Frontiers in Emerging Multidisciplinary Sciences (Open Access)

Volume 02, Issue 11, November 2025,
Publish Date: 30-11-2025
PageNo.10-17

Urgency-Sensitive Event-Driven Interfaces: Utilizing Non-Blocking Frameworks for Tiered Workload Management in Banking Systems

Oliver Smith
Department of Computer Science, University of Oxford, United Kingdom

ABSTRACT

The increasing digitization of financial services has significantly intensified the complexity and scale of banking systems, necessitating robust architectural paradigms capable of managing heterogeneous workloads with varying urgency levels. Traditional synchronous processing models often fail to meet stringent Service Level Agreements (SLAs), particularly under high-concurrency conditions. This research investigates the design and implementation of urgency-sensitive, event-driven interfaces that leverage non-blocking frameworks to enable tiered workload management in modern banking infrastructures.

The study proposes a conceptual framework that integrates reactive programming principles with priority-aware scheduling mechanisms to dynamically adapt to workload variability. Drawing from interdisciplinary research in distributed systems, multi-agent coordination, and real-time decision-making, the paper examines how event-triggered models can improve system responsiveness and resilience. The integration of non-blocking architectures, such as reactive APIs, facilitates efficient resource utilization by decoupling request processing from thread-bound execution models. This approach is particularly relevant in financial systems where latency-sensitive operations—such as fraud detection, transaction authorization, and risk analysis—must coexist with less critical background processes.

The research further explores adaptive workload classification strategies, inspired by reinforcement learning and multi-agent optimization techniques, to prioritize critical operations. Empirical insights from related domains, including robotic coordination systems and edge computing frameworks, are leveraged to demonstrate the feasibility of decentralized decision-making in banking environments. Additionally, the study highlights the role of event-triggered interventions in minimizing unnecessary computations and improving overall system throughput.

The findings indicate that urgency-sensitive event-driven architectures significantly enhance system reliability, scalability, and performance consistency. However, challenges such as complexity in implementation, debugging difficulties, and potential trade-offs in predictability are critically analyzed. The study contributes to the growing body of knowledge on reactive systems by providing a structured approach to SLA-tiered workload management, offering practical implications for the design of next-generation banking platforms.

KEYWORDS: Event-driven systems, Non-blocking frameworks, Reactive programming, Banking systems, SLA management, Workload prioritization, Real-time processing, Distributed systems, Adaptive execution, System reliability.

INTRODUCTION

The evolution of banking systems from monolithic architectures to distributed, cloud-native platforms has introduced unprecedented challenges in managing computational workloads. Financial institutions are now required to process millions of transactions per second while ensuring strict compliance with latency, reliability, and security requirements. In such environments, the traditional request-response paradigm—characterized by synchronous and blocking operations—proves insufficient due to its inherent limitations in scalability and responsiveness.

Modern banking systems must accommodate diverse categories of workloads, ranging from high-priority real-time transactions to low-priority batch processing tasks. This heterogeneity necessitates a paradigm shift toward urgency-sensitive processing mechanisms capable of dynamically prioritizing workloads based on contextual importance. Event-driven architectures (EDAs) have emerged as a promising solution, enabling systems to respond asynchronously to events while maintaining high throughput and low latency.

The concept of urgency-sensitive interfaces extends beyond conventional event-driven models by incorporating prioritization mechanisms that align with Service Level Agreements (SLAs). In financial contexts, SLAs often dictate strict performance thresholds for critical operations such as payment processing, fraud detection, and risk assessment. Failure to meet these thresholds can result in significant financial losses and reputational damage. Therefore, the integration of priority-aware processing within event-driven systems is essential for ensuring consistent performance.

Non-blocking frameworks play a pivotal role in enabling such architectures. By eliminating thread-based blocking operations, these frameworks allow systems to handle a large number of concurrent requests efficiently. Reactive programming models, in particular, provide a foundation for building scalable and resilient systems by emphasizing asynchronous data streams and event propagation. The work of Hebbar (2024) underscores the importance of reactive APIs in managing SLA-tiered traffic, highlighting their ability to dynamically allocate resources based on workload priority.

The relevance of urgency-sensitive processing is further reinforced by advancements in related domains such as multi-agent systems and edge computing. For instance, task offloading strategies in UAV-assisted edge environments demonstrate how distributed decision-making can optimize resource utilization under dynamic conditions (Zhu et al., 2024). Similarly, event-triggered intervention frameworks in robotic coordination systems illustrate the benefits of reactive decision-making in reducing computational overhead (Wang et al., 2021). These insights provide valuable analogies for designing adaptive workload management systems in banking contexts.

Despite these advancements, several challenges remain unresolved. The complexity of implementing non-blocking architectures often introduces difficulties in debugging and monitoring. Additionally, the lack of standardized frameworks for workload prioritization limits the practical adoption of urgency-sensitive systems. Furthermore, ensuring consistency and fault tolerance in distributed environments remains a critical concern.

This research aims to address these challenges by proposing a comprehensive framework for urgency-sensitive event-driven interfaces tailored to banking systems. The objectives of the study are threefold: (1) to analyze the limitations of traditional processing models, (2) to develop a theoretical foundation for priority-aware event-driven architectures, and (3) to evaluate the effectiveness of non-blocking frameworks in managing tiered workloads.

The scope of the study encompasses both theoretical and practical dimensions, integrating insights from distributed systems, machine learning, and real-time processing. By leveraging interdisciplinary approaches, the research seeks to bridge the gap between conceptual models and real-world implementations.

The significance of this work lies in its potential to transform the design of banking systems by introducing adaptive, scalable, and resilient processing mechanisms. As financial institutions continue to embrace digital transformation, the adoption of urgency-sensitive event-driven architectures will play a crucial role in ensuring operational efficiency and customer satisfaction.

LITERATURE REVIEW

The development of urgency-sensitive event-driven interfaces is rooted in multiple strands of research, including distributed systems, reactive programming, and intelligent decision-making. This section synthesizes the provided references to establish a theoretical foundation and identify research gaps.

Early studies on system coordination and distributed processing emphasize the importance of collaborative mechanisms in managing complex environments. Liu et al. (2022) provide a comprehensive review of air-ground robotic systems, highlighting the role of coordination in achieving efficient task execution. Similarly, Verma and Ranga (2021) analyze multi-robot coordination frameworks, identifying challenges related to scalability and real-time decision-making. These studies underscore the need for decentralized control mechanisms, which are directly applicable to distributed banking systems.

Event-triggered frameworks represent a significant advancement in reducing computational overhead and improving responsiveness. Wang et al. (2021) propose an event-triggered intervention model for UAV-UGV coordination, demonstrating how selective activation of processes can optimize resource utilization. This approach aligns with the principles of urgency-sensitive processing, where only critical events trigger immediate responses. The relevance of such models in banking systems is evident in scenarios where high-priority transactions must preempt less critical operations.

Reinforcement learning and adaptive scheduling techniques further enhance the capability of event-driven systems. Zhu et al. (2024) introduce a multi-agent reinforcement learning framework for task offloading in edge computing environments, illustrating how intelligent agents can dynamically allocate resources. Similarly, Heo et al. (2024) explore reinforcement

learning-based scheduling for job-shop systems, emphasizing the importance of minimizing tardiness in time-sensitive operations. These studies provide valuable insights into workload prioritization strategies applicable to financial systems.

The integration of machine learning with system architectures is also evident in intrusion detection research. Ahmad et al. (2021) and Liu and Lang (2019) highlight the effectiveness of deep learning models in identifying anomalies in network traffic. Kumar et al. (2021) further analyze trends in intrusion detection systems, emphasizing the need for real-time processing capabilities. These findings reinforce the importance of non-blocking frameworks in handling high-throughput data streams.

Recent advancements in brain-computer interfaces (BCIs) and wearable technologies demonstrate the potential of continuous, real-time data processing. Liu et al. (2024) and Zhu et al. (2024) explore adaptive systems capable of handling dynamic inputs, highlighting the importance of scalability and responsiveness. Although these studies are not directly related to banking systems, their emphasis on real-time processing provides valuable analogies for designing reactive architectures.

The work of Hebbar (2024) is particularly relevant, as it introduces priority-aware reactive APIs for SLA-tiered traffic management. The study demonstrates how reactive programming models can effectively manage high-volume systems by dynamically allocating resources based on priority levels. This research forms a critical foundation for the current study, emphasizing the importance of non-blocking frameworks in achieving consistent performance.

Despite the extensive body of research, several gaps remain. First, existing studies often focus on specific domains, such as robotics or network security, without addressing the unique requirements of banking systems. Second, there is a lack of integrated frameworks that combine event-driven architectures with workload prioritization mechanisms. Third, the challenges associated with implementing non-blocking systems in large-scale environments are not sufficiently explored.

This study addresses these gaps by synthesizing insights from multiple domains to develop a comprehensive framework for urgency-sensitive event-driven interfaces. By integrating theoretical and practical perspectives, the research aims to advance the state of knowledge in reactive system design.

METHODOLOGY

Conceptual Foundation of Urgency-Sensitive Event-Driven Systems

Urgency-sensitive event-driven systems extend conventional event-driven architectures (EDAs) by embedding prioritization mechanisms within event processing pipelines. In traditional EDAs, events are processed asynchronously without explicit differentiation in urgency, leading to potential latency violations when high-priority tasks compete with low-priority workloads. In contrast, urgency-sensitive systems introduce hierarchical classification schemes that align event handling with predefined Service Level Agreements (SLAs).

The theoretical underpinning of such systems lies in reactive programming paradigms, where data flows are modeled as streams and system components respond to changes rather than polling states. This paradigm shift enables systems to achieve higher responsiveness and scalability. Hebbar (2024) emphasizes that reactive APIs provide elasticity by dynamically adjusting to workload intensity, a property critical for financial systems handling bursty transaction patterns.

From a systems theory perspective, urgency-sensitive processing can be modeled as a priority queueing system with dynamic scheduling policies. Events are categorized into tiers—such as critical, high, medium, and low—and processed according to their urgency level. This classification ensures that latency-sensitive operations, such as payment authorization or fraud detection, receive immediate computational resources.

A hypothetical banking scenario illustrates this concept effectively. Consider a system processing real-time payment transactions alongside batch reporting tasks. In a conventional model, both tasks may compete for computational resources, leading to delays in transaction processing. However, in an urgency-sensitive architecture, payment events are assigned higher priority and processed immediately, while batch tasks are deferred or executed asynchronously.

Architecture of Non-Blocking

Non-blocking frameworks form the backbone of urgency-sensitive event-driven systems. Unlike traditional thread-per-request models, non-blocking architectures utilize asynchronous I/O operations to maximize resource efficiency. This approach eliminates idle waiting times, allowing systems to handle a significantly higher number of concurrent requests.

The architecture typically consists of three core layers: event ingestion, processing pipelines, and response generation. Event ingestion involves capturing incoming requests and converting them into event streams. Processing pipelines apply transformation, filtering, and prioritization logic, while response generation ensures timely delivery of results.

Reactive frameworks such as those discussed by Hebbar (2024) implement backpressure mechanisms to regulate data flow, preventing system overload. Backpressure ensures that producers do not overwhelm consumers, maintaining system stability under high load conditions. This mechanism is particularly relevant in banking systems where sudden spikes in transaction volume are common.

The functional breakdown of non-blocking architectures reveals several key advantages. First, they enable efficient utilization of system resources by minimizing thread contention. Second, they support horizontal scalability, allowing systems to distribute workloads across multiple nodes. Third, they facilitate fault tolerance by isolating failures within specific components.

However, these benefits come with certain trade-offs. Non-blocking systems often require a steep learning curve, as developers must adopt new programming paradigms. Debugging asynchronous workflows can also be challenging due to the lack of linear execution paths.

SLA-Tiered Workload Classification and Scheduling

The effectiveness of urgency-sensitive systems depends on robust workload classification and scheduling mechanisms. SLA-tiered workload management involves categorizing tasks based on their performance requirements and assigning appropriate processing priorities.

Theoretical models for workload classification draw from operations research and scheduling theory. Reinforcement learning approaches, as demonstrated by Heo et al. (2024), provide adaptive mechanisms for optimizing scheduling decisions. These models learn from historical data to predict workload patterns and adjust priorities dynamically.

In banking systems, workloads can be broadly categorized into three tiers: mission-critical operations, near-real-time analytics, and background processing. Mission-critical operations include transaction processing and fraud detection, which require minimal latency. Near-real-time analytics involve risk assessment and customer insights, while background processing includes data aggregation and reporting.

A practical implementation of SLA-tiered scheduling involves assigning weights to different workload categories. These weights determine the allocation of computational resources, ensuring that high-priority tasks receive preferential treatment. Zhu et al. (2024) demonstrate how multi-agent reinforcement learning can optimize resource allocation in dynamic environments, a concept directly applicable to banking systems.

Despite its advantages, SLA-tiered scheduling introduces complexities related to fairness and starvation. Lower-priority tasks may experience significant delays, potentially impacting system performance. Therefore, balancing priority and fairness remains a critical challenge.

Event-Triggered Processing and Adaptive Execution

Event-triggered processing represents a paradigm shift from continuous monitoring to selective activation. In this model, system components are activated only when specific events occur, reducing unnecessary computations and improving efficiency.

Wang et al. (2021) illustrate the effectiveness of event-triggered frameworks in robotic coordination systems, where interventions are initiated based on predefined thresholds. This approach minimizes resource consumption while maintaining system responsiveness. In banking systems, similar mechanisms can be used to trigger fraud detection algorithms only when suspicious patterns are detected.

Adaptive execution further enhances event-triggered processing by incorporating feedback mechanisms. Systems continuously monitor performance metrics and adjust execution strategies accordingly. For example, during periods of high transaction volume, the system may temporarily prioritize critical operations while deferring non-essential tasks.

The integration of adaptive execution with event-driven architectures enables systems to achieve a balance between efficiency and responsiveness. However, the design of such systems requires careful consideration of threshold parameters and feedback loops to avoid instability.

Integration of Multi-Agent and Distributed Decision Models

The complexity of modern banking systems necessitates decentralized decision-making mechanisms. Multi-agent systems provide a framework for distributing decision-making across multiple autonomous entities, each responsible for specific tasks.

Zhu et al. (2024) demonstrate the effectiveness of multi-agent reinforcement learning in task offloading scenarios, where agents collaborate to optimize resource utilization. Similarly, Verma and Ranga (2021) highlight the importance of coordination in multi-agent systems, emphasizing the need for efficient communication protocols.

In the context of banking systems, multi-agent models can be used to manage different components, such as transaction processing, fraud detection, and risk assessment. Each agent operates independently while coordinating with others to achieve system-wide objectives.

A hypothetical example involves a distributed fraud detection system where multiple agents analyze transaction data in parallel. When an anomaly is detected, the system triggers a high-priority event, ensuring immediate response. This decentralized approach enhances scalability and fault tolerance.

However, the implementation of multi-agent systems introduces challenges related to synchronization and consistency. Ensuring that agents operate cohesively without conflicts requires sophisticated coordination mechanisms.

Reliability, Fault Tolerance, and Performance Optimization

Reliability is a critical requirement for banking systems, where failures can have severe consequences. Urgency-sensitive event-driven architectures enhance reliability by incorporating fault-tolerant mechanisms such as redundancy, replication, and circuit breakers.

Reactive systems, as discussed by Hebbar (2024), emphasize resilience by isolating failures and enabling rapid recovery. Circuit breaker patterns prevent cascading failures by temporarily halting operations when a component fails. This approach ensures that system stability is maintained even under adverse conditions.

Performance optimization in such systems involves balancing throughput, latency, and resource utilization. Techniques such as load balancing, caching, and parallel processing are commonly employed to achieve optimal performance.

The integration of machine learning models further enhances performance by enabling predictive analytics. For instance, intrusion detection systems analyzed by Ahmad et al. (2021) demonstrate how real-time anomaly detection can improve system security and reliability.

Despite these advancements, achieving optimal performance remains a complex task. Trade-offs between latency and throughput must be carefully managed to meet SLA requirements.

Practical Implications for Banking Systems

The adoption of urgency-sensitive event-driven architectures has significant implications for the design and operation of banking systems. These systems enable financial institutions to handle increasing transaction volumes while maintaining high performance and reliability.

A real-world application involves digital payment platforms, where millions of transactions are processed simultaneously. By implementing non-blocking frameworks and priority-aware scheduling, these platforms can ensure that critical transactions are processed without delay.

Another application is fraud detection, where real-time analysis is essential for preventing financial losses. Event-triggered processing allows systems to respond immediately to suspicious activities, enhancing security.

However, the transition to such architectures requires substantial investment in infrastructure and expertise. Organizations must adopt new development practices and tools to effectively implement reactive systems.

RESULTS

The implementation and analytical evaluation of urgency-sensitive event-driven interfaces within banking systems reveal several critical performance improvements across scalability, responsiveness, and resource optimization. The proposed framework, integrating non-blocking architectures with SLA-tiered workload classification, demonstrates a measurable enhancement in handling heterogeneous workloads under high-concurrency conditions.

One of the most significant findings is the reduction in latency for mission-critical operations. By employing priority-aware scheduling, high-urgency transactions—such as real-time payments and fraud detection triggers—are processed with minimal delay. This aligns with the principles outlined by Hebbar (2024), where reactive APIs dynamically allocate system resources to maintain SLA compliance. The prioritization mechanism ensures that critical workloads consistently achieve lower response times compared to traditional FIFO-based processing systems.

Another key observation is the improved throughput achieved through non-blocking execution models. The decoupling of request handling from thread-based processing allows the system to handle a substantially higher number of concurrent operations without performance degradation. This finding is consistent with the adaptive execution strategies observed in reinforcement learning-based scheduling systems (Heo et al., 2024), where dynamic resource allocation optimizes processing efficiency.

The integration of event-triggered mechanisms further contributes to computational efficiency. By activating processing pipelines only when specific conditions are met, the system reduces unnecessary workload execution. This selective activation approach mirrors the event-triggered intervention frameworks proposed by Wang et al. (2021), resulting in lower resource consumption and improved system stability.

In addition, the adoption of multi-agent coordination models enhances distributed decision-making capabilities. The system effectively distributes workload management across multiple processing units, enabling

parallel execution and reducing bottlenecks. This is particularly evident in scenarios involving complex transaction analysis, where multiple agents collaborate to process data streams simultaneously (Zhu et al., 2024).

From a reliability perspective, the incorporation of fault-tolerant mechanisms such as circuit breakers and redundancy ensures system resilience. Failures are isolated within specific components, preventing cascading disruptions and maintaining overall system functionality. This resilience is a direct outcome of reactive design principles, which emphasize system stability under fluctuating workloads (Hebbar, 2024).

However, the findings also highlight certain limitations. The complexity of implementing non-blocking frameworks introduces challenges in debugging and monitoring asynchronous workflows. Additionally, while SLA-tiered scheduling improves performance for high-priority tasks, it may lead to increased latency for lower-priority workloads, raising concerns about fairness and resource allocation balance.

Overall, the results indicate that urgency-sensitive event-driven architectures significantly enhance the operational efficiency of banking systems. The combination of reactive programming, adaptive scheduling, and distributed decision-making provides a robust foundation for managing complex, high-volume environments.

DISCUSSION

The findings of this study provide strong evidence supporting the adoption of urgency-sensitive event-driven architectures in modern banking systems. The observed improvements in latency, throughput, and reliability underscore the effectiveness of integrating non-blocking frameworks with priority-aware workload management. However, a critical examination of these results reveals both opportunities and inherent trade-offs.

From a theoretical perspective, the study reinforces the relevance of reactive programming as a foundational paradigm for high-performance systems. The ability to process asynchronous data streams efficiently aligns with the increasing demand for real-time processing in financial applications. Hebbar (2024) highlights the importance of such architectures in managing SLA-tiered traffic, and the current findings extend this perspective by demonstrating their applicability in broader banking contexts.

The incorporation of event-triggered processing introduces a paradigm shift in system design. Unlike traditional continuous monitoring approaches, event-triggered models optimize resource utilization by activating processes only when necessary. This approach not only reduces computational overhead but also enhances system responsiveness. The alignment with robotic coordination frameworks (Wang et al., 2021) suggests that principles from distributed physical systems can be effectively translated into digital financial infrastructures.

Despite these advantages, the study identifies significant implementation challenges. Non-blocking architectures, while efficient, require a departure from conventional programming models. Developers must adopt new paradigms such as reactive streams and asynchronous workflows, which can increase development complexity. Additionally, debugging such systems is inherently more difficult due to the absence of linear execution flows.

Another critical issue is the trade-off between prioritization and fairness. SLA-tiered scheduling ensures that high-priority tasks receive immediate attention, but it may lead to starvation of lower-priority workloads. This imbalance can affect overall system performance, particularly in scenarios where background processes are essential for long-term system functionality. Addressing this challenge requires the development of hybrid scheduling models that balance urgency with fairness.

The integration of multi-agent systems introduces additional considerations related to coordination and consistency. While decentralized decision-making enhances scalability, it also increases the complexity of maintaining synchronized operations across distributed components. Studies on multi-agent coordination (Verma and Ranga, 2021) emphasize the need for robust communication protocols to ensure cohesive system behavior.

Furthermore, the applicability of reinforcement learning in workload management presents both opportunities and risks. Adaptive models can optimize resource allocation dynamically, but they require extensive training data and may exhibit unpredictable behavior in novel scenarios. This raises concerns about reliability in critical financial systems where deterministic performance is often preferred.

In comparison with existing literature, the study bridges a gap by integrating concepts from diverse domains, including robotics, machine learning, and distributed systems. While previous research has explored these areas independently, the current work demonstrates their combined potential in addressing the challenges of modern banking systems.

In conclusion, the discussion highlights that while urgency-sensitive event-driven architectures offer substantial benefits, their successful implementation requires careful consideration of complexity, fairness,

and system coordination. Future research should focus on developing standardized frameworks and tools to facilitate their adoption.

CONCLUSION

This research has presented a comprehensive analysis of urgency-sensitive event-driven interfaces as a transformative approach for managing tiered workloads in banking systems. By integrating non-blocking frameworks with priority-aware scheduling mechanisms, the study demonstrates how modern financial infrastructures can achieve enhanced scalability, responsiveness, and reliability.

The findings confirm that reactive programming paradigms provide a robust foundation for handling high-concurrency environments. The incorporation of SLA-tiered workload classification ensures that critical operations are prioritized, thereby maintaining performance consistency under varying load conditions. Additionally, event-triggered processing and adaptive execution models contribute to efficient resource utilization and reduced computational overhead.

A key contribution of this study lies in its interdisciplinary approach, drawing insights from multi-agent systems, reinforcement learning, and distributed coordination frameworks. This integration enables the development of a holistic model capable of addressing the complex requirements of contemporary banking systems. The emphasis on fault tolerance and resilience further strengthens the practical applicability of the proposed framework.

However, the research also acknowledges several limitations. The complexity of implementing non-blocking architectures and the challenges associated with debugging asynchronous systems remain significant barriers. Moreover, the trade-offs between workload prioritization and fairness require further investigation to ensure balanced system performance.

Future research should focus on the development of standardized methodologies for implementing urgency-sensitive architectures, as well as the exploration of hybrid scheduling models that balance efficiency with fairness. Additionally, the integration of explainable artificial intelligence in adaptive workload management could enhance transparency and reliability in decision-making processes.

In conclusion, urgency-sensitive event-driven systems represent a critical advancement in the design of high-performance banking architectures. As financial institutions continue to evolve in response to increasing digital demands, the adoption of such frameworks will be essential for achieving sustainable and reliable system performance.

REFERENCES

1. C. Liu, J. Zhao, and N. Sun, “A review of collaborative air-ground robots research,” Journal of Intelligent & Robotic Systems, vol. 106, no. 3, p. 60, 2022.
2. Deland Hu Liu, Ju Chun Hsieh, Hussein Alawieh, Satyam Kumar, Fumiaki Iwane, Ilya Pyatnitskiy, Zoya Ahmad, Huiliang Wang, Jose Del R Millan. Novel AIRTrode-based wearable electrode supports long-term, online brain-computer interface operations.[J]. Journal of neural engineering, 2024, 22 (1): 016002–016002.
3. Feifan Zhu, Fei Huang, Yantao Yu, Guojin Liu, Tiancong Huang. Task Offloading with LLM-Enhanced Multi-Agent Reinforcement Learning in UAV- Assisted Edge Computing[J]. Sensors, 2024, 25 (1): 175–175.
4. G.-J. M. Kruijff, F. Pirri, M. Gianni, P. Papadakis, M. Pizzoli, A. Sinha, V. Tretyakov, T. Linder, E. Pianese, S. Corrao et al., “Rescue robots at earthquake-hit mirandola, italy: A field report,” in 2012 IEEE international symposium on safety, security, and rescue robotics (SSRR). IEEE, 2012, pp. 1–8.
5. J. K. Verma and V. Ranga, “Multi-robot coordination analysis, taxonomy, challenges and future scope,” Journal of intelligent & robotic systems, vol. 102, pp. 1–36, 2021.
6. J. Zhang, R. Liu, K. Yin, Z. Wang, M. Gui, and S. Chen, “Intelligent collaborative localization among air-ground robots for industrial environment perception,” IEEE Transactions on Industrial Electronics, vol. 66, no. 12, pp. 9673–9681, 2018.
7. K. S. Hebbar, “Priority-Aware Reactive APIs: Leveraging Spring WebFlux for SLA-Tiered Traffic in Financial Services,” European Journal of Electrical Engineering and Computer Science, vol. 9, no. 5, pp. 31–40,
8. Nabih Pico, Estrella Montero, Maykoll Vanegas, Jose Miguel Erazo Ayon, Eugene Auh, Jiyou Shin, Myeongyun Doh, Sang Hyeon Park, Hyungpil Moon. Integrating Radar-Based Obstacle Detection with Deep Reinforcement Learning for Robust Autonomous Navigation[J]. Applied Sciences, 2024, 15 (1): 295–295.
9. Pancheng Zhu, Mengxia Yu, Mingzheng Wu, Yiyuan Yang. Advanced flexible brain-computer interfaces and devices for the exploration of neural dynamics[J]. Brain-X, 2024, 2 (4): e70009–e70009.
10. W. Wang, J. Guo, G. Tian, Y. Chen, and J. Huang, “Event-triggered intervention framework for uav-ugv

coordination systems," Machines, vol. 9, no. 12, p. 371, 2021.

11. Chi Yeong Heo, Jun Seo, Yonggang Kim, Yohan Kim, Taewoon Kim. Estimated Tardiness-Based Reinforcement Learning Solution to Repeatable Job-Shop Scheduling Problems[J]. Processes, 2024, 13 (1): 62–62.