

Project: High-Throughput Fan-Out Engine (Java 21)

Overview

- Streaming ingestion from CSV / JSONL / Fixed-width.
- Fan-out to REST, gRPC, MQ, Wide-DB mock sinks.
- Per-sink transformations, rate limiting, retries, DLQ.
- Backpressure via bounded queues and virtual-thread workers.

Architecture (summary)

- InputReader -> Source Queue -> Dispatcher -> Per-sink Queue -> Sink Workers
- Transformer per sink type: JSON, Protobuf (manual wire encoding), XML, Avro/CQL map.

Backpressure & Concurrency

- Bounded queues for source and sinks prevent OOM.
- Virtual threads per sink worker scale with CPU cores.
- Dispatcher blocks when any sink queue is full, slowing ingestion safely.

Resilience

- Each record retried up to 3 times per sink.
- Permanent failures are appended to dlq/<sink>.jsonl.

Observability

- Every 5 seconds prints: ingested, dispatched, throughput, per-sink success/failure/retries.

Assumptions

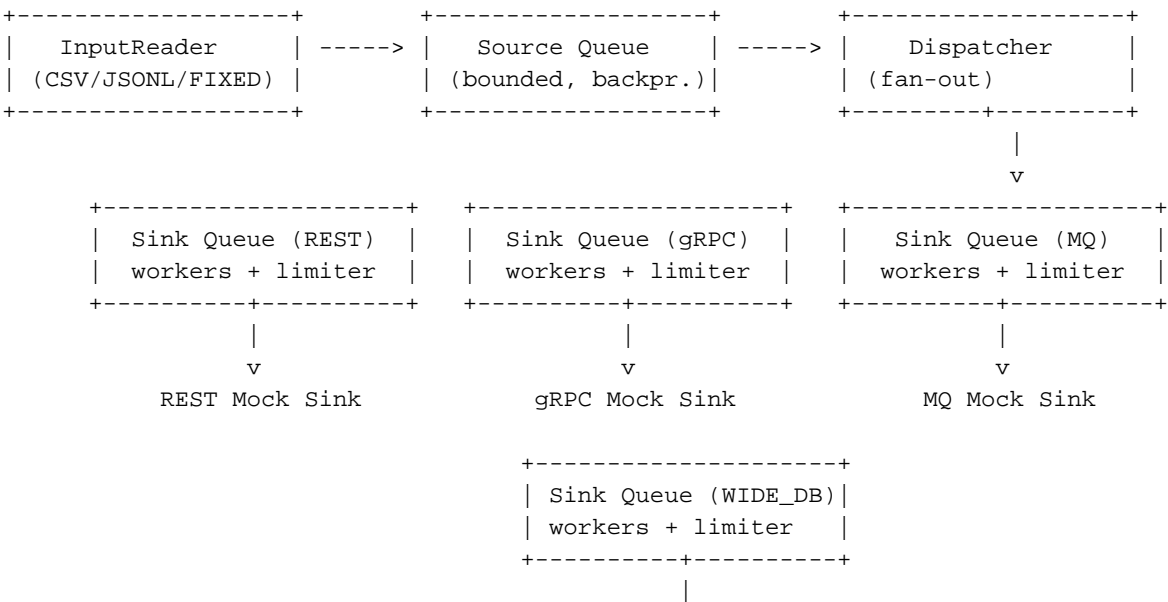
- Records are small per line and fit in memory.
- Timestamps are numeric when provided.
- Protobuf payload uses standard wire encoding via CodedOutputStream.

Prompts Used

- â Generate a Java 21 fan-out engine design using Strategy + Factory.â
- â Implement streaming ingestion for CSV/JSONL with backpressure and bounded queues.â
- â Provide a minimal Maven project with tests and README.â

Architecture Diagram

High-Throughput Fan-Out Engine - Architecture



v

Wide DB Mock Sink

Each sink applies a Strategy Transformer (JSON / Protobuf / XML / Avro+CQL map).
Retries (max 3) + DLQ write on failure. Metrics every 5 seconds.