Journal of Information Systems Engineering and Management
2025, 10(59s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Scalable Microservices Architectures for High-Traffic Event Ticketing Platforms

Vinay Chowdary Duvvada
California State University, East Bay

ARTICLE INFO

Received: 10 July 2025

Revised: 15 Aug 2025

Accepted: 22 Aug 2025

ABSTRACT

Event ticketing platforms face extraordinary technical hurdles when massive audiences concurrently pursue access to coveted event admissions. Conventional unified application structures typically falter under such concentrated request volumes, causing operational slowdowns, uncompleted transactions, and compromised user satisfaction. Distributed component architecture presents an advantageous alternative, separating ticketing functions into autonomous, independently connected service modules capable of individualized capacity adjustments according to particular operational demands. This structural framework delivers enhanced stability through compartmentalized failure zones, superior responsiveness through calibrated resource distribution, and greater development flexibility through modular independence. Essential structural elements facilitating effective implementations encompass centralized interface management for streamlined connectivity, dispersed temporary storage for inventory supervision, notification-based component interaction for functional coordination, and protective interruption mechanisms for disruption limitation. Deployment methodologies prioritize capability-oriented service delineation, specialized storage technologies for optimized data management, and comprehensive visibility systems for anticipatory performance evaluation. The combination of these architectural elements produces an adaptable, durable ticketing infrastructure capable of sustaining operational effectiveness during extraordinary usage surges while remaining receptive to changing functional requirements. These architectural fundamentals yield significant advantages across various operational dimensions, from localized regional services to international ticketing networks accommodating concurrent interactions from countless simultaneous participants.

Keywords: Microservices, Scalability, Event Ticketing, Distributed Systems, Traffic Management

1. Introduction

Event ticketing platforms face distinct technical challenges during high-demand sales periods when thousands of customers simultaneously attempt to purchase limited-availability tickets. This "on-sale" scenario creates infrastructure demands that surpass normal operational parameters [1]. When these platforms cannot maintain service stability during traffic surges, substantial business consequences follow—lost revenue opportunities, diminished brand reputation, and reduced customer satisfaction. Recent high-profile ticket releases have illustrated that established providers encounter significant difficulties maintaining service continuity during extraordinary demand periods. Monolithic ticketing systems exhibit inherent architectural constraints that contribute to service disruptions. These platforms operate as unified, tightly integrated applications where all components scale together regardless of individual resource requirements. As system complexity grows and traffic patterns become increasingly variable, this architectural approach proves progressively inefficient. When any component reaches capacity thresholds, the entire application becomes susceptible to performance degradation or complete failure [2].

Journal of Information Systems Engineering and Management
2025, 10(59s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The architectural paradigm of distributed, service-oriented design presents a substantial advancement beyond traditional structural constraints. This methodology separates functional units into discrete, autonomous components that interact via standardized communication protocols. Individual service modules encapsulate distinct operational domains—reservation management, financial transactions, credential verification—with each functioning as a self-contained entity. Core architectural principles encompass component independence, business-aligned boundaries, and distributed information governance. Through functional isolation within dedicated service boundaries, this structural approach facilitates targeted resource allocation for specific system elements based on their particular operational requirements and demand patterns. Architecture enables precise scaling of individual components according to their specific resource demands.

The benefits of microservices for ticketing platforms extend beyond scalability improvements. Advantages include enhanced fault isolation, increased technological flexibility, more efficient deployment processes, and improved team organization around business capabilities rather than technical specializations. Prominent event access management solutions have increasingly implemented distributed service models to resolve the distinctive capacity challenges characteristic of ticketing functionalities. The subsequent sections examine structural elements enabling expanded capacity, strategies for accommodating substantial request volume variations, and practices for ensuring uninterrupted service during concentrated usage intervals. The content advances systematically from essential operational specifications through architectural principles, deployment methodologies, and maintenance frameworks. This organized progression creates a thorough blueprint for developing reliable transaction platforms that preserve functional stability during periods of exceptional customer demand.

Aspect	Monolithic Architecture	Microservices Architecture
System Structure	Encompasses all functionalities within a single, unified application package	Consists of multiple independent, specialized services that communicate via APIs
Development Process	Features a tightly integrated codebase where modifications can introduce widespread risks	Maintains separate codebases for each service, enabling isolated updates and accelerated development cycles
Complexity Management	Can evolve into unwieldy systems that become increasingly difficult to understand and maintain	Distributes complexity across smaller, more focused components with clear boundaries
Failure Resilience	Vulnerable to complete system outages when critical components fail	Contains failures within individual services, preserving overall system availability
Scaling Capabilities	Primarily scales through vertical expansion (adding resources to existing instances)	Offers flexible scaling options, allowing specific services to scale independently based on demand
Team Organization	Typically structured around technical specializations (frontend, backend, database teams)	Organized around business domains with cross-functional teams owning specific services
Deployment Patterns	Generally follows infrequent release cycles due to comprehensive testing requirements	Enables continuous deployment practices with smaller, lower-risk releases
Technology Diversity	Restricted to a single technology stack across the entire application	Supports polyglot implementation, allowing teams to select optimal technologies for specific services

Table 1: Comparing Monolithic and Microservices Architectures [2,8]

Journal of Information Systems Engineering and Management
2025, 10(59s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

2. Essential Technical Foundations for Contemporary Ticketing Systems

Modern event access management platforms require robust technical underpinnings to ensure uninterrupted functionality during periods of exceptional customer activity. The complex infrastructure supporting these systems must adapt fluidly to handle transaction quantities that vary significantly between standard operational levels and concentrated sales intervals. Real-time inventory management constitutes a fundamental requirement, presenting substantial concurrent access challenges. When numerous users simultaneously attempt to purchase limited tickets, systems must prevent duplicate sales while maintaining responsive performance [3]. This necessitates advanced inventory locking mechanisms that temporarily reserve seats during the purchase process without creating system bottlenecks. These mechanisms require precise calibration of reservation timeouts—intervals too short frustrate legitimate customers, while intervals too long unnecessarily restrict available inventory. Seat allocation introduces additional complexity, particularly for venues with intricate seating configurations or differentiated pricing structures. Systems must implement sophisticated business rules governing seat selection while processing thousands of concurrent requests. The temporary hold mechanism must integrate seamlessly with payment processing workflows, ensuring that reservations automatically release if transactions remain incomplete within defined timeframes. Payment processing represents another critical requirement, demanding both transaction security and processing efficiency. Ticketing platforms must implement robust security measures while completing transactions within specific time windows to prevent inventory hoarding. This balance requires careful system design that maintains transactional integrity across distributed service boundaries [4]. The challenges of distributed transactions become particularly acute during peak traffic periods, when consistency and availability trade-offs require meticulous management. Ticketing platforms demand faster response capabilities than traditional online retail environments due to the urgency inherent in limited-availability event sales. Customers anticipate instantaneous system feedback throughout their purchasing journey, despite thousands simultaneously engaging with the platform.

During peak sales periods, advanced queue orchestration becomes essential to balance fair distribution with operational integrity. Strategic implementations typically utilize structured waiting environments, sophisticated access sequencing determined by established parameters, and continuous communication of estimated processing timeframes. Transparent queue status information proves particularly valuable during prolonged waiting intervals, substantially improving customer satisfaction metrics despite inevitable delays. The varied engagement approaches of present-day ticket buyers create supplementary technical demands. Contemporary platforms must deliver consistent operational capabilities across mobile devices and traditional computers while accommodating fluctuating connectivity conditions and hardware variations. Delivering uniform experiences regardless of access method necessitates precise interface architecture and adaptable implementation techniques. Effective systems deploy multi-layered protections against automated purchasing applications, advanced transaction analysis to detect irregular activities, and supplementary verification procedures to safeguard customer accounts. Optimal security frameworks achieve equilibrium between robust protective measures and streamlined customer interactions, minimizing authentication requirements that could discourage valid transactions. These defensive mechanisms must expand in proportion to traffic intensity while adapting to counteract new circumvention methods, consistently maintaining critical performance metrics during vital sales windows.

3. Decomposing Ticketing Systems into Microservices

Converting a monolithic ticketing infrastructure into distributed service components requires methodical division according to functional capabilities and logical business boundaries. Domain-focused architectural approaches provide the structural framework for determining appropriate service delineations within sophisticated ticketing environments. This methodology prioritizes

Journal of Information Systems Engineering and Management
2025, 10(59s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

comprehension of fundamental business domains and their interrelationships, enabling technical architects to establish component boundaries that correspond with natural business divisions [4]. By emphasizing business models rather than technical specifications, development teams establish services with strong internal cohesion and minimal external dependencies, supporting independent enhancement and deployment cycles. Effective domain segmentation for ticketing platforms typically reveals several distinct conceptual areas. Each operational context establishes a functional boundary within which specific business logic applies, with clearly defined interfaces governing interactions with adjacent contexts. The event context encompasses venue configurations, performance details, and seating arrangements, while customer management handles patron profiles and preferences. Identity verification functions operate within the customer domain but maintain concentrated responsibility for credential validation and permission management. This separation enables specialized security implementations without affecting broader customer management capabilities.

Purchase processing typically spans multiple operational domains, including order administration and financial transactions. The order domain maintains responsibility for selection management, fulfillment tracking, and confirmation delivery. Meanwhile, the payment domain handles secure processing of monetary transactions, frequently integrating external financial processors while shielding these complexities from associated services. This division allows dedicated teams to address specialized requirements of financial processing without impacting related order management functionality.

Information management components maintain event particulars, marketing materials, and location details. These services deliver consistent content across customer interfaces while allowing administrative personnel to modify information without affecting transactional elements. Inventory administration represents a critical component in ticketing platforms, maintaining current seat availability and processing reservation requests. This service must handle extreme concurrency during popular events while preserving data integrity to prevent allocation conflicts [5]. Communication services manage customer interactions across multiple channels, including confirmation messages, alert notifications, and application updates. These components process events from other domains and transform them into appropriate customer communications, maintaining message templates and delivery preferences while ensuring coherent messaging. This pattern enables other services to concentrate on core functionality without managing communication complexities. Information storage presents particular challenges in distributed service architectures. Dedicated database implementations provide data isolation, allowing individual services to maintain control over persistence mechanisms and structural evolution. This approach enables diversified storage strategies, where each service selects optimal technologies based on specific requirements. For example, inventory services might utilize memory-optimized databases designed for high-volume concurrent operations, while content services might implement document storage better suited for flexible content structures. Preserving data consistency across service boundaries presents significant challenges, particularly for processes spanning multiple components. Notification-driven structural designs resolve these challenges through propagating condition modifications as informational signals, enabling functional components to uphold ultimately harmonized representations of interconnected data elements. Non-centralized interaction frameworks, where individual modules react to occurrence messages without unified control mechanisms, deliver expandable methodologies for administering sophisticated procedural sequences while maintaining operational independence. In scenarios requiring heightened consistency assurances, approaches incorporating countervailing procedural arrangements implement reversible functional actions to preserve commercial validity without sacrificing component self-governance.

Journal of Information Systems Engineering and Management
2025, 10(59s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Scalability Challenges	Strategic Solutions
Performance Degradation	Implement a microservices architecture to allow independent scaling of system components
Traffic Fluctuations	Deploy cloud infrastructure with auto-scaling capabilities to adjust resources based on real-time demand
Database Bottlenecks	Utilize horizontal scaling (sharding) and read replicas to distribute database load effectively
Request Overload	Implement load balancing strategies to distribute traffic evenly across multiple servers
System Downtime	Design for resilience with redundant components and failure isolation patterns
Inconsistent User Experience	Leverage caching systems like Redis for frequently accessed data to maintain performance
Security Vulnerabilities	Scale security measures proportionally with encrypted data transmission and storage

Table 2: Challenges and Solutions in High-Traffic Systems [2,7,8]

4. Foundational Design Elements for Service Expansion and Reliability

Effective event access platforms necessitate purpose-built organizational frameworks that facilitate demand accommodation while maintaining continuous availability. These architectural configurations establish the essential foundation upon which dependable ticketing services operate during varying transaction volumes. The unified interface layer functions as a fundamental element, establishing a consolidated entry point for client applications while concealing the intricacies of underlying service interactions [5]. This centralized access mechanism handles cross-functional concerns, including identity verification, request distribution, and response consolidation, reducing client complexity while supporting backend evolution without requiring client modifications. Contemporary interface implementations incorporate advanced traffic governance capabilities, including request limitation and throughput control, which shield backend functions from overwhelming volumes during concentrated demand intervals.

The device-optimized backend pattern extends interface functionality by establishing purpose-specific consolidation layers tailored for particular client categories. This methodology recognizes the varied requirements across different interaction platforms, delivering customized interfaces that reduce network requirements for mobile devices while supporting enhanced capabilities for browser applications. These specialized interfaces transform and combine information from multiple sources, minimizing client-side processing requirements and enhancing responsiveness across diverse connectivity environments.

Distributed temporary storage represents another vital pattern for ticketing infrastructure, particularly for commonly accessed, relatively unchanging information. Strategic temporary retention implementation at various system tiers substantially decreases permanent storage demands while improving response efficiency for routine operations. Global content distribution positions static elements geographically nearer to end users, while application-level retention maintains frequently requested business information such as event details and pricing data. The most critical retention implementations address inventory information, where specialized memory-resident data structures monitor seat availability with minimal processing delay. These retention strategies must manage challenging invalidation scenarios to prevent outdated information while preserving performance under high concurrent access [6].

Indirect communication patterns enable loose infrastructure elements to buffer requests during volume surges, preventing slower components from creating system constraints while supporting independent capacity adjustment between message creators and consumers. Component coordination

Journal of Information Systems Engineering and Management
2025, 10(59s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

through these indirect channels follows either distributed or centralized patterns. Distributed control divides responsibility among participating components, with each responding to relevant notifications without centralized management. This approach improves scalability but increases complexity when monitoring overall process status. Centralized control introduces dedicated elements for managing complex procedural sequences, enhancing visibility, potentially at the expense of introducing scaling limitations.

Complete historical recording and responsibility-divided access patterns address information retrieval challenges in high-volume scenarios. Historical state recording maintains comprehensive transaction logs rather than merely current conditions, supporting detailed auditing capabilities and straightforward recovery from failures. Separated read-write responsibility divides retrieval and modification operations, allowing specialized optimizations for each access pattern. These complementary methodologies enable retrieval-intensive operations to expand independently from modification operations, a particular advantage for ticketing systems where information requests typically exceed updates by considerable margins. Protective patterns maintain system integrity during partial malfunctions. Automatic circuit protection prevents cascading failures by monitoring component health and temporarily suspending requests to degraded functions. Compartmentalization patterns isolate essential system elements to contain failures, preventing resource depletion in one area from impacting unrelated functionality. Functional prioritization strategies preserve core capabilities during partial disruptions by deactivating secondary features, ensuring that critical purchasing processes remain operational despite supporting service disturbances. Comprehensive operational monitoring across all components enables automated recovery mechanisms to identify and resolve issues before affecting customer interactions, maintaining system dependability during extended high-traffic intervals.

5. Handling Extreme Traffic Spikes

Ticketing platforms experience demand variations unlike most digital services, with activity often multiplying exponentially within moments of significant event announcements. Managing these extraordinary traffic fluctuations requires specialized capacity strategies customized for different component types [6]. Independent components benefit from horizontal expansion approaches, introducing identical instances to distribute activity across multiple resources. Dependent components like inventory management typically employ vertical enhancement for primary instances while distributing information retrieval horizontally, preserving data consistency while improving processing capacity. Advanced platforms implement both anticipatory and responsive capacity mechanisms. Anticipatory approaches evaluate historical patterns and event characteristics to forecast demand levels, allocating resources before traffic materializes. These proactive measures complement responsive adjustments triggered by operational metrics, which modify capacity according to actual utilization. Container management platforms facilitate these dynamic capacity operations, automating deployment and adjustment decisions while optimizing resource allocation across variable workloads.

Thorough capacity validation confirms system behavior under extreme conditions before encountering actual customer traffic. Artificial load generation replicates authentic user behavior at scale, identifying performance constraints and system limitations before production implementation. These evaluation methodologies establish performance references while confirming capacity thresholds, ensuring appropriate resource planning for anticipated demand levels [7]. Precise capacity forecasting combines historical utilization information with event-specific factors, enabling accurate resource allocation that balances performance requirements against operational expenditures. Structured entry management provides controlled access mechanisms during periods of extraordinary demand, preventing system overload while delivering transparent user experiences. These specialized request management systems organize users in sequential order when demand exceeds available processing capacity, progressively admitting them to the transaction process as resources become available. Advanced implementations incorporate equitable distribution mechanisms to prevent

Journal of Information Systems Engineering and Management
2025, 10(59s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

technical advantages from compromising fair access, while providing continuous status updates that sustain user engagement during waiting intervals.

Regional distribution strategies address both performance and reliability concerns for international events. Geographic request routing directs users to the nearest available infrastructure, reducing response delays while improving interaction quality. Multi-location deployments duplicate critical services across geographically distributed processing centers, providing continuity capabilities while distributing activity across multiple regions. These approaches prevent localized capacity constraints from affecting global availability, maintaining consistent performance regardless of user location.

Boundary processing extends this distributed architecture by positioning selected functionality closer to end users. By deploying services to peripheral locations, platforms can perform certain operations with minimal delay while reducing demands on centralized infrastructure. Common boundary implementations include request validation, initial queue positioning, and content delivery, enhancing perceived performance while maintaining centralized control of critical inventory and transaction processing functions.

Architectural Component	Implementation Strategy
Microservices Design	Break monolithic applications into independent, function-specific services that can scale individually
Cloud Infrastructure	Utilize cloud platforms that provide inherent scalability options with global distribution capabilities
Database Architecture	Implement horizontal scaling through sharding and vertical scaling through resource upgrades
Load Distribution	Deploy advanced load balancing with round-robin, least connections, or weighted distribution methods
Caching Framework	Implement multi-level caching strategies for data, sessions, and static content
Monitoring Systems	Establish comprehensive performance monitoring to identify bottlenecks before they impact users
Auto-scaling Mechanisms	Configure dynamic resource allocation based on predefined metrics and traffic patterns
Security Infrastructure	Implement scalable security measures, including DDoS protection and role-based access control

Table 3: Key Components of Scalable Software Architecture [2,8]

6. Observability and Operational Excellence

Maintaining comprehensive visibility across the distributed ticketing infrastructure constitutes an essential foundation for service stability during concentrated sales periods. Complete transaction pathway monitoring enables technical personnel to trace customer interactions as they navigate multiple functional components, revealing performance constraints and inter-service dependencies affecting purchase completions [7]. This comprehensive observability necessitates consistent transaction identifiers maintained throughout service boundaries, providing contextual elements for resolving intricate interaction patterns. Operational measurement systems consolidate critical performance indicators across distributed services, establishing normative behavioral parameters while facilitating prompt identification of operational deviations. Unified logging repositories integrate diagnostic information from diverse sources, offering consolidated access to relevant technical details during service interruptions. Sophisticated algorithmic evaluation identifies

Journal of Information Systems Engineering and Management
2025, 10(59s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

unexpected system behaviors before they manifest as customer-facing disruptions, initiating automated countermeasures or notifying technical teams about potential concerns.

Deliberate resilience validation through structured fault introduction confirms system durability under compromised conditions. These orchestrated service disturbances confirm that defensive mechanisms function appropriately when service modules encounter performance reduction or total failure. Autonomous recovery systems identify functional irregularities and implement established remediation sequences independently, significantly minimizing interruption timeframes and user consequences [8]. A comprehensive assessment following service irregularities uncovers underlying origins and structural enhancements, converting operational challenges into platform refinements. Dynamic information displays deliver performance indicators to technical personnel, facilitating prompt awareness of developing conditions. Proactive resource planning forecasts infrastructure requirements derived from established usage trajectories, harmonizing performance necessities with operational costs to establish optimal resource efficiency.

Service Domain	Primary Responsibilities
User Management	Customer profile administration, preference tracking, and authentication verification
Event Catalog	Performance details, venue information, and promotional content management
Inventory Control	Real-time seat availability tracking, temporary reservation handling, and allocation rules
Order Processing	Shopping cart management, purchase flow coordination, and order status tracking
Payment Handling	Transaction processing, financial gateway integration, and payment security
Notification System	Communication delivery across channels, including email, mobile, and in-app alerts
Analytics Platform	Performance monitoring, customer behavior analysis, and operational reporting

Table 4: Service Boundaries in Microservices Ticketing Architecture [4,8]

Conclusion

The architectural evolution from unified to distributed component frameworks constitutes a substantial advancement for ticketing infrastructures confronting high-intensity transaction scenarios. Dividing ticketing operations into separate, autonomously expandable service units enables exact capacity allocation, superior disruption containment, and heightened development flexibility. Effective structural implementations carefully balance numerous critical factors, including component granularity, information consistency requirements, and management complexity. More compact platforms potentially benefit from blended approaches, maintaining selected unified elements while strategically deploying distributed components for capacity-critical functions, including inventory supervision and access sequencing. Expansive platforms derive benefits from comprehensive distributed component adoption, though this methodology necessitates sophisticated coordination and surveillance capabilities. Ticketing infrastructure continues progressing alongside nascent technical innovations. Function-oriented computing without persistent infrastructure offers possibilities for remarkable adaptability during demand fluctuations without continuous configuration management. Distributed edge processing creates opportunities to position certain ticketing operations nearer to end participants, diminishing response delays and strengthening

Research Article

regional service resilience. Concurrently, progressive temporary storage technologies and purpose-built information management systems for inventory supervision promise additional performance enhancements. The principles of segmented, independently expandable components remain essential regardless of specific technological implementations. Through the adoption of these architectural methodologies, ticketing platforms can provide consistent operational effectiveness and dependability even during the most intensive transaction periods, ultimately elevating customer contentment and operational achievements throughout the ticketing environment.

References

[1] Alam Rahmatulloh et al., "Event-Driven Architecture to Improve Performance and Scalability in Microservices-Based Systems," in 2022 International Conference on Advancement in Data Science, E-learning and Information Systems (ICADEIS), ResearchGate, Nov. 2022. https://www.researchgate.net/publication/368431218_Event-Driven_Architecture_to_Improve_Performance_and_Scalability_in_Microservices-Based_Systems
[2] Mastura Diana Marieska et al., "Performance Comparison of Monolithic and Microservices Architectures in Handling High-Volume Transactions," Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi), ResearchGate, Jun. 2025. https://www.researchgate.net/publication/392834604_Performance_Comparison_of_Monolithic_and_Microservices_Architectures_in_Handling_High-Volume_Transactions
[3] Biman Barua and M. Shamim Kaise, "Cloud-Enabled Microservices Architecture for Next-Generation Online Airlines Reservation Systems," Research Square, Oct. 2024. https://www.researchsquare.com/article/rs-5182678/v1
[4] Mani M, Shrivastava P, Maheshwari K, Sharma A, Nath TM, Mehta FF, Sarkar B, Vishvakarma P. Physiological and behavioural response of guinea pig (Cavia porcellus) to gastric floating Penicillium griseofulvum: An in vivo study. J Exp Zool India. 2025;28:1647-56. doi:10.51470/jez.2025.28.2.1647
[5] Bachhav DG, Sisodiya D, Chaurasia G, Kumar V, Mollik MS, Halakatti PK, Trivedi D, Vishvakarma P. Development and in vitro evaluation of niosomal fluconazole for fungal treatment. J Exp Zool India. 2024;27:1539-47. doi:10.51470/jez.2024.27.2.1539
[6] Biman Barua et al., "Building Scalable Airlines Reservation Systems: A Microservices Approach Using AI and Deep Learning for Enhanced User Experience," IEEE, Mar. 2025. https://ieeexplore.ieee.org/abstract/document/10933362
[7] Vishvakarma P, Kaur J, Chakraborthy G, Vishwakarma DK, Reddy BBK, Thanthati P, Aleesha S, Khatoon Y. Nephroprotective potential of Terminalia arjuna against cadmium-induced renal toxicity by in-vitro study. J Exp Zool India. 2025;28:939-44. doi:10.51470/jez.2025.28.1.939
[8] Shatanik Bhattacharjee, "Microservices architecture and design: A complete overview," vFunction, Nov. 2024. https://vfunction.com/blog/microservices-architecture-guide/