ISSN 2278-3091
Volume 14, No.6, November – December 2025
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse031462025.pdf
https://doi.org/10.30534/ijatcse/2025/031462025

Event-Driven Architecture (EDA) in IT Systems

Roilian Mykyta
Engineering manager, LeanDNA, USA, n.roylyan@outlook.com

Received Date: October 18, 2025 Accepted Date: November 23, 2025 Published Date: December 06, 2025

ABSTRACT

In this article, the application of event-driven architecture in information systems focused on supply management and production processes is investigated. The theoretical foundation of the event-driven approach is examined, including the key mechanisms of message delivery and storage, as well as the differences between message queue systems and log-based event streams. The practical applicability of event-driven architecture in logistics and supply chain management is also considered. Particular attention is paid to high-load industries, where the synchronicity of supplies and the stability of processes critically depend on the predictability of latencies and the reliability of event-driven workflows.

Key words: Event-driven architecture, information systems, supply chains, inventory management, event stream processing, Apache Kafka, Apache Pulsar.

1. INTRODUCTION

Contemporary information systems work within the conditions of rising business process dynamism and data volume growth rates. Classic architecture models based on cyclical querying and centralized processing are increasingly loosing their efficiency due to the rising delay between the detection of an event and corresponding managerial intervention and consequently slower system reaction and lower resilience. Here, architecture models that ensure instantaneous detection of changes and autonomous distribution of information to pertinent subsystems are assuming special significance.

In recent years, the concept of Event-Driven Architecture (EDA) has become widespread in both commercial applications and industrial information systems. Its adoption is driven not only by advances in the performance of event-streaming platforms but also by a paradigm shift in the design of digital supply chains and enterprise services. Academic and applied research increasingly views EDA as a foundational instrument for moving from reactive models of management to proactive and predictive ones, in which systems do not merely register deviations but ensure their immediate integration into decision-making cycles.

The research is intended to analyze the potential and limitations of EDA in the context of information systems that support logistics and production management. The methodology combines a review of existing event-streaming platforms, an examination of their architectural principles, and comparative performance evaluations presented in recent studies.

2. MAIN PART. THEORETICAL FOUNDATIONS OF EDA

The current state of EDA represents a model for designing software systems in which the central element is the concept of an event. In this context, an event is defined as a recorded fact that reflects a change in the state of an object or process within an information system. The capture and dissemination of events make it possible to decouple the source of change from the consumers of information.

In contrast to traditional request–response architectures, EDA is based on asynchronous interaction between components. Event producers publish information without specifying a particular recipient, while interested consumers subscribe to the relevant categories of events independently. In practice, the choice of an EDA depends on the nature of the tasks and the system's requirements (table 1).

Table 1: Recommendations for considering EDA [1, 2]

Characteristic/requirement	Recommendation
Real-time responsiveness is critical	EDA
High scalability and throughput are needed	EDA
Loose coupling and flexibility are desired	EDA or Microservices
Complex event processing and pattern detection are required	EDA
Asynchronous communication is preferred	EDA
Strict data consistency is mandatory	Traditional monolithic architectures or carefully designed microservices with transaction management

Simple, CRUD-based applications	Traditional monolithic architectures or microservices
Legacy system integration	EDA (with adapters or wrappers) or SOA
High performance computing and tightly coupled components	Monolithic architectures
Small, simple applications with limited scalability needs	Monolithic architectures

Mechanisms of event delivery and storage play a particularly important role in event-driven approach. Two technological solutions are most commonly employed: message queue systems and log-based event streams. Message queues are designed to transfer messages from a producer to one or more consumers, ensuring timely routing. Log-based event streams, by contrast, preserve the sequence of events in the form of an append-only log, which makes it possible to replay event flows and maintain a long-term history of changes.

This principle is implemented in a number of modern event-streaming platforms. Apache Kafka, for instance, is widely adopted for maintaining durable event logs and enabling large-scale stream processing across diverse applications [3]. One prominent example is Apache Pulsar, whose architecture is based on the interaction of brokers that handle event ingestion and distribution, the BookKeeper system for long-term storage, and the ZooKeeper service for coordination. The conceptual structure of a Pulsar cluster illustrates the key characteristics of a log-based event stream and demonstrates the interconnection of its core components (figure 1).

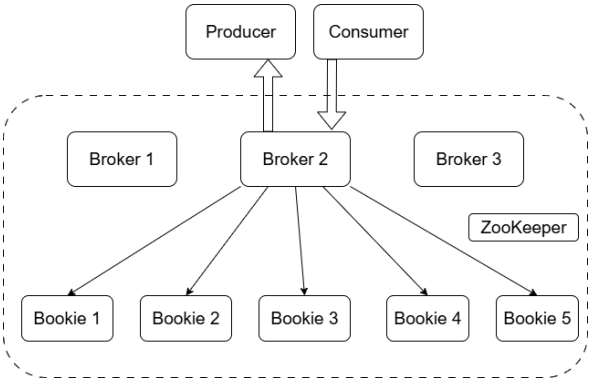

Figure 1: Apache Pulsar architecture [4]

A central issue in the design of EDA is the guarantee of message delivery. Three delivery models are typically distinguished: at-most-once, at-least-once, and exactly-once. Each of these approaches entails specific advantages and limitations. The first minimizes latency but allows for the potential loss of events. The second ensures reliable preservation of events, yet requires additional mechanisms to handle duplicates. The third provides the strongest delivery guarantees but incurs additional overhead associated with transactional support.

Stream processing in EDA relies on the temporal characteristics of events. A distinction is made between the time an event occurs, the time it enters the system, and the time it is actually processed. The architecture of event-driven systems is typically based on the principle of eventual consistency. This means that different components of the system update their states asynchronously and not always simultaneously. To tackle this problem, architectural patterns such as Command Query Responsibility Segregation (CQRS) and event sourcing can be employed. The latter derives the current state from the accumulated sequence of events.

In literature, empirical evaluations of advanced EDA patterns, particularly event sourcing and CQRS, demonstrate tangible improvements in performance and reliability across a variety of operational metrics. These improvements are reflected in reductions of processing latency, mitigation of integration-related incidents, and enhanced consistency in inventory management (table 2).

Table 2: Performance improvements with advanced EDA patterns [5]

Challenge/solution	Before implementation	After implementation
Idempotent processing – inventory discrepancies	4,7% increase	0,35% occurrence
Channel inventory convergence (95.8% of transactions)	Variable	2,3 seconds
Concurrent modifications during peak	23-47 per minute	99,3% automatic resolution
Processing latency with partitioning	842 ms	67 ms
Database load with caching	Baseline	24%
Integration-related incidents	Baseline	17%

Taken together, these principles establish EDA as a theoretical foundation for building scalable and flexible information systems. The architecture is grounded in asynchronous interaction, loose coupling of components, the use of stream processing, and the assurance of data consistency through event-driven exchange mechanisms.

2.1 Application of EDA in Logistics and Supply Chain Management

The event-driven approach in logistics conceptualizes the supply chain as a continuous stream of signals reflecting the status of orders, inventories, shipments, and production operations. Unlike the periodic data extractions employed in traditional planning systems, here informational changes are recorded as soon as they occur and immediately trigger managerial responses.

Supply monitoring in an event-driven model is achieved through the alignment of planned and actual milestones. The creation of an order establishes the initial trajectory; supplier confirmation specifies delivery dates and quantities; shipping notices, carrier telemetry, and checkpoint passage records form the basis of arrival forecasts; and the goods receipt event closes the cycle and initiates warehouse placement operations. Any deviation in timing or quantity becomes a trigger for managerial action: automatic notification of responsible parties, reprioritization of orders or reassignment of materials. This model of supply management is implemented through an event-driven loop that links sources of change with services responsible for orders, inventory, and payments, as represented in the conceptual structure of such interactions (figure 2).

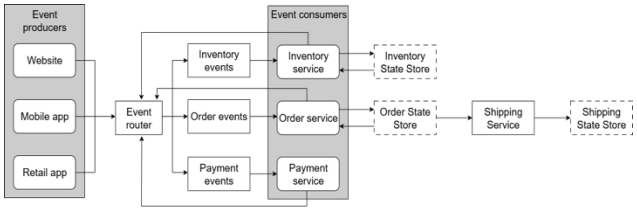

Figure 2: EDA for supply chain management [6]

Real-time inventory management is built on the aggregation of events related to inflows and outflows and the maintenance of a material «availability ledger». Within such a ledger, distinctions are made between physical stock, reserved quantities, expected receipts, returns, and quality holds; each change is recorded as an event and immediately reflected in available-to-promise calculations.

Intra-organizational logistics and the integration of warehousing with production benefit significantly from events that capture changes in the status of materials within the enterprise. Withdrawals from storage, order picking, movements between warehouse locations, issuance to the shop floor, returns, and quality holds are all recorded as discrete facts and aligned with the production takt [7]. Deviations from the planned takt identified at early stages trigger resource reallocation. Notably, the impact of event-driven management is assessed through indicators directly linked to supply chain risk.

Integration of the event layer with planning processes enables the reconciliation of short-term responsiveness with medium-term optimization. Signals related to schedule shifts, demand fluctuations, or capacity availability immediately inform operational decisions, while aggregated facts, organized into time windows, feed into sales and operations planning processes.

2.2 Practical significance of EDA for High-Load Industries

The event-driven approach has been widely adopted across diverse domains, with major companies such as Netflix, Spotify, and Uber employing it to support their operational and customer-facing systems [8]. Yet high-load industries demand more. Beyond handling large data volumes, they require predictable response times, resilience, reproducibility, and auditability. In these settings, EDA serves as a technological backbone that integrates continuous operational flows with tightly controlled production cycles.

The distinct requirements of high-load industries are evident already at the stage of goal formulation. Managerial decisions must be taken within a predefined «latency budget», which is determined by the production takt time, the duration of supply chain segments, and the economic cost of downtime. In the aerospace sector, this implies that the informational cycle, from detecting a deviation to initiating a corrective action, must be completed within fractions of the duration of a critical operation.

Latency characteristics in real-world systems are determined less by average values and more by the behavior observed in the «long tails» of the distribution. For high-load industries, it is the 99th percentile of delays that most accurately defines the predictability and controllability of production processes, as illustrated in figure 3.

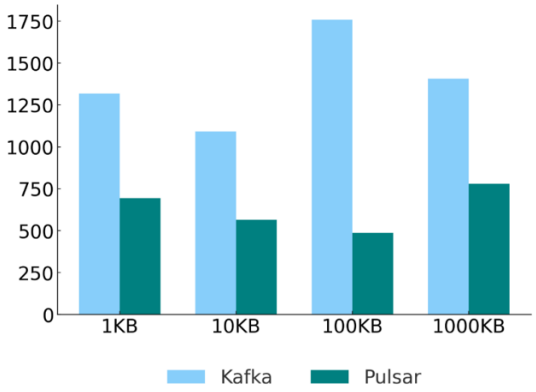

Figure 3: Benchmarks of the 99th percentile end-to-end latency, ms [9]

Reliability in such contexts is understood more broadly than the mere fault tolerance of individual software components. The system must remain operational under partial degradation, guarantee determinism of results during reprocessing, and provide operators with an observable «lifeline» for every critical event. To achieve this, redundant routing paths can be employed. In production systems, traceability of materials and operations plays a particularly critical role. In the aerospace industry, it is necessary to confirm the full «genealogy» of components and assemblies, linking each unit to quality documentation, measurement data, material batches, and personnel actions.

Synchronization of supply flows and production takt under high-load conditions is not feasible without prioritization of event streams. At the informational level, this involves separating events into service classes. Delivery streams critical to takt are processed within a «hot» path with minimal latency, while background analytics and low-priority notifications are routed through «warm» and «cold» layers.

Capacity planning and scalability in high-load environments rely on explicit calculations. The starting point is the peak intensity of events across key flows. Superimposed on this baseline are the chosen target latency thresholds and the permissible depth of queues under peak load. Capacity planning requires not only theoretical models of event flows and latency budgets, but also empirical evidence of how different architectures behave under increasing load. Comparative benchmarks provide insight into scalability limits, demonstrating how EDA maintain higher throughput as the number of concurrent users grows (figure 4).

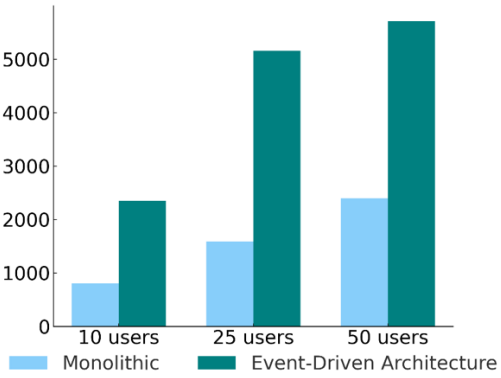

Figure 4: Throughput comparison of monolithic and EDA under varying user loads, requests per minute [10]

Failure scenarios in real-world production are inevitable and therefore must be anticipated and rehearsed in advance. The practical value of the event-driven approach in high-load industries lies in its ability to localize deviations before they escalate into production disruptions [11]. In aerospace manufacturing, this is achieved through early signals from supply chains, inbound quality control, and production measurements, all of which are synchronized to a common timeline and compared with the takt schedule.

In this way, EDA provides a robust and controllable framework for the operation of high-load production systems. It enables the alignment of requirements for responsiveness, resilience, and traceability with the practical constraints of equipment and logistics. In practice, success depends less on the choice of a particular platform than on the discipline of design.

3. CONCLUSION

The EDA forms the basis of constructing information system structures that are highly scalable and can function within highly dynamic and uncertain conditions. Through its use, it is possible to integrate processing of data streams, reaction of the system with automated deviation responses and end-to-end traceability of processes that is especially essential within supply chain and production control. In high-load industries, the event-driven paradigm contributes to reducing latency, improving system resilience and the predictability of decision-making cycles, while also facilitating the integration of heterogeneous services into a unified, manageable environment. The combination of EDA's principles with current event-streaming platforms creates the basis for the creation of digital ecosystems oriented toward real-time decision-making and greater reliability in production and logistics processes.

REFERENCES

1. R. Manchana. **Event-Driven Architecture: Building Responsive and Scalable Systems for Modern Industries,** *International Journal of Science and Research (IJSR)*, Vol. 10(1), pp. 1706-16, 2021.
2. Y. Drogunova. **The impact of software quality assurance practices on the competitiveness of the technology sector,** *International Journal of Advanced Research in Science, Communication and Technology*, Vol. 5(1), pp. 735-740, 2025.
3. S. Bolgov. **Using Kafka in financial data processing systems: benefits and challenges,** *Universum: technical sciences: electron. scientific journal*, № 2(131), pp. 13-16, 2025.
4. Architecture Overview / Apache Pulsar // URL: https://pulsar.apache.org/docs/next/concepts-architecture-overview/ (date of application: 19.08.2025).
5. N. Jose. **Event-Driven Architecture in Retail: Real-Time Inventory Synchronization for Omnichannel Retail,** *International Journal of Computing and Engineering*, Vol. 7(16), pp. 13-23, 2025.
6. Event-driven architectures / Google Cloud // URL: https://cloud.google.com/eventarc/docs/event-driven-architectures (date of application: 22.08.2025).
7. A. Kovalenko. **Architectural and algorithmic methods for enhancing the resilience of high-load backend services in the financial sector,** *Norwegian Journal of development of the International Science*, № 158, pp. 87-91, 2025.
8. Z. Qin, Q. Shuai, G. Wang, P. Zhang, M. Cao, M. Chen. **Architecture and System of E-Commerce,** *InE-Commerce: Concepts, Principles, and Application*, P. 221-326, 2022.
9. M. S. H. Chy, M. A. R. Arju, S. M. Tella, T. Cerny. **Comparative Evaluation of Java Virtual Machine-Based Message Queue Services: A Study on Kafka, Artemis, Pulsar, and RocketMQ,** *Electronics*, Vol. 12(23), № 4792, 2023.
10. H. Cabane, K. Farias. **On the impact of event-driven architecture on performance: An exploratory study,** *Future Generation Computer Systems*, Vol. 153, pp. 52-69, 2024.
11. G. N. Seryan, A. V. Adamyan. **Transformation of corporate financial strategies in the era of digital ecosystems,** *Professional Bulletin: Economics and Management*, № 1/2025, pp. 10-16, 2025.