Journal of Computer Science and Technology Studies
ISSN: 2709-104X
DOI: 10.32996/jcsts
Journal Homepage: www.al-kindipublisher.com/index.php/jcsts

JCSTS
AL-KINDI CENTER FOR RESEARCH AND DEVELOPMENT

| RESEARCH ARTICLE

Engineering at Scale: Fanout, De-duplication, and Latency in Real-Time Event Platforms

Vinaya Nadig
Independent Researcher, USA
Corresponding Author: Vinaya Nadig, **E-mail**: nadigvinaya@gmail.com

| ABSTRACT

Modern real-time event platforms serve as critical infrastructure for digital ecosystems, delivering time-sensitive information across diverse applications from financial services to conversational interfaces. These platforms confront escalating demands while navigating the inherent tension between speed, reliability, and accuracy. This article explores architectural foundations enabling massive-scale message distribution, including publish-subscribe models, partitioned event buses, and geographic distribution strategies. It delves into de-duplication techniques essential for preventing message duplication across complex topologies, from deterministic fingerprinting to machine learning-based approaches. The discussion extends to multi-surface orchestration mechanisms that intelligently route notifications across heterogeneous device ecosystems based on contextual factors and recipient state. Performance optimization methodologies, including latency budgeting, predictive resource allocation, and graceful degradation patterns, complete this examination of scalable event platform design. Throughout, particular attention focuses on how artificial intelligence transforms traditional event processing into dynamic, context-aware systems capable of autonomous operation at a global scale.

| KEYWORDS

Event-driven architecture, Real-time notification, De-duplication, Multi-surface orchestration, Latency optimization

| ARTICLE INFORMATION

ACCEPTED: 01 August 2025 **PUBLISHED:** 8 September 2025 **DOI:** 10.32996/jcsts.2025.7.9.48

I. Introduction

Real-time event platforms have evolved into essential components of modern digital infrastructure, powering everything from financial systems to conversational AI interfaces. These platforms leverage event-driven architectures that fundamentally transform how distributed applications communicate, particularly in microservices environments where decoupled components must interact efficiently while maintaining system resilience. Research demonstrates that event-driven approaches deliver significant performance advantages over traditional request-response patterns, especially as system complexity increases and more services need to communicate [1].

The scale challenges facing these platforms are formidable. Contemporary event systems must process millions of messages per minute while maintaining consistent delivery guarantees across geographically distributed infrastructures. The publish-subscribe model central to these architectures enables dynamic reconfiguration of message flows without disrupting core processing pipelines, a critical capability for global deployments. Event-driven systems have proven particularly adept at handling traffic spikes through asynchronous processing, which provides natural load balancing and graceful degradation mechanisms that preserve core functionality even under extreme conditions [1].

Artificial intelligence now enhances numerous aspects of event platform operation. Machine learning algorithms optimize routing decisions based on real-time system conditions, detect anomalies in message patterns, and enable predictive resource scaling. These AI capabilities represent a fundamental shift from static, rule-based event processing to dynamic, context-aware message handling. Predictive analytics applied to historical event data facilitates proactive capacity planning, while intelligent automation reduces operational overhead through autonomous decision-making that optimizes both performance and resource utilization [2].

Modern event platform design confronts an inherent engineering trilemma: balancing speed, reliability, and accuracy. Each dimension presents unique challenges - minimizing delivery latency, ensuring reliable message transmission, and maintaining accuracy through proper de-duplication. Research indicates that optimizing for low latency often increases duplicate delivery rates, while stringent de-duplication mechanisms typically introduce processing overhead affecting system responsiveness. Finding the appropriate balance requires sophisticated monitoring and continuous refinement of platform architecture [2].

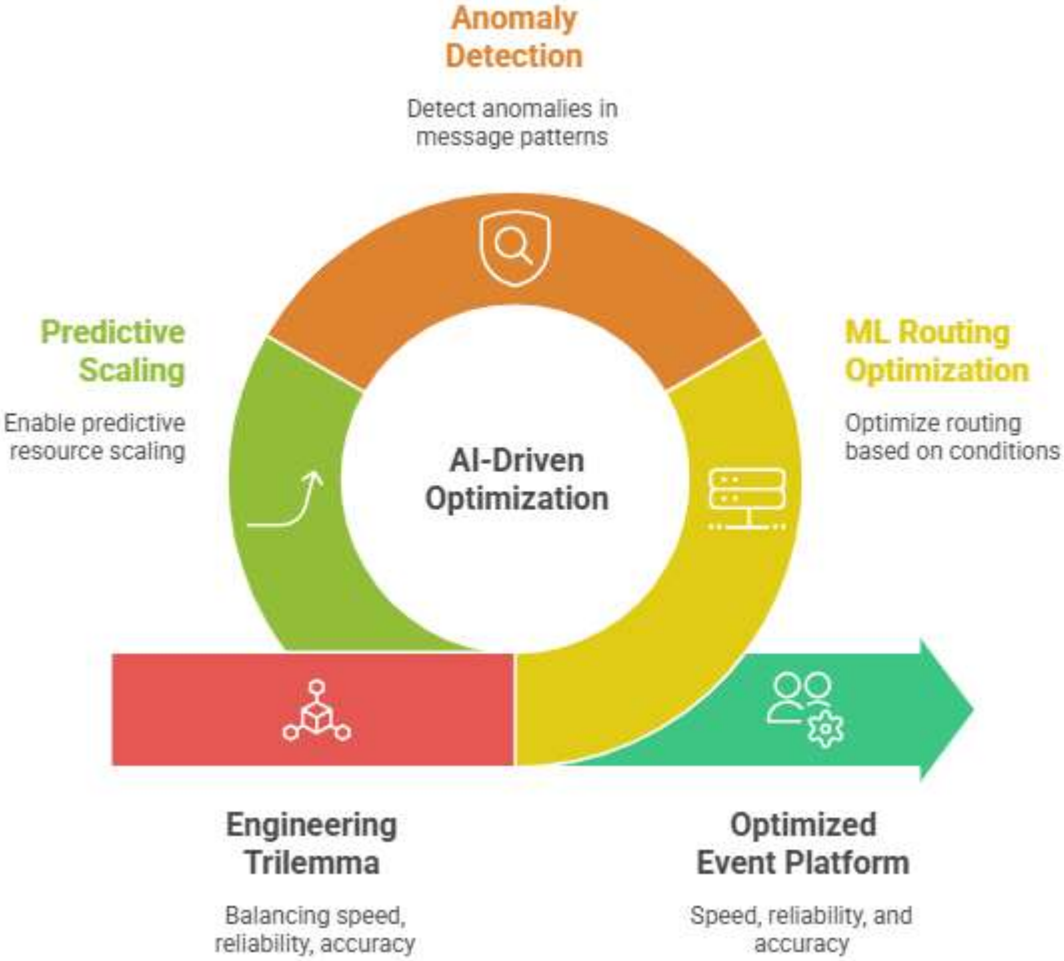
Anomaly Detection
Detect anomalies in message patterns
Predictive Scaling
Enable predictive resource scaling
ML Routing Optimization
Optimize routing based on conditions
AI-Driven Optimization
Engineering Trilemma
Balancing speed, reliability, accuracy
Optimized Event Platform
Speed, reliability, and accuracy

Fig 1: Illustration showing the interaction between key components: (A) Real-time event processing pipeline, (B) ML-based optimization layer, (C) Performance monitoring feedback loop, and (D) Automated resource allocation system. Arrows indicate data flow and control signals between components, demonstrating how AI enhances traditional event processing through dynamic optimization [1, 2]

This article examines architectural approaches that address these competing demands in AI-enhanced event platforms. The discussion covers foundational distribution patterns, de-duplication techniques, multi-surface orchestration strategies, and performance optimization methodologies. Throughout, the focus remains on practical design patterns drawn from real-world implementations serving diverse application domains at a global scale.

II. Architectural Foundations for Scalable Event Distribution

The publish-subscribe paradigm has evolved significantly through distributed event aggregation techniques that enhance scalability in modern event platforms. These advanced strategies reduce network traffic by combining multiple atomic events into semantically meaningful composite events, creating efficiency gains particularly apparent in high-volume scenarios. Distributed aggregation mechanisms strategically positioned within network topologies minimize unnecessary message propagation while preserving the essential decoupling between publishers and subscribers. The programmatic definition of aggregation patterns enables systems to adapt to changing workload characteristics without requiring architectural redesign, a critical capability for dynamic operating environments [3].

Partitioned event buses provide foundational support for conversational and proactive intelligence systems at scale. By segmenting event streams into logical partitions that can be processed independently, these architectures enable horizontal scaling across computational resources. Effective partitioning strategies balance processing locality against distribution uniformity to prevent performance-degrading hotspots. Session-based partitioning schemes prove particularly valuable for conversational interfaces, keeping related interactions within the same processing boundaries. Dynamic partition rebalancing further enhances these architectures by adapting to changing load patterns in real-time [3].

The architectural choice between stateful and stateless message brokering presents fundamental trade-offs based on application requirements. Stateful brokers maintain explicit knowledge of consumer position and delivery guarantees, enabling sophisticated exactly-once semantics at the cost of increased complexity. Stateless approaches prioritize throughput and elastic scaling, simplifying operations while potentially relaxing delivery guarantees. Hybrid architectures that externalize state management to specialized subsystems while maintaining stateless processing paths have proven effective for applications with mixed reliability requirements [3].

Component	Core Idea	Key Benefit
Distributed Event Aggregation	Merge atomic events into composite events via strategic network positioning.	Reduces traffic, adapts to workload changes.
Partitioned Event Buses	Split streams into partitions with session-based grouping and dynamic rebalancing.	Enables horizontal scaling, prevents hotspots.
Stateful vs. Stateless Brokering	Choose between strong delivery guarantees or higher throughput; hybrid models are possible.	Balances reliability and scalability.
Geographic Distribution	Use edge caching, regional routing, and location-based filtering.	Lowers latency, meets regulatory needs.

Table 1: Architectural Foundations for Scalable Event Distribution [3, 4]

Geographic distribution strategies have become essential as event platforms expand globally. Spatial data infrastructure principles provide frameworks for distributing event processing across geographic boundaries while maintaining performance. Edge caching positions frequently accessed message streams closer to consumers, reducing transit times and bandwidth requirements. Regional routing algorithms dynamically direct messages based on network conditions, regulatory boundaries, and latency requirements. These techniques must account for varied data sovereignty regulations, particularly for sensitive information. Location-based filtering approaches ensure notifications reach users through contextually appropriate devices based on spatial proximity and environmental factors [4].

III. De-duplication Strategies in Distributed Environments

Distributed event processing systems face fundamental consistency challenges when implementing exactly-once delivery semantics. Message-passing protocols must contend with network unreliability, process failures, and clock synchronization problems that become increasingly complex at scale. Various acknowledgment schemes attempt to address these challenges, including positive acknowledgment with retransmission and negative acknowledgment mechanisms. However, these approaches introduce complexity when scaled across numerous processing nodes spanning multiple geographic regions. Without careful design, message duplication becomes practically unavoidable without significant performance penalties [5].

Deterministic fingerprinting provides essential mechanisms for identifying duplicate messages regardless of arrival path or timing. Content-based techniques generate unique identifiers derived from message payloads using cryptographic hash functions that produce consistent outputs for identical inputs. Context-based fingerprinting extends this approach by incorporating metadata such as source identifiers, sequence numbers, and timestamps. The combination of content and contextual signals significantly enhances de-duplication accuracy, particularly when messages undergo transformation during processing while maintaining semantic equivalence [5].

Token-based verification systems address cross-device identity management in multi-surface environments. These frameworks ensure that each logical notification reaches its intended destination exactly once across heterogeneous device ecosystems. Master Data Management principles establish authoritative sources of truth for entity resolution, while record linkage techniques identify correspondences between different representations of the same entity across disparate systems. These approaches employ both deterministic rules for exact matching and probabilistic methods for handling incomplete identity information [6].

Machine learning approaches represent significant advancements over traditional rule-based de-duplication systems. Supervised learning techniques classify potential duplicates based on temporal, spatial, and content characteristics of message flows. Unsupervised methods identify natural groupings in message streams that may indicate duplication patterns without requiring labeled training data. Natural language processing techniques enhance capabilities for text-heavy messages by identifying semantic similarity despite surface-level differences [6].

The trade-offs between perfect de-duplication and processing overhead necessitate calibrated approaches based on application requirements. Achieving theoretical perfect guarantees typically requires synchronous verification mechanisms that introduce substantial latency. Probabilistic approaches using approximate data structures offer alternatives that reduce computational requirements while accepting a small probability of false negatives. Many implementations employ tiered strategies that apply progressively more expensive verification techniques based on message criticality [6].

IV. Multi-surface Orchestration and Intelligent Routing

Context-aware delivery systems transform notification distribution by considering user context across multiple devices. Modern platforms utilize contextual signals to determine optimal notification timing and channels based on factors including device usage patterns, time of day, location, and current activity. This approach improves user experience by reducing interruptions during focused activities while ensuring the timely delivery of important information. The system monitors active device usage and coordinates notification delivery accordingly, routing alerts to the most appropriate device based on current engagement patterns. Environmental factors further influence these decisions, with delivery strategies adapting based on whether a user appears to be in transit, at work, or at home [7].

Device capability recognition enables adaptive message transformation across heterogeneous endpoints. This framework identifies specific device characteristics, including screen dimensions, display type, audio capabilities, and input methods, to customize content appropriately. The transformation process may involve resizing visual elements, reformatting text, transcoding media, or changing modalities, such as converting visual alerts to audio notifications for delivery to screenless devices. These capabilities prove particularly valuable in multi-device ecosystems where users transition between different interfaces throughout their day [7].

Priority-based routing algorithms implement structured frameworks for categorizing and delivering notifications based on urgency and importance. This approach establishes formal models that consider both notification characteristics and recipient context. The framework classifies notifications along multiple dimensions, including time-sensitivity, personalization level, and required response actions. Routing logic incorporates user cognitive state and task engagement to minimize interruption costs while ensuring critical information reaches recipients promptly [8].

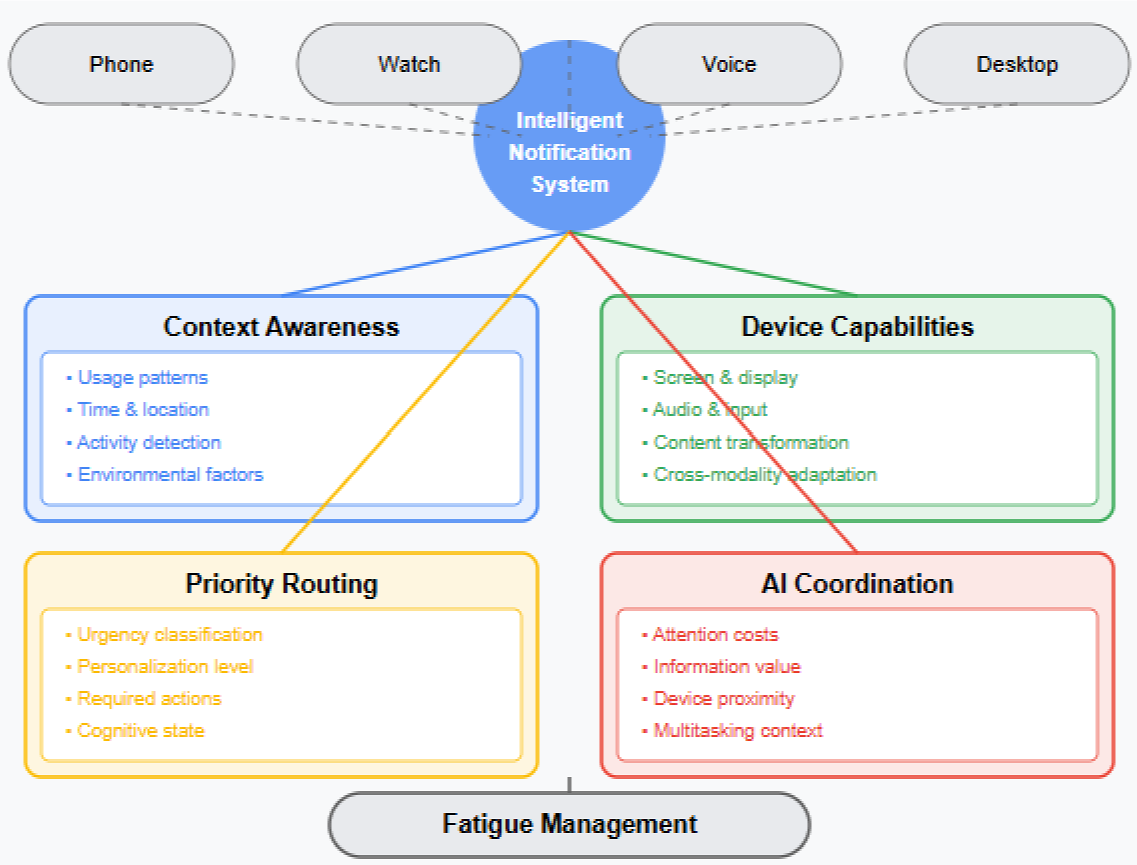

Fig 2: Multi-surface Orchestration [7, 8]

AI-driven coordination frameworks optimize delivery across device ecosystems by modeling interruption effects on user cognition. The management system evaluates potential notification strategies based on estimated attention costs and information value. This approach recognizes that each interruption imposes a cognitive burden through task switching, making indiscriminate notification delivery counterproductive. The coordination mechanism considers message importance, current activity, device proximity, and expected response requirements when determining optimal delivery timing and modality. By understanding the multitasking context, these systems make more informed decisions about when interruptions are justified [8].

V. Latency Engineering and Performance Optimization

End-to-end latency budgeting establishes structured performance management in large-scale distributed systems. This methodology divides processing pipelines into segments with specific performance allocations, creating enforceable contracts between system components. As highlighted in research on tail latency management, this approach recognizes that even when individual components have reasonable average performance, the overall user experience depends on complete operations across potentially thousands of servers and network paths. Effective budgeting accounts for the fundamental variability in distributed environments where performance fluctuates due to resource contention, background activities, and network conditions [9].

Predictive resource allocation leverages techniques for managing performance variability at scale. The strategy employs redundancy through multiple request issuance, effectively hedging against unpredictable tail latency by sending identical requests to multiple servers and using the first response. Micro-partitioning of work limits the impact radius of performance anomalies, ensuring slow operations affect only a small portion of overall system throughput. Cross-request adaptation dynamically adjusts resource allocation based on observed performance patterns, reducing allocation to inconsistent servers while favoring more predictable nodes [9].

Graceful degradation enables systems to maintain core functionality during load spikes through tiered service levels. Research emphasizes the importance of deadline-aware request scheduling, which prioritizes requests that can still meet service objectives while intelligently shedding load when necessary. This approach may involve early rejection of requests unlikely to be completed within acceptable timeframes and gradual quality reduction for services with flexible fidelity requirements. Tied requests maintain consistent user experiences by ensuring related operations receive coordinated handling, preventing inconsistencies during degraded operation [9].

Real-time monitoring forms the foundation of effective latency management in distributed systems. Research on latency elicitation demonstrates the importance of comprehensive instrumentation that captures performance across complex processing pipelines. Effective monitoring distinguishes between intrinsic delays inherent to processing requirements and queuing delays from resource contention, as these categories demand different optimization approaches. Closed-loop systems leverage this data to implement dynamic adjustments, continuously refining system behavior based on observed results [10].

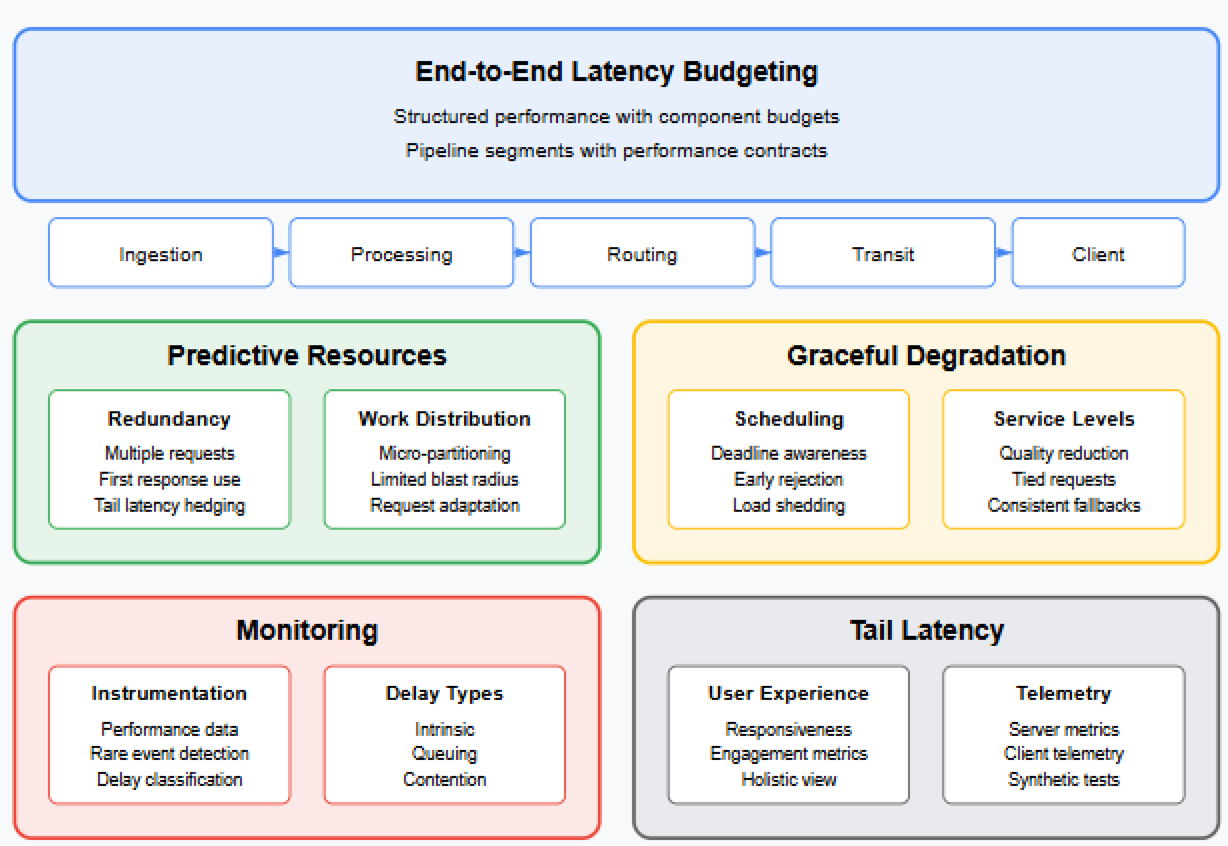

Fig 3: Latency Engineering [9, 10]

Conclusion

The evolution of real-time event platforms reflects broader shifts in distributed system design, emphasizing adaptability, intelligence, and user-centricity. Architectural patterns like distributed event aggregation and partitioned buses provide essential foundations for horizontal scaling, while sophisticated de-duplication strategies balance accuracy against performance costs. Context-aware orchestration represents a paradigm shift from application-centric to user-centric notification models, recognizing the complex interplay between information delivery and human attention. Latency management techniques acknowledge that user experience depends not just on average performance but on consistent responsiveness across all interactions. As these platforms continue evolving, artificial intelligence will play an increasingly central role, from predictive resource allocation to semantic message routing and automated incident response. Future advancements will likely focus on intent-based distribution, where systems understand not just what information to deliver but its purpose and significance within human workflows. The ultimate goal remains creating event distribution infrastructures that balance technical performance with human factors to deliver precisely the right information, at the right time, through the optimal channel.

Funding: This research received no external funding.
Conflicts of Interest: The authors declare no conflict of interest.

Publisher's Note: All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers.

References

[1] Alam Rahmatulloh et al., "Event-Driven Architecture to Improve Performance and Scalability in Microservices-Based Systems," ResearchGate, 2022. [Online]. Available: https://www.researchgate.net/publication/368431218_Event-Driven_Architecture_to_Improve_Performance_and_Scalability_in_Microservices-Based_Systems

[2] Saif Ahmad et al., "Optimizing IT Service Delivery with AI: Enhancing Efficiency Through Predictive Analytics and Intelligent Automation," ResearchGate, 2025. [Online]. Available: https://www.researchgate.net/publication/389447928_Optimizing_IT_Service_Delivery_with_AI_Enhancing_Efficiency_Through_Predictive_Analytics_and_Intelligent_Automation

[3] Navneet Kumar Pandey et al., "Distributed Event Aggregation for Publish/Subscribe Systems," ResearchGate, 2013. [Online]. Available: https://www.researchgate.net/publication/317091881_Distributed_Event_Aggregation_for_PublishSubscribe_Systems

[4] Matthes Rieke et al., "Geospatial IoT; The Need for Event-Driven Architectures in Contemporary Spatial Data Infrastructures," MDPI, 2018. [Online]. Available: https://www.mdpi.com/2220-9964/7/10/385

[5] Myat Pwint Phyu and Ni Lar Thein, "Using Efficient Deduplication Method in Large-scale Distributed Storage System". [Online]. Available: https://www.uit.edu.mm/wp-content/uploads/2020/05/MPP-6.pdf

[6] Phillip Thomas, "Machine Learning for Data Deduplication and Record Linkage in MDM Systems," ResearchGate, 2025. [Online]. Available: https://www.researchgate.net/publication/394384136_Machine_Learning_for_Data_Deduplication_and_Record_Linkage_in_MDM_Systems

[7] Technical Disclosure Commons, "User Context Aware Notification Delivery," Defensive Publications Series, 2022. [Online]. Available: https://www.tdcommons.org/cgi/viewcontent.cgi?article=6598&context=dpubs_series

[8] Shamsi T. Iqbal, "A Framework For Intelligent Notification Management In Multitasking Domains," ResearchGate, 2008. [Online]. Available: https://www.researchgate.net/publication/32964914_A_Framework_For_Intelligent_Notification_Management_In_Multitasking_Domains

[9] Rajrup Ghosh and Yogesh Simmhan, "Distributed Scheduling of Event Analytics across Edge and Cloud," arXiv:1608.01537v4, 2017. [Online]. Available: https://arxiv.org/pdf/1608.01537

[10] Cheng Zhang et al., "Tail-Learning: Adaptive Learning Method for Mitigating Tail Latency in Autonomous Edge Systems," ACM, 2025. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/3737289