwell heatmap by state; Failure codes by vendor; Real-time Missed-by-Push vs Rescued-by-Poll.

–Appendix E: References

See `references.bib` for canonical patterns: Sagas, Outbox, Event Sourcing.