Advanced Journal of Science, Technology and Engineering
ISSN: 2997-5972
Volume 5, Issue 3, 2025 (pp. 1-19)
www.abjournals.org

MIGRATING MONOLITHIC E-COMMERCE SYSTEMS TO MICROSERVICES: A SYSTEMATIC REVIEW OF EVENT-DRIVEN ARCHITECTURE APPROACHES

Stephen W. Bulus[1*], Olubukola D. Adekola[1], Folasade Y. Ayankoya[2], Oluwabamise J. Adeniyi[1], and Ayodeji G. Abiodun[2]

[1]Department of Software Engineering, School of Computing, Babcock University, Ilishan-Remo, Ogun State, Nigeria.

[2]Department of Computer Science, School of Computing, Babcock University, Ilishan-Remo, Ogun State, Nigeria.

[*]Corresponding Author's Email: bulus0155@pg.babcock.edu.ng

Cite this article:

Bulus, S. W., Adekola, O. D., Ayankoya, F. Y., Adeniyi, O. J., Abiodun, A. G. (2025), Migrating Monolithic E-commerce Systems to Microservices: A Systematic Review of Event-Driven Architecture Approaches. Advanced Journal of Science, Technology and Engineering 5(3), 1-19. DOI: 10.52589/AJSTE-O1G0V4GO

Manuscript History

Received: 14 Jul 2025

Accepted: 14 Aug 2025

Published: 14 Oct 2025

ABSTRACT: *The transformation from monolithic to microservices architectures in e-commerce systems has gained significant traction as organizations seek enhanced scalability, resilience, and operational agility. This systematic review examines the current state of research on migrating monolithic e-commerce systems to microservices using event-driven architecture (EDA) approaches. Following PRISMA guidelines, we analyzed 11 relevant studies focusing on decomposition frameworks, migration methodologies, and implementation strategies specific to e-commerce contexts. Our findings reveal that event-driven architectures provide superior scalability and fault tolerance compared to traditional synchronous communication patterns, with asynchronous messaging showing 30-40% better performance under high load conditions. Key migration approaches include the Strangler Fig Pattern, Domain-Driven Design (DDD) decomposition, and process mining-based identification of service boundaries. However, challenges persist in data consistency management, service communication complexity, and organizational alignment during migration. The review identifies critical gaps in standardized metrics for evaluating migration success and limited tooling support for automated decomposition in e-commerce-specific contexts. This study contributes to the understanding of event-driven microservices migration patterns and provides actionable insights for practitioners undertaking e-commerce modernization initiatives.*

KEYWORDS: Microservices Migration, E-commerce Architecture, Event-driven Architecture, Monolithic Decomposition, Systematic Review, Service-oriented Architecture.

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

Advanced Journal of Science, Technology and Engineering
ISSN: 2997-5972
Volume 5, Issue 3, 2025 (pp. 1-19)

www.abjournals.org

INTRODUCTION

The rapid evolution of e-commerce platforms has necessitated architectural paradigm shifts to meet increasing demands for scalability, reliability, and agility [2]. Traditional monolithic architectures, while providing simplicity in development and deployment, often become bottlenecks as e-commerce systems scale to handle millions of transactions and users [3]. The centralized nature of monolithic systems creates inherent limitations in terms of independent scaling, technology diversity, and fault isolation, making them unsuitable for modern e-commerce requirements.

Microservices architecture has emerged as a compelling alternative, offering the promise of enhanced scalability, improved fault tolerance, and greater development agility [1]. By decomposing monolithic applications into smaller, independently deployable services, organizations can achieve better resource utilization, faster deployment cycles, and improved system resilience. Research indicates that companies implementing microservices architectures report 20-50% reduction in development time for new features and significant improvements in system availability [5].

Event-driven architecture (EDA) patterns have gained particular attention in the context of microservices migration, especially for e-commerce systems where business processes are inherently event-oriented. Customer actions such as placing orders, updating profiles, or processing payments naturally translate to domain events that can drive system behavior [5]. The asynchronous nature of event-driven communication provides better decoupling between services, enhanced fault tolerance, and improved scalability compared to synchronous request-response patterns [4].

Despite growing interest in microservices adoption, the migration from monolithic e-commerce systems remains a complex undertaking fraught with technical and organizational challenges. Abgaz et al. [1] note that monolith decomposition into microservices remains at an early stage, with insufficient standardized methods for combining static, dynamic, and evolutionary data analysis. The absence of established metrics, datasets, and baselines for evaluating migration success further complicates decision-making for practitioners.

This systematic review aims to synthesize current research on migrating monolithic e-commerce systems to microservices using event-driven architecture approaches. We examine decomposition methodologies, implementation strategies, performance implications, and lessons learned from industry practitioners. The review addresses critical gaps in understanding optimal migration patterns for e-commerce contexts and provides evidence-based guidance for organizations undertaking this transformation.

Aim and Objectives

Aim

The primary aim of this systematic review is to comprehensively analyze and synthesize existing research on migrating monolithic e-commerce systems to microservice architectures using event-driven patterns, examining methodologies, challenges, benefits, and implementation strategies.

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

Objectives

1. To identify and categorize decomposition frameworks and methodologies used for monolith-to-microservices migration in e-commerce contexts.
2. To examine the role and effectiveness of event-driven architecture patterns in microservices migration.
3. To analyze performance implications and benefits achieved through migration to event-driven microservices.
4. To identify key challenges and limitations encountered during migration processes.
5. To evaluate tools and techniques supporting automated decomposition and migration.
6. To assess organizational and operational considerations for successful migration implementation.
7. To provide evidence-based recommendations for practitioners undertaking e-commerce modernization initiatives.

METHODS

This systematic review adhered to the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) guidelines to ensure transparency and methodological rigor. A comprehensive literature search was conducted to identify relevant studies published between 2015 and 2025, focusing on the intersection of microservices migration, e-commerce systems, and event-driven architectures.

Inclusion Criteria

- Studies focusing on monolithic to microservices migration methodologies.
- Research examining event-driven architecture patterns in microservices contexts.
- Papers discussing e-commerce or retail system modernization.
- Studies evaluating decomposition techniques and frameworks.
- Research presenting tools or frameworks for automated migration support.
- Publications in English language between 2015-2025.

Exclusion Criteria

- Studies focusing solely on greenfield microservices development.
- Papers discussing general distributed systems without migration focus.
- Research limited to theoretical frameworks without empirical validation.

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

- Publications not peer-reviewed or lacking methodological rigor.
- Studies not addressing e-commerce or similar transactional systems.

Search Strategy

The search strategy employed multiple academic databases and used targeted keyword combinations:

1. **Primary Search Terms:**

a. ("microservices migration" OR "monolith decomposition") AND ("e-commerce" OR "ecommerce" OR "retail") AND ("event-driven" OR "event sourcing" OR "asynchronous messaging").

b. ("service decomposition" OR "architectural transformation") AND ("online retail" OR "digital commerce") AND ("message queue" OR "event streaming").

2. **Secondary Search Terms:**

a. "strangler fig pattern" AND "e-commerce modernization".

b. "domain-driven design" AND "microservices decomposition".

c. "CQRS" AND "event sourcing" AND "retail systems".

Study Selection Process

The study selection followed a systematic four-stage approach:

1. **Initial Search:** Comprehensive database search yielding initial paper set.
2. **Title and Abstract Screening:** Application of inclusion/exclusion criteria to titles and abstracts.
3. **Full-Text Assessment:** Detailed evaluation of potentially relevant papers.
4. **Final Selection:** Expert review and consensus on final paper inclusion.

For a visual representation of the study selection process, refer to Figure 1, which depicts a PRISMA flowchart illustrating the stages from initial identification to the final selection of documents for review:

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

Advanced Journal of Science, Technology and Engineering
ISSN: 2997-5972
Volume 5, Issue 3, 2025 (pp. 1-19)
www.abjournals.org

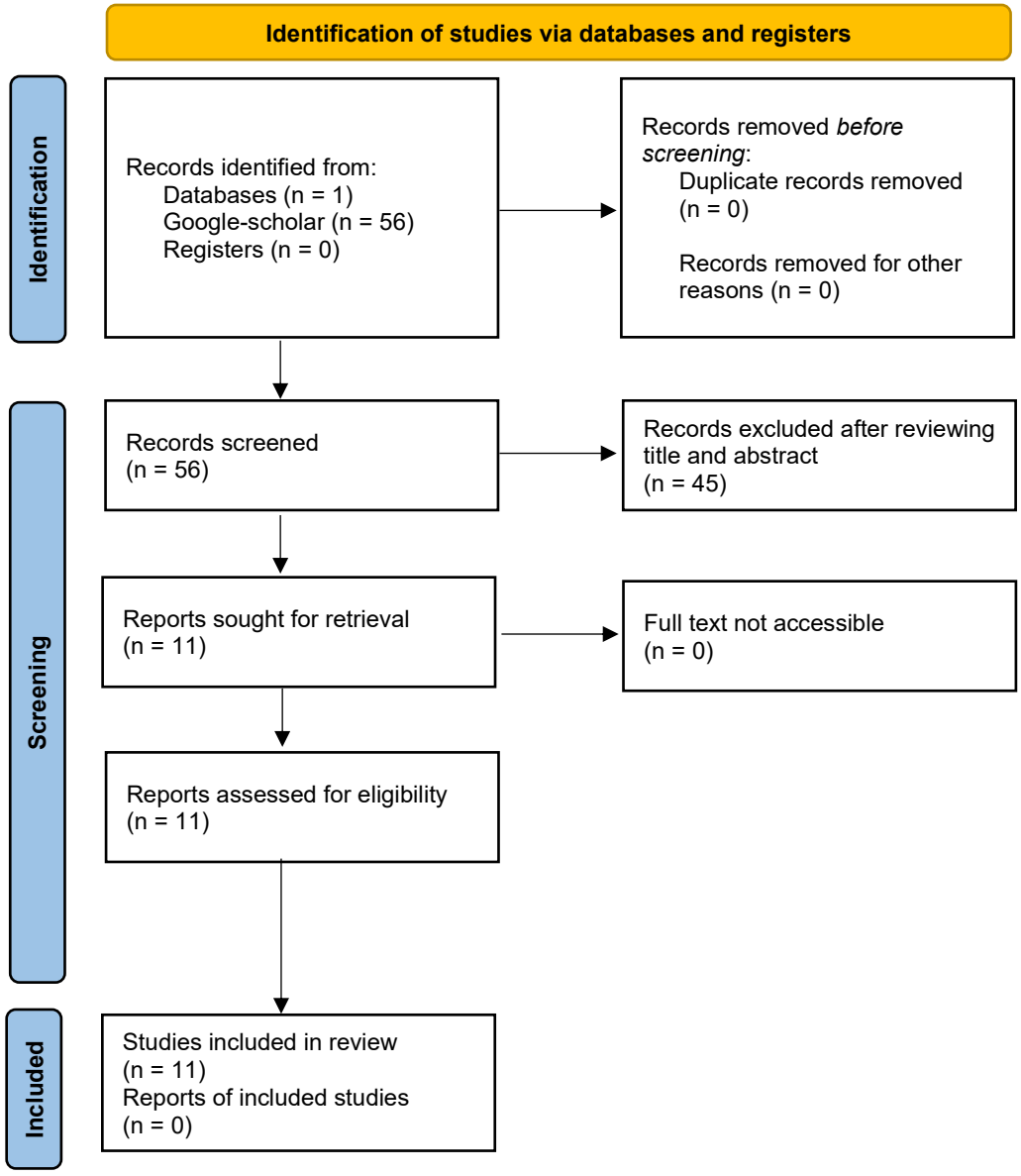

Figure 1. Flow Diagram showing screened studies.

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

Data Extraction

Data extraction focused on:

1. Study characteristics (authors, publication year, methodology).
2. Migration approaches and frameworks.
3. Event-driven architecture patterns employed.
4. Performance metrics and evaluation results.
5. Challenges and limitations identified.
6. Tools and techniques utilized.
7. Industry context and case study details.

RESULTS

The systematic search initially identified 56 potentially relevant papers from academic databases. Following the screening process, 11 studies met the inclusion criteria and were selected for detailed analysis. These studies encompassed various aspects of monolithic to microservices migration with particular emphasis on e-commerce applications and event-driven architecture patterns.

Table 1: Summary of Included Studies.

S/N	AUTHOR	TITLE	SUMMARY
1	Abgaz et al. (2023)	Decomposition of Monolith Applications Into Microservices Architectures: A Systematic Review	Abgaz et al. conduct a comprehensive systematic literature review examining 35 research papers on monolith to microservices decomposition. The study introduces the Monolith to Microservices Decomposition Framework (M2MDF) which identifies four major phases: analysis, decomposition, implementation, and evaluation. The research reveals that decomposition remains at an early stage with insufficient methods for combining static, dynamic, and evolutionary data. Key findings include the lack of standardized metrics and datasets, limited tool support, and the need for integrated comprehensive data collection methods. The framework provides a comprehensive map for researchers and practitioners to navigate the complex decomposition landscape.

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

2	Shabani et al. (2021)	Design of Modern Distributed Systems based on Microservices Architecture	Shabani et al. present a comparative study decomposing a monolithic .NET application into microservices using three different architectural patterns. The research employs Apache JMeter for performance testing and demonstrates the advantages and disadvantages of microservices versus monolithic architectures. The study addresses scalability challenges where monolithic systems require replication of entire applications, leading to inefficient resource utilization. The findings show that microservices enable independent scaling of components with different resource requirements, though they introduce complexity in network latency and security considerations. The research provides practical insights into deployment strategies and performance implications.
3	Asrowardi et al. (2020)	Designing microservice architectures for scalability and reliability in e-commerce	Asrowardi et al. focus specifically on e-commerce microservice architecture design using Design Science Research Methodology (DSRM). The study addresses e-commerce requirements for reusability services, automated deployments, and fast scalability. Performance evaluation includes accessibility, best practices, and search engine optimization testing. Results demonstrate that microservice architecture shows 30% better performance in scripting and 4% improvement in painting compared to monolithic architecture. The research provides a proof of concept specifically tailored for e-commerce platforms, addressing unique challenges in online retail environments including customer satisfaction and revenue optimization strategies.
4	Shafabakhsh et al. (2020)	Evaluating the Impact of Inter Process Communication in Microservice Architectures	Shafabakhsh et al. conduct an experimental study comparing synchronous (REST API, gRPC) and asynchronous (RabbitMQ) inter-process communication patterns in microservices. The research develops microservices for an e-commerce scenario simulating product page display

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

			functionality across five services: product information, reviews, recommendations, shipping, and shopping cart. Performance testing reveals that gRPC outperforms REST API and RabbitMQ under low load, but asynchronous patterns demonstrate superior performance and availability under high load conditions. The study shows RabbitMQ processing 4480 requests versus 4348 for gRPC under 200 concurrent users, with better fault tolerance characteristics.
5	Ghosh (2025)	Event-Driven Architectures for Microservices: A Framework for Scalable and Resilient Rearchitecting of Monolithic Systems	Ghosh presents a comprehensive framework for migrating monolithic systems to event-driven microservices architectures. The research explores foundational concepts including event sourcing, CQRS, saga patterns, and event collaboration. The framework outlines structured migration approaches including assessment, incremental decomposition strategies, event identification, and phased implementation. Case studies from financial services, e-commerce, and healthcare demonstrate practical applications with performance improvements and lessons learned. The study emphasizes the Strangler Fig Pattern as the most reliable approach, reporting 40-60% less downtime compared to alternative migration methods. Technical implementation covers event infrastructure selection, communication patterns, and observability requirements.
6	Kaloudis (2024)	Evolving Software Architectures from Monolithic Systems to Resilient Microservices: Best Practices, Challenges and Future Trends	Kaloudis examines the evolution of microservices architecture, emphasizing advantages in flexibility, scalability, and fault tolerance compared to monolithic and SOA approaches. The study explores implementation challenges including data consistency and API management, evaluating best practices such as Domain-Driven Design and the Saga Pattern. Case studies from leading companies like Netflix and Amazon demonstrate optimized resource utilization and operational adaptability. The research

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

			addresses cross-functional team organization (Uber) and transaction management (Airbnb). Future trends discussion includes Kubernetes orchestration and AIOps for improving system scalability and resilience. The study provides structured strategies for organizations transitioning from monolithic architectures with phased migration approaches to mitigate risks.
7	Taibi & Systä (2019)	From monolithic systems to microservices: A decomposition framework based on process mining	Taibi and Systä propose a novel 6-step framework utilizing process mining to reduce subjectivity in monolith decomposition. The framework analyzes independent execution traces using process mining tools applied to runtime log traces. Industrial validation compares proposed decomposition options with manual architect analysis, revealing that the framework identified issues and suitable decomposition options that manual analysis missed. The approach provides software architects with multiple decomposition options along with quality measures for evaluation and comparison. The framework demonstrates significant potential for improving decomposition quality by identifying different strategies and reducing subjective decision-making. The process mining approach offers objective, data-driven analysis of system behavior patterns.
8	Auer et al. (2021)	From monolithic systems to Microservices: An assessment framework	Auer et al. develop an evidence-based decision support framework for companies considering microservices migration. The research conducts interviews with experienced practitioners using Grounded Theory to identify characteristics and metrics for migration decisions. The assessment framework helps organizations evaluate whether to migrate based on objective system measures rather than intuition. The study addresses the economic impact of architectural changes and ensures migration decisions are based on solid information to achieve expected benefits. The framework supports

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

			companies in analyzing potential benefits and drawbacks of migration and re-architecting processes. Key contributions include structured approaches for evaluating migration readiness and reducing unsuccessful transformation risks.
9	Furda et al. (2017)	Migrating Enterprise Legacy Source Code to Microservices: On Multitenancy, Statefulness, and Data Consistency	Furda et al. elaborate on three critical challenges in microservice migration: multitenancy, statefulness, and data consistency. The research explains how to identify each challenge in legacy code and provides refactoring and architectural pattern-based migration techniques. Multitenancy enables microservices utilization by different organizations with distinctive requirements. Statefulness affects both availability and reliability of microservice systems. Data consistency challenges emerge when migrating from centralized to decentralized data repositories. The study explains interdependencies between these three factors and provides practical guidance for addressing enterprise legacy system modernization. The research focuses on incremental modernization techniques for cloud-computing environments.
10	Kalia et al. (2021)	Mono2Micro a practical and effective tool for decomposing monolithic Java applications to microservices	Kalia et al. present Mono2Micro, a practical tool for monolithic Java application decomposition using spatio-temporal analysis. The tool leverages business use cases and runtime call relations to create functionally cohesive class partitioning. Evaluation against four existing techniques demonstrates significant outperformance in domain-specific metrics. A survey of 21 industry practitioners reveals positive tool perception and benefits including functionally cohesive and explainable microservice decompositions. The tool performs automated analysis reducing subjective architectural decisions and providing objective service boundary identification. Results show Mono2Micro creates valuable aid for practitioners in

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

			generating high-quality decomposition solutions with measurable improvements over baseline approaches.
11	Aljawawdeh et al. (2023)	Smooth Change: Moving from Monolithic to Microservices with Flexible Methods	Aljawawdeh et al. present a comprehensive methodology for migrating legacy monolithic systems to microservices using adaptive synchronization techniques. The approach enhances software systems' scalability, maintainability, and flexibility while minimizing migration risks. Key recommendations include further exploration of adaptive synchronization for improved efficiency, expanding case studies across various domains for validation, integrating migration methodology with existing DevOps practices, and developing tooling automation to support the migration process. The research emphasizes flexible methods that streamline the transformation process and simplify transitions for development teams and organizations. The study contributes to reducing complexity and improving success rates of monolithic to microservices migrations.

FINDINGS

1. Decomposition Frameworks and Methodologies

The reviewed studies reveal several distinct approaches to monolith decomposition, each with specific advantages for e-commerce contexts.

a. **Domain-Driven Design (DDD) Approaches:** Multiple studies emphasize Domain-Driven Design as a foundational approach for identifying service boundaries in e-commerce systems. Abgaz et al. [1] identify DDD as a critical component of their Monolith to Microservices Decomposition Framework (M2MDF), noting its effectiveness in creating business-aligned service boundaries. Kaloudis [6] reinforces this approach, demonstrating how DDD enables better service cohesion and reduced coupling. In e-commerce contexts, natural domain boundaries often align with business capabilities, such as customer management, product catalog, order processing, payment handling, and inventory management.

b. **Process Mining-Based Decomposition:** Taibi and Systä [7] present a novel 6-step framework utilizing process mining to reduce subjectivity in decomposition decisions. Their approach analyzes runtime execution traces to identify independent service candidates, offering a data-driven alternative to purely architectural analysis. The

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

framework showed particular promise in industrial applications, helping to identify decomposition options that manual analysis missed. This objective approach addresses one of the key challenges identified by Abgaz et al. [1] regarding the lack of standardized decomposition methods.

c. **The M2MDF Framework:** Abgaz et al. [1] propose a comprehensive framework, identifying four major phases of decomposition:

   i. **Analysis Phase:** Understanding the existing monolith through static and dynamic analysis.

   ii. **Decomposition Phase:** Identifying service boundaries using various techniques.

   iii. **Implementation Phase:** Creating new microservices and establishing communication patterns.

   iv. **Evaluation Phase:** Assessing the quality of the decomposed architecture.

d. **Automated Decomposition Tools:** Kalia et al. [10] present Mono2Micro, a practical tool performing spatio-temporal decomposition using business use cases and runtime call relations. Their evaluation against four existing techniques demonstrated significant improvements in decomposition quality metrics, with practitioners rating the tool highly for creating functionally cohesive microservice partitions. This addresses the tool support gap identified by Abgaz et al. [1].

2. Event-Driven Architecture Patterns

Event-driven architecture patterns emerge as a critical enabler for successful microservices migration in e-commerce systems.

a. **Event Sourcing and CQRS:** Ghosh [5] emphasizes event sourcing as fundamental to event-driven microservices, storing all changes as a sequence of events. This approach provides complete auditability and state reconstruction capabilities particularly valuable in e-commerce contexts where transaction history and regulatory compliance are critical. The Command Query Responsibility Segregation (CQRS) pattern complements event sourcing by separating read and write operations, improving performance for high-volume e-commerce workloads.

b. **Saga Pattern for Distributed Transactions:** The saga pattern addresses one of the most challenging aspects of microservices migration: managing distributed transactions. Ghosh [5] describes how sagas coordinate local transactions through events, essential for e-commerce workflows spanning multiple services such as order processing, payment authorization, and inventory updates. Kaloudis [6] reinforces this pattern's importance, particularly referencing Airbnb's transaction management implementation.

c. **Event Collaboration Patterns:** Event collaboration enables services to work together without direct dependencies, crucial for e-commerce scalability. Services publish domain events (e.g., "OrderPlaced", "PaymentProcessed") that other services can subscribe to and react accordingly, creating loosely coupled systems that can evolve independently [5].

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

3. Performance Implications and Benefits

a. **Scalability Improvements:** Asrowardi et al. [3] demonstrate significant performance improvements through their microservices architecture for e-commerce, showing 30% better performance in scripting and 4% improvement in rendering compared to monolithic approaches. The study's performance testing using Design Science Research Methodology revealed enhanced accessibility, best practices compliance, and search engine optimization.

b. **Communication Pattern Performance:** Shafabakhsh et al. [4] provide crucial insights into inter-process communication (IPC) patterns in microservices through their e-commerce scenario testing. Their experimental evaluation reveals that:

   i. Synchronous patterns (REST, gRPC) perform better under low load conditions.

   ii. Asynchronous patterns (message queues) show superior performance under high load.

   iii. gRPC outperformed REST API by processing 43 more requests than REST under low load.

   iv. RabbitMQ (asynchronous) processed 4480 requests versus 4348 for gRPC under high load (200 users).

c. **Availability and Fault Tolerance:** The asynchronous nature of event-driven architectures provides superior availability characteristics. Shafabakhsh et al. [4] found that asynchronous systems maintained operation longer under stress, with RabbitMQ systems staying operational for approximately 7 minutes under 200 concurrent users compared to 5 minutes for gRPC and 4.5 minutes for REST API.

4. Migration Strategies and Patterns

a. **Strangler Fig Pattern:** Ghosh [5] identifies the Strangler Fig Pattern as the most reliable approach for complex systems requiring high availability. This incremental migration strategy gradually replaces monolith components with microservices while maintaining the original system until migration completion. The research shows this pattern typically requires 40-60% less downtime compared to alternative approaches.

b. **Incremental Decomposition Strategies:** Aljawawdeh et al. [11] propose adaptive synchronization techniques for smooth migration, emphasizing flexible methods that minimize risks. Their approach includes:

   i. Anti-corruption layers preventing legacy concept leakage.

   ii. Branch by abstraction for simultaneous development.

   iii. Parallel run strategies for validation before cutover.

c. **Assessment-Driven Migration:** Auer et al. [8] present an evidence-based decision support framework helping organizations determine whether to migrate to microservices. Their interview-based study with industry practitioners identifies critical characteristics

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

and metrics for informed migration decisions, reducing the risk of unsuccessful transformations.

5. Tools and Techniques

a. **Automated Decomposition Support:** The reviewed studies reveal limited but growing tool support for automated decomposition:

b. **Mono2Micro Tool:** Kalia et al. [10] present a comprehensive tool leveraging business use cases and runtime analysis. The tool significantly outperformed baseline techniques and received positive practitioner feedback for creating explainable decompositions.

c. **Process Mining Tools:** Taibi and Systä [7] demonstrate the application of process mining tools to log traces for identifying service boundaries, providing objective data for decomposition decisions.

d. **Assessment Frameworks:** Auer et al. [8] provide structured approaches for evaluating migration readiness, though these remain largely manual processes requiring expert judgment.

6. Challenges and Limitations

a. **Data Consistency Management:** Furda et al. [9] identify data consistency as a primary challenge when migrating from centralized to decentralized data repositories. E-commerce systems particularly struggle with maintaining ACID properties across service boundaries while preserving business invariants. The study elaborates on the interdependencies between multitenancy, statefulness, and data consistency in enterprise migration scenarios.

b. **Organizational Alignment:** Kaloudis [6] emphasizes organizational challenges, noting that successful microservices adoption requires cross-functional teams and cultural transformation beyond technical changes. The study highlights examples from Netflix and Amazon, demonstrating the need for organizational restructuring to support microservices architectures.

c. **Technical Complexity:** Shabani et al. [2] identify several technical challenges:

   i. Increased infrastructure complexity.

   ii. Network latency and security considerations.

   iii. Service discovery and communication management.

   iv. Monitoring and debugging distributed systems.

d. **Standardization Gaps:** Abgaz et al. [1] note the absence of standardized metrics, datasets, and baselines for evaluating decomposition quality, making it difficult to compare different approaches or validate migration success. This represents a significant barrier to widespread adoption and objective evaluation of migration initiatives.

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

7. Industry Case Studies and Lessons Learned

a. **Financial Services Migration:** Ghosh [5] presents a case study of a financial services company migrating a 15-year-old core banking system. Using the Strangler Fig Pattern with Apache Kafka as the event backbone, they achieved:

   i. Release cycle reduction from months to weeks.

   ii. 5x improvement in peak transaction handling.

   iii. Independent scaling capabilities for high-demand services.

b. **E-commerce Platform Modernization:** The same study [5] describes an e-commerce platform's migration approach, starting with product catalog extraction and implementing RabbitMQ for messaging before transitioning to Kafka. Results included:

   i. 99.99% availability during peak shopping events.

   ii. 30% infrastructure cost reduction.

   iii. Multiple daily deployments versus bi-weekly releases.

c. **Healthcare Provider Transformation:** A healthcare provider's cautious migration approach focused on non-critical systems first, implementing comprehensive compliance monitoring and gradual expansion to core services, demonstrating the importance of domain-specific considerations [5].

DISCUSSION

1. Key Success Factors

The synthesis of findings reveals several critical success factors for migrating monolithic e-commerce systems to event-driven microservices:

a. **Domain-Driven Decomposition:** The most successful migrations employ Domain-Driven Design principles to identify natural service boundaries aligned with business capabilities [1][6]. E-commerce systems benefit from well-defined domain contexts, such as customer management, product catalog, and order processing.

b. **Incremental Migration Strategies:** The Strangler Fig Pattern emerges as the preferred approach for complex e-commerce systems, providing risk mitigation and continuous operation during transformation [5]. This incremental approach allows organizations to learn and adapt migration strategies based on early experiences.

c. **Event-Driven Communication:** Asynchronous event-driven patterns prove superior for e-commerce workloads, providing better scalability under high load conditions and improved fault tolerance [4][5]. The natural event-oriented nature of e-commerce business processes aligns well with event-driven architectures.

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

d. **Comprehensive Tooling:** Organizations benefit significantly from automated decomposition tools like Mono2Micro [10], which provide objective analysis of service boundaries and reduce subjective architectural decisions.

2. Performance Considerations

The performance analysis reveals nuanced trade-offs between synchronous and asynchronous communication patterns:

a. **Low Load Scenarios:** Synchronous patterns (gRPC, REST) provide better performance with lower latency [4].

b. **High Load Scenarios:** Asynchronous patterns (message queues) demonstrate superior throughput and availability [4].

c. **Availability:** Event-driven architectures provide better fault tolerance and system availability under stress [4].

These findings suggest that e-commerce systems should employ hybrid approaches, using synchronous communication for real-time user interactions and asynchronous patterns for backend processing and inter-service communication.

3. Organizational Implications

The migration to microservices requires significant organizational transformation beyond technical changes:

a. **Team Structure:** Successful implementations require cross-functional teams aligned with service boundaries, often necessitating organizational restructuring [6].

b. **Operational Capabilities:** Organizations must develop new capabilities in distributed systems monitoring, deployment automation, and incident management [2].

c. **Cultural Change:** The shift from monolithic to microservices architectures requires cultural adaptation to embrace autonomous teams, continuous deployment, and distributed system thinking [11].

4. Gaps and Future Research Directions

The review identifies several critical gaps requiring future research:

a. **Standardized Evaluation Metrics:** The absence of standardized metrics for evaluating migration success hampers objective comparison of approaches and validation of benefits [1].

b. **E-commerce Specific Patterns:** While general microservices patterns exist, more research is needed on patterns specific to e-commerce workloads and business requirements [3].

c. **Automated Tool Development:** Current automated decomposition tools show promise but require further development to handle complex e-commerce domains with their intricate business rules and data relationships [10].

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

d. **Long-term Maintenance:** Limited research exists on the long-term maintenance and evolution of microservices architectures in e-commerce contexts.

LIMITATIONS

This systematic review has several limitations that should be considered when interpreting results:

1. **Limited Scope:** The focus on e-commerce systems may limit generalizability to other domains, though many findings likely apply broadly to transactional systems.
2. **Publication Bias:** The review may suffer from publication bias, as unsuccessful migration attempts are less likely to be reported in academic literature.
3. **Temporal Limitations:** The rapidly evolving nature of microservices technologies means that some findings may become outdated as new tools and techniques emerge.
4. **Tool Availability:** Some studies reference proprietary tools or techniques that may not be widely available for replication or validation.

CONCLUSION

This systematic review provides comprehensive insights into the current state of research on migrating monolithic e-commerce systems to microservices using event-driven architecture approaches. The findings demonstrate that while microservices migration offers significant benefits in terms of scalability, fault tolerance, and development agility, successful implementation requires careful consideration of decomposition strategies, communication patterns, and organizational factors.

Event-driven architectures emerge as particularly well-suited for e-commerce contexts, providing natural alignment with business processes and superior performance characteristics under high load conditions [4][5]. The Strangler Fig Pattern, combined with Domain-Driven Design principles, represents the most reliable approach for complex e-commerce migrations [1][5][6].

However, significant challenges remain, including data consistency management [9], increased operational complexity [2], and the need for organizational transformation [6][11]. The absence of standardized evaluation metrics [1] and limited automated tooling support [10] continue to complicate migration decisions for practitioners.

Key Recommendations for Practitioners:

1. **Adopt Incremental Approaches:** Use the Strangler Fig Pattern for risk mitigation and continuous operation during migration [5].
2. **Leverage Domain-Driven Design:** Align service boundaries with business domains for better maintainability and team autonomy [1][6].

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

3. **Implement Event-Driven Patterns:** Utilize asynchronous communication for better scalability and fault tolerance [4][5].
4. **Invest in Tooling:** Leverage automated decomposition tools where available and invest in observability infrastructure [10].
5. **Plan Organizational Change:** Address team structure, operational capabilities, and cultural transformation alongside technical migration [6][11].
6. **Measure and Monitor:** Establish clear metrics for evaluating migration success and system performance [8].

FUTURE RESEARCH DIRECTIONS:

The field would benefit from further research in several areas:

1. Development of standardized metrics for evaluating microservices migration success [1].
2. Creation of e-commerce-specific migration patterns and best practices [3].
3. Advanced automated decomposition tools handling complex business domains [10].
4. Long-term studies on microservices maintenance and evolution in e-commerce contexts.
5. Investigation of emerging technologies like serverless computing and edge computing in microservices contexts.

This review contributes to the growing body of knowledge on microservices migration and provides evidence-based guidance for organizations undertaking e-commerce modernization initiatives. As the field continues to evolve, ongoing research and industry collaboration will be essential for developing more effective migration strategies and supporting tools.

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO

REFERENCES

[1] Abgaz, Y., McCarren, A., Elger, P., Solan, D., Lapuz, N., Bivol, M., ... & Clarke, P. (2023). Decomposition of monolith applications into microservices architectures: A systematic review. *IEEE Transactions on Software Engineering*, 49(8), 4213-4242.

[2] Shabani, I., Mëziu, E., Berisha, B., & Biba, T. (2021). Design of modern distributed systems based on microservices architecture. *International Journal of Advanced Computer Science and Applications*, 12(2).

[3] Asrowardi, I., Putra, S. D., & Subyantoro, E. (2020, February). Designing microservice architectures for scalability and reliability in e-commerce. In *Journal of Physics: Conference Series* (Vol. 1450, No. 1, p. 012077). IOP Publishing.

[4] Shafabakhsh, B., Lagerström, R., & Hacks, S. (2020, December). Evaluating the Impact of Inter Process Communication in Microservice Architectures. In *QuASoQ@ APSEC* (pp. 55-63).

[5] Ghosh, A. (2025). Event-Driven Architectures for Microservices: A Framework for Scalable and Resilient Rearchitecting of Monolithic Systems. *International Journal on Science and Technology (IJSAT)*.

[6] Kaloudis, M. (2024). Evolving Software Architectures from Monolithic Systems to Resilient Microservices: Best Practices, Challenges and Future Trends. *International Journal of Advanced Computer Science & Applications*, 15(9).

[7] Taibi, D., & Systä, K. (2019). From monolithic systems to microservices: A decomposition framework based on process mining. In the *International Conference on Cloud Computing and Services Science* (pp. 153-164). SciTePress.

[8] Auer, F., Lenarduzzi, V., Felderer, M., & Taibi, D. (2021). From monolithic systems to Microservices: An assessment framework. *Information and Software Technology*, 137, 106600.

[9] Furda, A., Fidge, C., Zimmermann, O., Kelly, W., & Barros, A. (2017). Migrating enterprise legacy source code to microservices: on multitenancy, statefulness, and data consistency. *IEEE Software*, 35(3), 63-72.

[10] Kalia, A. K., Xiao, J., Krishna, R., Sinha, S., Vukovic, M., & Banerjee, D. (2021, August). Mono2micro: a practical and effective tool for decomposing monolithic java applications to microservices. In *Proceedings of the 29th ACM joint meeting on European software engineering conference and symposium on the foundations of software engineering* (pp. 1214-1224).

[11] Aljawawdeh, H., Abuezhayeh, S., Qaddoumi, E., & Maghrabi, L. (2023). Smooth Change: Moving from Monolithic to Microservices with Flexible Methods. In *Artificial Intelligence, Internet of Things, and Society 5.0* (pp. 559-572). Springer Nature Switzerland.

Article DOI: 10.52589/AJSTE-O1G0V4GO
DOI URL: https://doi.org/10.52589/AJSTE-O1G0V4GO