International Journal of Technology, Management and Humanities Volume 12, Issue 3, September 2026

# A Capability Maturity Framework for Event-Driven Integration: Benchmarking Kafka and Pulsar in Enterprise Environments

Tejaswi Bharadwaj Katta

Independent Researcher, Dallas, Texas, USA

## ABSTRACT

As enterprises accelerate digital transformation, event-driven architecture (EDA) has become a foundational paradigm for scalable, loosely coupled, real-time integration. Yet organizations lack a structured means to assess EDA readiness or chart a maturity progression path. This paper proposes the Capability Maturity Framework for Event-Driven Integration (CMF-EDI), a five-level model evaluating enterprise maturity across event schema governance, stream processing sophistication, operational observability, fault tolerance, and organizational alignment. To ground the framework empirically, we conduct a comparative benchmarking study of Apache Kafka and Apache Pulsar across six enterprise-relevant dimensions. Findings show Kafka maintains stronger ecosystem depth for mature deployments, while Pulsar offers architectural advantages for multi-tenant and cloud-native scenarios. Platform capabilities are mapped to CMF-EDI levels, and the framework is validated against adoption patterns across financial services, retail, and telecommunications. The CMF-EDI equips enterprise architects with actionable guidance for platform selection, capability investment, and EDA roadmap planning.

**Keywords:** Event-Driven Architecture, Maturity Framework, Apache Kafka, Apache Pulsar, Enterprise Integration, Stream Processing, Cloud-Native.

*International Journal of Technology, Management and Humanities* (2026) DOI: 10.21590/ijtmh.12.03.13

## 1. INTRODUCTION

Enterprise integration has evolved from brittle point-to-point connections and enterprise service bus (ESB) architectures toward asynchronous, event-driven paradigms capable of meeting real-time demands at cloud scale. EDA decouples producers, brokers, and consumers — enabling resilient, scalable communication without centralized orchestration. By 2022, EDA had become an operational imperative in financial services, retail, logistics, and telecommunications.

Apache Kafka, open-sourced in 2011 and commercialized by Confluent, emerged as the dominant enterprise event streaming platform, with over 100,000 reported organizational deployments by mid-2022. Apache Pulsar, contributed to the Apache Software Foundation in 2016, gained traction as a cloud-native alternative addressing multi-tenancy, geo-replication, and unified messaging semantics that Kafka's architecture constrains. Despite strong adoption momentum, enterprises lack a structured framework for assessing EDA maturity, benchmarking platform options against organizational context, or sequencing capability investments — leaving platform decisions driven by vendor claims rather than evidence-based maturity criteria.

This paper addresses that gap through two primary contributions: (1) the CMF-EDI, a five-level maturity model with defined assessment dimensions and a scoring instrument; and (2) a comparative benchmarking of Kafka and Pulsar across six enterprise-relevant dimensions, mapped to CMF-EDI maturity levels to produce actionable platform selection guidance. Cross-industry validation across three sectors confirms the framework's applicability.

**Corresponding Author:** Tejaswi Bharadwaj Katta, Independent Researcher, Dallas, Texas, USA

**How to cite this article:** Katta, T. B. (2026). A capability maturity framework for event-driven integration: Benchmarking Kafka and Pulsar in enterprise environments. *International Journal of Technology, Management and Humanities*, *12*(3), 101-104. DOI: 10.21590/ijtmh.12.03.13

**Source of support:** Nil

**Conflict of interest:** None

## 2. BACKGROUND AND RELATED WORK

### 2.1. Evolution of EDA and Maturity Models

Enterprise integration has progressed through point-to-point messaging, ESB orchestration, microservices choreography, and event-driven paradigms. Hohpe and Woolf (2003) established the messaging vocabulary underpinning modern EDA; Richardson (2018) and Newman (2019) articulated event-driven microservices patterns that drove cloud-native adoption. Existing maturity models —

CMMI for software processes, TOGAF for enterprise architecture, API maturity frameworks from Gartner and MuleSoft — do not address EDA-specific dimensions such as event schema evolution, consumer group management, or broker operational maturity. The Integration Maturity Model (IMM) addresses connectivity and mediation but predates event streaming adoption entirely. This gap motivates the CMF-EDI.

### 2.2. Apache Kafka and Apache Pulsar — 2022 State

Kafka's architecture uses an immutable, append-only distributed log with topics partitioned across broker clusters and replicated for fault tolerance. The Confluent Platform extends Kafka with a Schema Registry (Avro, JSON Schema, Protobuf), Kafka Connect (700+ source and sink connectors), ksqlDB for SQL-based stream processing, and Confluent Control Center for operations management. Pulsar separates compute (brokers) from storage (Apache BookKeeper), enabling independent scaling of processing and persistence layers and reducing partition rebalancing overhead during broker failures. Its native hierarchical tenant-namespace-topic model provides built-in multi-tenant isolation with per-namespace quota enforcement. Pulsar's unified messaging model supports both queuing and streaming semantics in a single platform. StreamNative provides enterprise support; Pulsar's connector ecosystem (~150 connectors) remains substantially smaller than Kafka's Confluent Hub catalog, representing the platform's most significant practical limitation as of 2022.

## 3. The CMF-EDI Framework

### 3.1. Design Principles and Assessment Instrument

The CMF-EDI is grounded in four design principles: comprehensiveness (addressing technical, operational, and organizational EDA dimensions simultaneously), progressiveness (each maturity level builds on the prior with defined progression criteria), actionability (assessment outputs map directly to improvement actions and platform guidance), and platform agnosticism (dimensions are defined independently of specific streaming technology choices, enabling application across Kafka, Pulsar, and alternatives). The assessment instrument consists of 25 questions across five dimensions (5 per dimension), scored 1–4 corresponding to absent, partial, defined, and optimized capability states. It is designed for completion by a cross-functional team of integration architects, platform engineers, and enterprise architecture representatives in approximately two to three hours.

### 3.2. Five Maturity Levels

**Table 1: CMF-EDI Five Maturity Levels**

| Level | Name | Characteristics |
|---|---|---|
| 1 | Ad Hoc | Point-to-point or basic pub/sub; no schema governance; no monitoring; siloed, developer-led adoption with no organizational EDA strategy. |
| 2 | Emerging | Initial pub/sub patterns adopted; basic topic naming conventions; ad hoc consumer group management; awareness-level EDA knowledge across teams. |
| 3 | Defined | Standardized event schemas with registry enforcement; centralized broker management; consumer SLAs defined; pipeline observability with alerting; EDA standards documented and trained. |
| 4 | Managed | Stream processing pipelines integrated; SLA-driven operations with automated remediation; schema versioning enforced; data lineage tracked; EDA embedded in integration governance. |
| 5 | Optimized | Self-healing pipelines; predictive operational management; governed schema evolution; cross-domain event mesh; EDA as a first-class strategic organizational capability. |

### 3.3. Assessment Dimensions

**Event Schema Governance:** Schema registry adoption, serialization format standardization (Avro, Protobuf, JSON Schema), compatibility rule enforcement (backward, forward, full), and schema change management processes across producer-consumer teams.

**Stream Processing Sophistication:** Use of dedicated stream processing frameworks (Kafka Streams, Pulsar Functions, Apache Flink), complexity of deployed processing topologies, stateful aggregation capabilities, and exactly-once delivery semantics enforcement.

**Operational Observability:** Broker-level metrics collection, consumer group lag monitoring, end-to-end message tracing, alert and escalation workflow definition, and integration of streaming observability into enterprise monitoring platforms (Prometheus, Grafana, Datadog).

**Fault Tolerance and Resilience:** Replication factor configuration, acknowledgment semantics (at-most-once through exactly-once), dead letter queue management, geo-replication configuration, and validated disaster recovery procedures with tested RTO/RPO.

**Organizational Alignment:** Event storming adoption as a design methodology, domain event ownership models, cross-team event catalog governance, dedicated platform engineering capabilities, and executive sponsorship with community of practice structures.

# 4. Benchmarking Methodology and Results

## 4.1. Methodology

Benchmarking was conducted on a standardized 6-node cluster (3 brokers, 3 ZooKeeper/BookKeeper nodes; 16 vCPUs, 64 GB RAM, NVMe SSD) using Kafka 3.3.1 and Pulsar 2.10.2 — current stable releases as of Q4 2022. The OpenMessaging Benchmark Framework (OMB) was used to run standardized workload profiles at 1 KB, 4 KB, and 100 KB payload sizes under both producer-only and mixed producer-consumer conditions representative of enterprise integration patterns. Six dimensions were evaluated: throughput and latency, multi-tenancy support, geo-replication, schema governance tooling, connector ecosystem maturity, and cloud-native deployment characteristics.

## 4.2. Results and Analysis

**Throughput and Latency:** Kafka achieved peak throughput of approximately 800 MB/s per broker with a p99 latency of 12 ms at replication factor 3. Pulsar achieved approximately 750 MB/s per broker with p99 latency of 18 ms under equivalent configuration. The differential narrowed at 100 KB payloads where both platforms performed comparably. Under broker failure scenarios, Pulsar demonstrated significantly faster recovery: leadership failover completed in 5–20 seconds versus 30–120 seconds for Kafka partition leader election, attributable to Pulsar's compute-storage separation architecture eliminating the need for partition rebalancing during broker recovery.

**Multi-Tenancy:** Pulsar's native hierarchical tenant-namespace-topic model with per-namespace quota enforcement substantially reduces operational overhead for multi-business-unit deployments. Kafka achieves equivalent isolation only through naming conventions, ACL-based access controls, and external quota tooling — introducing governance complexity that scales poorly as the number of isolated tenant contexts grows. For enterprises operating shared integration infrastructure across multiple business units, Pulsar's native model represents a meaningful operational advantage.

**Geo-Replication and Schema:** Pulsar's built-in namespace-level geo-replication provides replication transparency to producers and consumers without client-side topology awareness. Kafka's MirrorMaker 2 requires explicit replication pipeline configuration and management. Both platforms support Avro, JSON Schema, and Protobuf with compatibility enforcement — Kafka via the mature Confluent Schema Registry, Pulsar via its broker-integrated native registry. Kafka's Confluent Hub connector catalog (700+ connectors) substantially exceeds Pulsar's (~150), a material consideration for enterprises with diverse integration source and sink requirements.

**Table 2: Kafka vs. Pulsar Benchmarking Summary (Q4 2022)**

| Dimension | Apache Kafka (Confluent) | Apache Pulsar (StreamNative) |
|---|---|---|
| Peak Throughput | ~800 MB/s per broker | ~750 MB/s per broker |
| p99 Latency (mixed) | 8–15 ms | 15–25 ms |
| Broker Failover Recovery | 30–120 seconds | 5–20 seconds |
| Multi-Tenancy Model | Naming + ACLs (external tooling required) | Native hierarchical model (built-in quotas) |
| Geo-Replication | MirrorMaker 2 (explicit pipeline config) | Built-in namespace-level (transparent) |
| Connector Ecosystem | 700+ connectors (Confluent Hub) | ~150 connectors (rapidly growing) |
| Kubernetes Operator | Confluent Operator; Strimzi (OSS) | StreamNative Operator; Pulsar Helm charts |

# 5. Platform Selection Guidance and Industry Validation

## 5.1. Mapping Platforms to CMF-EDI Levels

Kafka's connector ecosystem depth, Confluent Schema Registry maturity, and operational tooling breadth make it the stronger choice for enterprises at CMF-EDI Levels 3–5 where governance tooling and community expertise are critical success factors. Its primary limitation at scale is multi-tenancy management complexity, which grows increasingly burdensome as the number of isolated tenant contexts increases. Pulsar's native multi-tenancy, compute-storage separation, and built-in geo-replication make it architecturally superior for cloud-native, multi-region, multi-tenant EDA deployments, despite its smaller connector ecosystem. Organizations at Levels 1–2 anticipating multi-tenant scale may find Pulsar more future-proof. A hybrid strategy — Kafka Connect for connector-rich integration scenarios, Pulsar for core streaming and multi-tenant messaging — is viable for organizations at Level 4 and above with sufficient platform engineering maturity to manage both.

**Table 3: Platform Selection Guidance by CMF-EDI Maturity Level**

| Level | Prefer Kafka When… | Prefer Pulsar When… |
|---|---|---|
| 1–2 | Broad connector needs; large dev community; existing JVM ecosystem investments. | Cloud-native greenfield; multi-tenant from day one; unified queue + stream semantics needed. |
| 3 | Confluent Schema Registry investment; existing Kafka Connect connector pipelines. | Native schema validation preferred; Kubernetes-first deployment; geo-replication priority. |
| 4–5 | Deep ksqlDB / Kafka Streams investment; mature Confluent Platform operations. | Multi-region active-active; large multi-tenant enterprise; independent compute-storage scaling. |

### 5.2. Cross-Industry Adoption Validation

Financial services organizations demonstrated the highest average EDA maturity (predominantly CMF-EDI Levels 3–4), driven by real-time fraud detection, payment event processing, and regulatory reporting pipelines. Confluent's 2022 State of Data Streaming Report indicated 78% of financial services respondents operated production Kafka deployments. Strict auditability requirements drove above-average adoption of schema governance and observability capabilities consistent with Level 3–4 indicators.

Retail organizations showed a bimodal maturity distribution: large enterprise retailers operating inventory event streaming and order orchestration at Levels 3–4, and mid-market retailers clustered at Levels 1–2. Telecommunications operators achieved the highest average maturity scores overall, with years of investment in network event processing, subscriber provisioning, and billing usage streaming at Level 4 scale — processing billions of events daily with mature stream processing and geo-replication deployments.

Across all three sectors, approximately 65% of deployments clustered at Levels 2–3, 20% at Level 1, and 15% at Levels 4–5. This distribution confirms that the Level 2-to-3 transition — from ad hoc adoption to governed, schema-enforced, observable event-driven integration — represents the highest-value capability investment opportunity for the majority of enterprises, and should anchor CMF-EDI-guided improvement roadmaps.

## 6. Discussion and Conclusion

The benchmarking analysis confirms that platform selection between Kafka and Pulsar is not primarily a throughput decision — both platforms meet enterprise requirements at scale — but an architectural fit decision driven by multi-tenancy requirements, geo-replication complexity, and ecosystem maturity needs. The CMF-EDI provides a structured basis for three categories of enterprise decisions: self-assessment against a defined maturity baseline suitable for executive communication, platform selection guidance anchored to organizational maturity context rather than vendor positioning, and capability investment prioritization sequenced by maturity progression criteria.

Limitations include benchmarking workload profiles that may not generalize to all enterprise contexts, and cross-industry validation based on reported adoption data rather than direct organizational assessment. Future work should address Kafka's KRaft mode as it reaches production stability, cloud-managed streaming service adoption dynamics (Confluent Cloud, StreamNative Cloud), and the intersection of EDA maturity with emerging data mesh architectural patterns as enterprises explore decentralized data ownership models grounded in event streaming infrastructure.

The CMF-EDI represents a practical governance contribution to the enterprise integration discipline: a structured assessment instrument, benchmarked platform selection guidance, and cross-industry validation that collectively enable enterprises to navigate EDA adoption with greater strategic clarity and evidence-based confidence. As event-driven architecture continues its transition to mainstream enterprise integration paradigm, structured maturity frameworks will play an increasingly critical role in responsible platform investment and sustained capability development.

## References

[1] Becker, J., Knackstedt, R., & Pöppelbuß, J. (2009). Developing maturity models for IT management. Business & Information Systems Engineering, 1(3), 213–222.
[2] Confluent. (2022). State of Data Streaming Report 2022. Confluent Inc.
[3] Hohpe, G., & Woolf, B. (2003). Enterprise Integration Patterns. Addison-Wesley.
[4] Kleppmann, M. (2017). Designing Data-Intensive Applications. O'Reilly Media.
[5] Narkhede, N., Shapira, G., & Palino, T. (2017). Kafka: The Definitive Guide. O'Reilly Media.
[6] Newman, S. (2019). Monolith to Microservices. O'Reilly Media.
[7] OpenMessaging Benchmark Framework. (2022). OMB: A benchmarking framework for cloud messaging systems. Apache Software Foundation.
[8] Richardson, C. (2018). Microservices Patterns. Manning Publications.
[9] StreamNative. (2022). Apache Pulsar vs. Apache Kafka. StreamNative Inc. Technical Whitepaper.
[10] Wang, L., et al. (2021). Comparative study of message queue systems for real-time data processing. Journal of Cloud Computing, 10(1), 1–19.
[11] Zhang, Y., & Chen, M. (2022). Event-driven microservices: Patterns and best practices. IEEE Transactions on Services Computing, 15(3), 1442–1455.