Modernizing Transactional Systems: Deep Dive from Monolithic to Micro services Architectures

[1*]Krishna Kumaar Ragothaman

[*]Principal Software Engineer, Walmart Global Tech, Sunnyvale, CA, United States

[*]Corresponding Author Email id: krishnarsrch@gmail.com

*Orcid id: https://orcid.org/0009-0000-8637-0640

Abstract

The modern digital era depends on the transactional systems to maintain and process larger volumes of real-time transactions diagonally with the sectors such as e-commerce, banking, and online reservations which makes sure the protected payment processing, accurate account preservation, and real-time accessibility, are evolved from conventional monolithic and client-server architectures to more advanced micro services and server less computing models to meet increasing demands for accessibility, agility, and flexibility. As transaction volumes increase, monolithic system which combines the databases, business logic, and user interfaces into one unit face challenges with scalability and complexity. By splitting the applications into tiny, separate services, each of which oversees a distinct business process, micro services designs, on the other hand, enable fault isolation, autonomous scaling, and greater flexibility. Server less computing further converts the transactional systems by abstracting structural management, enabling businesses to focus on feature enhancement and cost-effectiveness through automatic scaling based on demand. This review inspects the evolution of transactional system architectures, highlighting the transference from monolithic structures to micro services and server less models. It investigates the role of these architectures in the sectors such as e-commerce, banking, and online transactions, addressing research questions about the most effective architectures for transactional applications and the influence of intelligent algorithms on real-time analytics and decision-making. By evaluating the benefits and challenges of each architectural approach, the review recommends a micro services architecture for one of the domains, highlighting its advantages in managing higher transactions, ensuring real-time accessibility, and providing scalable solutions, while also acknowledging difficulties in process management, data reliability, and system association. The review concludes that micro services architecture stands out as the most effective solution for high-volume transactional domains, offering superior scalability, real-time accessibility, and modular flexibility. However, it also emphasizes the need to address challenges in process orchestration, data consistency, and seamless system integration.

Keywords: Transactional Applications, Architectures, Monolithic Architectures, Micro service Architectures

1. Introduction

The transactional systems [1] is considered as the significant in sectors such as e-commerce, banking and online reservations in the maintenance and processing of the larger volumes of transactions in real time. It assures the crucial process such as safer processing of payments [2], exact maintenance of accounts and real time process accessibility. In the e-commerce sector [3], the transactional systems accomplishes the orders on products, payments and the inventory appraises. In the banking sector, it manages the money transactions with the assurance of accurate money transfers. Additionally, the online reservations [4] secures the real time process and inhibits the double-booking. In order to manage the unified business process, the frameworks of the transactional systems needs a consistent, reliable and accessible specifically in cases of system downs [5].

Correspondingly, in earlier days, the above mentioned sectors depends upon the conventional architectures of systems such as monolithic [6] and the client-server models. Also, the monolithic systems involves the components such as user interface, logics regarding the business and the database as a single unit, which is simple, since faces issues due to the growth of difficulties in systems and quantity of transactions [7, 8]. Additionally, the e-commerce sector [9] faces numerous challenges during the high shopping times, the banking sector [10] stumbles upon the difficulties due to the large transaction quantities and the online reservations [11] faces the issues in the management of larger simultaneous users. Similarly, the client-server architectures which is splitted from the client and server side functions enhances the scalability which also restricts in the performance and

flexibility. Those systems don't have the ability for effectively handling the developing difficulties of the recent integrations such as real time appraises and higher concurrencies.

Due to the requirements for additional accessible, active and resistant systems results in the development of recent architectures such as micro-services and server-less computations [12, 13]. Moreover, the micro services architectures splits the applications into small, individual services in which each of them manages a particular business process. For instance, the e-commerce sectors consists of payments and inventory managements with the customer support, where the above divisions autonomously balances the demands which results in the generation of fault separation, rapid distribution and flexibility. Additionally, the server less computations [14] converts the transactional systems with the extraction of structural management which permits the business for focussing on the feature enhancement. It habitually balances depending upon the demands, by making them high cost effective. Since, difficulties such as vendor lock-in and restricted control on the structures exists [15].

Hence, this conversion from the conventional architectures to the advanced architectures such as micro-services and server-less is considered to be crucial for tackling the enhanced difficulties in the transactional systems [16]. They permits the business for managing the higher transactions by assuring the real time accessibility with the generation of accessible solutions in the increasing demands, since faces difficulties in process management, data reliability and the system organization. Modernizing transactional systems from monolithic to micro services architectures involves several methods and techniques, including service decomposition, API gateway management, and containerization. Service decomposition breaks down complex applications into smaller, independent services that can be developed, deployed, and scaled autonomously. API gateways facilitate secure and efficient communication between micro services, while containerization with technologies like Docker ensures consistency and scalability across environments. However, challenges arise in terms of data consistency, as distributed systems require effective strategies for maintaining transactional integrity. Additionally, service orchestration and communication between multiple micro services can lead to complexity in process management. Finally, ensuring security and compliance in a distributed system environment requires robust monitoring and auditing mechanisms to mitigate vulnerabilities.

1.1 Motivations

The increasing demand for real-time, high-volume transactions in domains like e-commerce, banking, and online reservations has exposed the limitations of traditional monolithic systems. These legacy architectures struggle with scalability, agility, and fault isolation, making them inefficient for modern digital needs. Businesses require systems that can respond quickly to spikes in user activity while maintaining data integrity and availability. The need for modular, easily upgradable components has fuelled the shift toward micro services and server less computing. This transformation is driven by the goal of delivering faster, more resilient, and cost-effective digital services.

1.2 Innovations

The modernization of transactional systems has introduced key innovations such as containerization, dynamic service orchestration, and event-driven communication. Micro services enable independent scaling and development, while server less models like FaaS abstract away infrastructure concerns, allowing developers to focus solely on business logic. Advanced load balancing, intelligent routing, and integration of AI/ML for predictive analytics enhance system responsiveness and decision-making. Furthermore, innovations in security models, such as zero-trust architecture and block chain-based validation, are improving trust in distributed transactions. These advancements collectively redefine how transactional systems are built, managed, and scaled.

1.3 Paper objectives

The objectives of the below review paper includes:

- To evaluate the evolution of architectures from monolithic to modern architectures for transactional applications.
- To investigate the role of architectures in various domains such as e-commerce, banking and online transactions.
- To propose the micro- service architecture for any one of the domains based on the advantages.

1.4 Paper organization

The review paper is organized as follows, Section-2 elaborates the transactional application based on existing studies. Following that, Section-3 provides different architectures used in various domains and Section-4 about the evaluation of transactional applications. After that, Section-5 and Section-6 includes the challenges and the future recommendations of previous works. The final section includes the conclusion.

2. Methodology

The below review paper specifically focuses on the transactional systems by analysing the past five years between 2021-2025 published research studies, which serves as a key exclusion criterion. In this study, we utilized a thorough search methodology and a careful examination of the selected literature. Additionally, the language of publication was considered an important exclusion factor; non-English publications, not specific on e-commerce and study with insufficient experiments were filtered out by configuring the search engine to exclude them prior to retrieving results from digital libraries. Besides, it performed a manual evaluation by reviewing the titles and abstracts conclusion, and keywords of the paper and then selected the relevant studies while excluding those considered irrelevant. The process of manually reviewing and assessing the relevance of each publication is referred to as analysis. Table 1 describes a summary of the search results based on the database.

Table.1 Search Results

Article Name	**Number of Publication**
IEEE	5
MDPI	2
Springer	3
Taylor & Francis	0
Elsevier	3
Research Gate	0
Wiley	0
SSRN	1
Others	45
Total	59

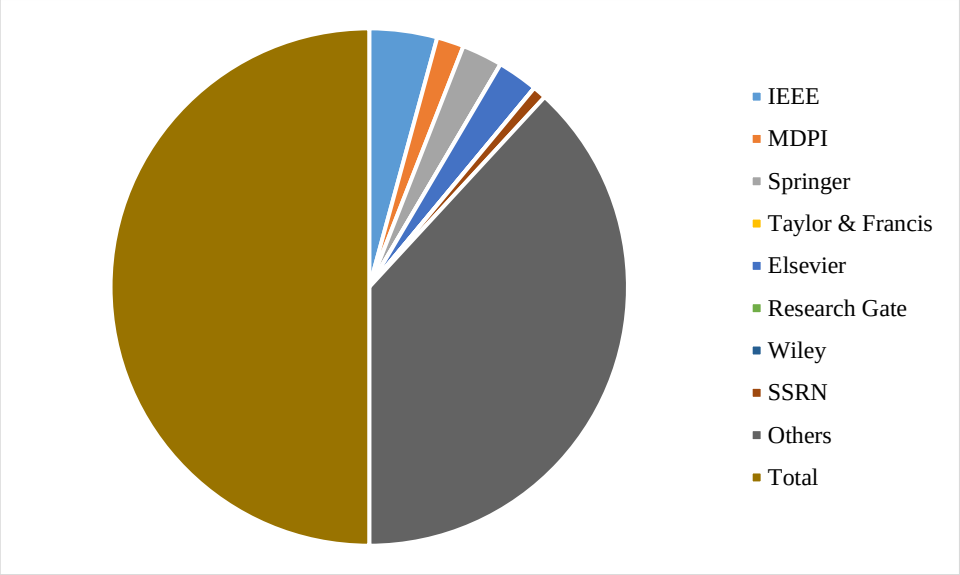


Figure.1 Search Results

Figure 1 and Table 1 illustrates the standard publications of the papers discussed in this review paper to investigate the transactional applications. The review is proceeded with research papers from 2021 – 2025 which is described in table 2.

3. Transactional Applications

The prevailing transactional systems in e-commerce, banking and the online reservations faces difficulties such as slow process with the privacy apprehensions and the issues of data reliability. The e-commerce sectors faces high traffic, the banking sectors face larger quantity of transactions and the online reservations face higher concurrencies and the real time apprises. The enhancements in the technology such as superior structure of network, cloud process and the modern database, permits the conversion of monolithic to the accessible and effective distributed architectures by tackling the challenges and increasing the performance of systems.

The transactional applications in the e-commerce sector is considered as crucial in enabling the online transactions among the consumers and sellers. Additionally, they restructures the process such as maintenance of orders, assuring the decreased invisibility with the maintenance of accuracy in data and has incorporated different data platforms for the enhancement of customer feedbacks and has augmented the process of business. Moreover, as the e-commerce extends, data security has been considered as significant. The study [17] has examined the privacy extortions such as ruptures and phishing and their influence on business and the customers. Additionally, developing technologies such as bio-metric verification and the post-quantum cryptography with the development of payments and the inventory management, by assuring a unified shopping's. The above applications improves the fulfilments of consumers with the generation of real time apprises and safer payments. Moreover, the transactional applications plays a vital role in dynamic sales and operational effectiveness in the digital markets. Correspondingly, the study [18] has examined the difficulties in the integration of real time processing in the e-commerce sector which has focussed in the management of larger data by active plans for safe online shopping platform and has highlighted the technology, guidelines and user's responsiveness.

Alternatively, the study [19] has demonstrated a model for protecting the e-commerce architectures by incorporating the modern encryption, real time analytics and the potential data supremacy. It has highlighted the privacy with the systematic accuracy with the examination of technologies such as homo-morphed encryption, blockchain and the AI anomaly identification. Additionally, it has incorporated the regulations such as GDPR and CCPA which has assured the safe, consistent and compliance handling for the e-commerce sector. Similarly, the study [20] has suggested a block chain based autonomous transaction model for the e-commerce sector which has utilized the IoT devices for the enhancement of effectiveness and the accuracy in decreasing the manual communications. Additionally, it has tackled the data storing issues with the utilization of block chain for a safe, scattered data savings and protection of user security from the external intimidations.

The transactional applications in banking sector is vital in the maintenance of daily finance process which enables the consumers for performing processes such as withdrawals, deposits and fund transfers effectively. It assures the safe and real time transactions, which improves the knowledge of users and hope. Additionally, they also assists in the maintenance of account and generates visions to the outlaying patterns. Moreover, they are considered to be crucial in the recent banking which provides accessibility and operational efficiency. They has also involved numerous of financial transactions daily. Transactional applications in banking involve handling a multitude of financial transactions daily, which must be processed accurately and reliably. These systems are designed to ensure that each transaction adheres to the ACID properties: Atomicity, Consistency, Isolation, and Durability. Atomicity ensures that a transaction is completed in its entirety or not at all; consistency maintains the integrity of the database by ensuring that each transaction transitions the system from one valid state to another; isolation ensures that concurrent transactions do not interfere with each other, providing a consistent view of the data; and durability guarantees that once a transaction is committed, it remains so, even in the event of system failures.

The transactional applications in the online reservations are significant in the maintenance of reservations for processes such as travels, rooms and events. Additionally, they enables the real time accessibility, safer payments and the prior confirmation with the improvement in user experience and satisfaction. It permits the easier changes and deletions in the enhancement of customer fulfilment. Moreover, they reorganize an effective and user-friendly reservation processes.

4. Architectural Models for Transactions

Notably, the architectural models for transactions generates a structured model for designing and executing the transaction process. Additionally, they explains the communications among several components such as user interfaces, application servers and databases by assuring the data integrity and reliability. Moreover, they assists in the enhancements of scalability and performance in tackling the privacy apprehensions. Hence, they results in the outlines in the development of potential transaction systems which deals with the needs of business.

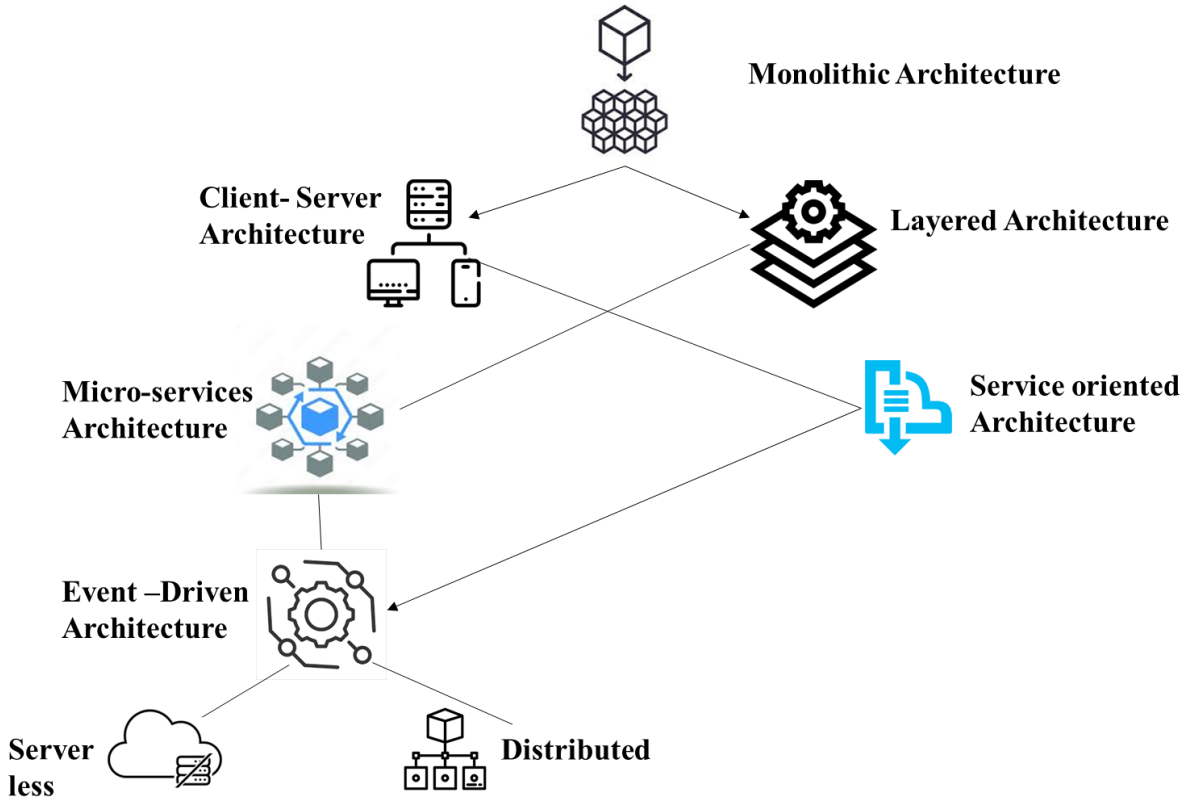


Figure.2 Architectures in Transactional Applications

Figure.2 shows the evolution of architectures from the monolithic to micro services including the client-server architecture, layered architecture, service oriented architecture, event driven architecture with the server less and distributed architecture.

4.1 Monolithic Architecture

The monolithic architecture associates the process such as transaction management, database management and the user interfaces to a single codebase which streamlines the growth and assures reliability. It obstructs the potential and scalability resulting in the difficulties for modifying to the higher transaction quantities

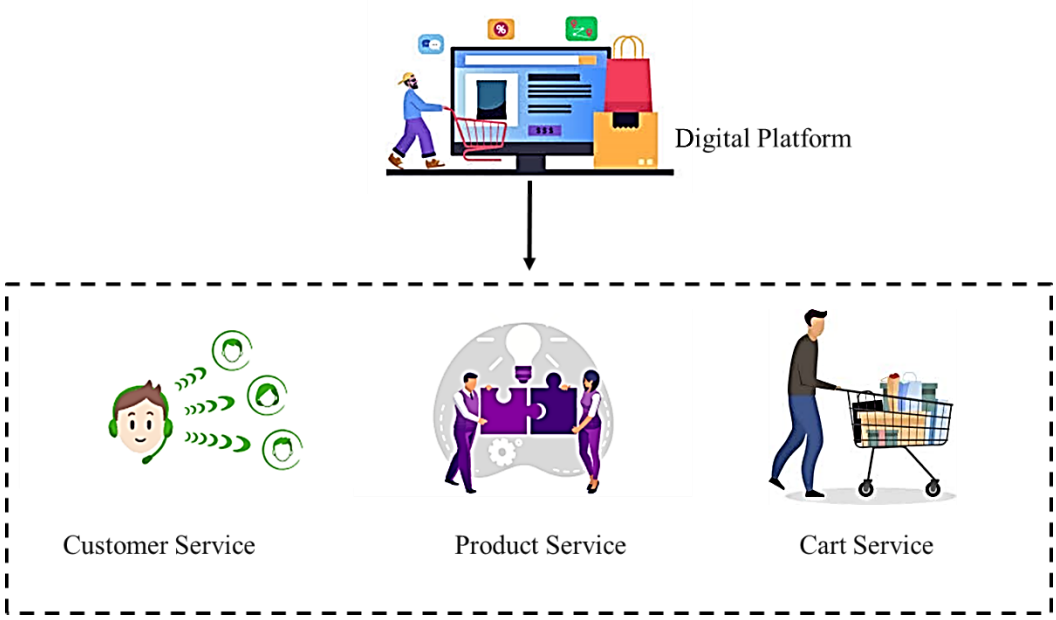


Figure.3 Monolithic Architecture

Moreover, the issues in any of the part of the systems distress the complete applications. The study [21] has demonstrated that the earlier systems has been monolithic consisting of the application, database and the user interface which has been incorporated compactly. Though the system has shortened the growth and testing, several challenges has been identified which has involved the scalability difficulties, complex enhancements and failure points. As a result, several systems has been converted into distributed architectures, since several parameters such as price, difficulties and the performance has returned the systems to monolithic methods. Additionally, the monolith has become the icon of anticipation which has been related to the donation systems. It has been a practical choice depending upon the needs of the system and specifications. The merits of the monolithic architectures includes the rapid distribution topology, ignores the difficulties in multiple distributed systems, simple management of workflow by the developers. Also, simple monitoring and the trouble shooting where the end-to-end testing has been done. Additionally, the re-usage of the code has been made enough without repetitions. Additionally, the monolith has become the icon of anticipation which has been related to the donation systems. It has been a practical choice depending upon the needs of the system and specifications. The merits of the monolithic architectures includes the rapid distribution topology, ignores the difficulties in multiple distributed systems, simple management of workflow by the developers. Also, simple monitoring and the trouble shooting where the end-to-end testing has been done. Additionally, the re-usage of the code has been made enough without repetitions [22].

The utilization of the monolithic architecture has demonstrated certain advantages in which the strong association among the domain entities has enabled the faster development [23]. Additionally, the integration of monolithic architectures has assured the reliable values in the software metrics such as difficulties and arranging capabilities nevertheless the quantity of features which has been integrated [24]. Moreover, the monolithic architecture has provided a consistent and stable environment in the development of software and it operations. Moreover, the single organised unit is referred to as the monolith. It has required the concurrent placement of the entire process. Organising the code into a unified operation has comprised the monolithic architecture. Similarly, the updated

single process monolith has divided the single process into numerous different modules which has created individually. Since, the distribution has required the pair of modules which has been an optimal solution for the initiatives and software store rooms due to the definition of module boundaries which has generated a particular quantity of works [25].

4.2 Client-Server Architecture

The client-server architecture has a crucial role in the distributed systems where the clients and servers has requested the resources from the other network which has resulted in the associated resources for evaluating randomly with the utilization of different applications via several architectures. It has involved the two-tier model which has been composed of client server tiers which has been a small setup whereas the three tier model has added the business logic layer for the scalability and privacy concerns. It has resulted in the difficult designs due to the maintenance demands. Additionally, certain advantages such as centralized data storage for privacy with simplicity and the scalability charesrterestics among the two tier architectures where the difficulties such as storage of centralized data, scaling difficulty and the implementation of N-tired method has made the architecture less interpretations [26].

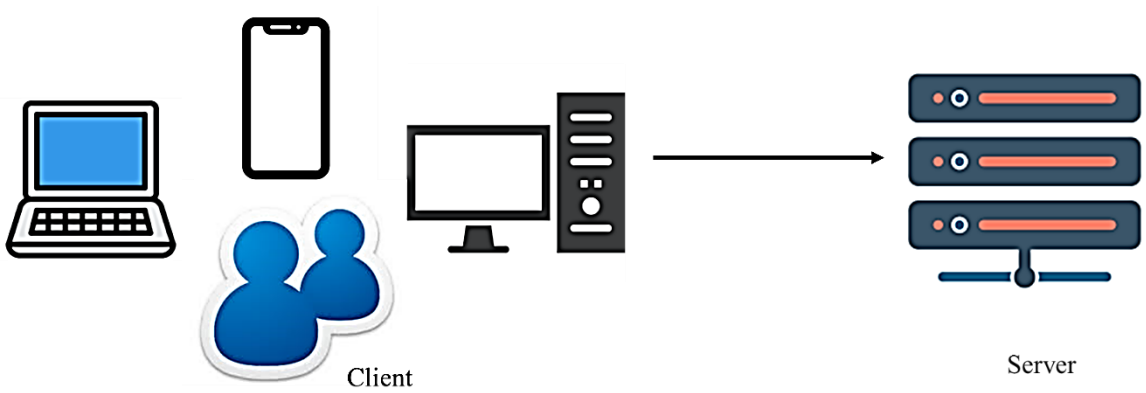


Figure.4 Client-Server Architecture

The existing research on applying enterprise architecture to customer service in industries like transportation offers valuable insights into improving operational efficiency through high-performance computing. However, it falls short in providing a comprehensive comparison with other architectural models, such as micro services or cloud-based solutions that could also enhance customer service. Additionally, the studies often lack real-world data or case examples, making it difficult to assess the practical effectiveness of the proposed approaches. Furthermore, the integration of emerging technologies like AI, IoT, or block chain is rarely discussed, leaving a gap in how future innovations could improve the architecture. Including these aspects, along with clear performance metrics, would provide a more holistic view and make the research more applicable to evolving industry demands [27].

4.3 Layered (N-tier) Architecture

The layered architecture in the transactional applications has classified the systems into different layers with each of them liable for particular tasks such as user communications, business logics and the data maintenance. The classification has assured the modularity which has made the system easy for maintenance, scaling and testing

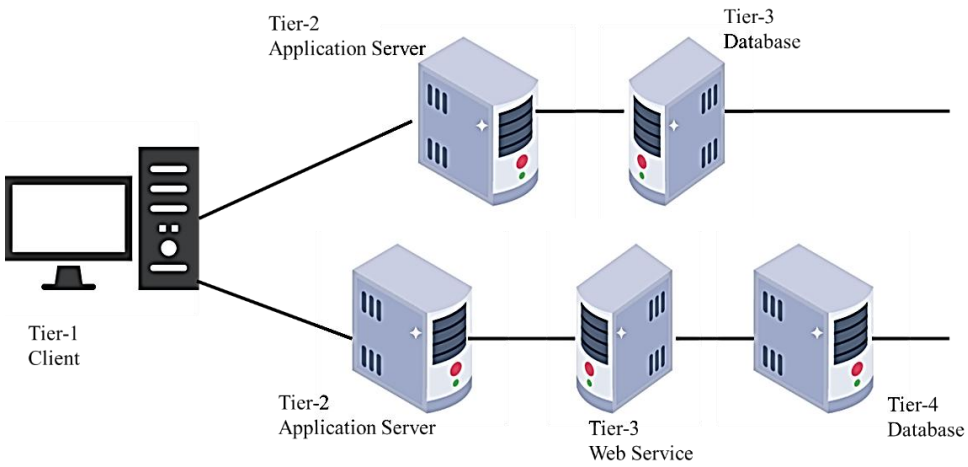


Figure.5 Layered (N-tier) Architecture

In the transactional applications, the layers has assisted in the management of difficult process with the assurance of reliability in the process such as payments. Additionally, with the isolation of diverse concepts, each layer has developed independently without influencing others. The layered architectures has enhanced the maintenance and potential of transactional systems which has enabled the casual transactions and high performance. The layered architecture has displayed the structure of software as layers. The study has suggested the advanced methods for assessing the layered architecture with the destructions such as back calls, skip calls, cyclic dependencies with the evaluation of logical separation among the layers. With the utilization of three tired layered architectures which has been common in java applications, the extraction of software architectures from the source code and integration to the additional metrics occur for the evaluation of layering criteria's. Additionally, the assessment of five software systems has revealed the fine-tuned criteria's for generating the exact examinations in the architectural reliability in contrast to the conventional metrics which has demonstrated the potential of verifying the observation for the layering criteria in the time of growth [28].

The study has revealed the decoupling the transaction processing from the prevailing two layered cloud based databases and has provided the transaction process as independent. With the construction of TaaS (Transaction as a Service) layer, the transactions are processed independently for the higher resource usage. Moreover, an execution-transaction-storage three layer cloud native database with the connection of TaaS, the AP engines has authorized with the ACID TP ability, numerous standalone TP engine instances which has been integrated for assisting the multi-master distributed TP for the straightened scalability. Additionally, the multiple execution engines with the diverse data models has been incorporated for assisting the multi-model transactions where the high performance has been obtained via wide TaaS optimizations and reliable growths [29].

4.4 Event-Driven Architecture

The EDA (Event-Driven Architecture) in the transactional applications has focused on the decoupling of components during the events. During the time of transactions, the event is provided and spread with the activation of services for responding asynchronously. The model has permitted the real time applicability along with the scalability and potentiality where the systems has responded to the events happening. They are adaptable for the applications with difficult, active process such as alerts regarding the orders.

Additionally, the EDA has enhanced the sensitivity and permitted the models for scaling individually with the reduction in the dependences among the components. The assumption of micro services architectures in the sector of banking and insurance has converted the process for permitting the scalability, flexibility and the increased experiences of users. The study [30] has examined the main design patterns which are crucial in the growth of scalable micro services in the finance sector. It has revealed the patterns such as event driven architecture which

has assured the real time applications with the circuit breaker pattern which has enhanced the fault acceptance during the time of failures. Additionally, the database per service pattern has maintained the loose coupling and has enhanced the performance with the allowance of every micro service for maintaining the responsible database.

Also, the BFF (Backend for Frontend) pattern has been highlighted for the creation of custom-made APIs which has improved the satisfaction of the users. In addition with, it has assured the consistency in the distributed transactions. Due to the requirement for quickness and real time applications in the digital business the background has resulted as dominant. The EDA (Event Driven Architecture) has been developed as the main component for the business for processing and examining the data during the time of event, by developing the faster decision making and replies. The study [31] has investigated the incorporation of EDA for obtaining the real time visions in highlighting the efficiency through diverse industries. Along with the growth on the real time data for the decision-making, EDA has deliberated the potential prototype for permitting the real time event processing and time analysis.

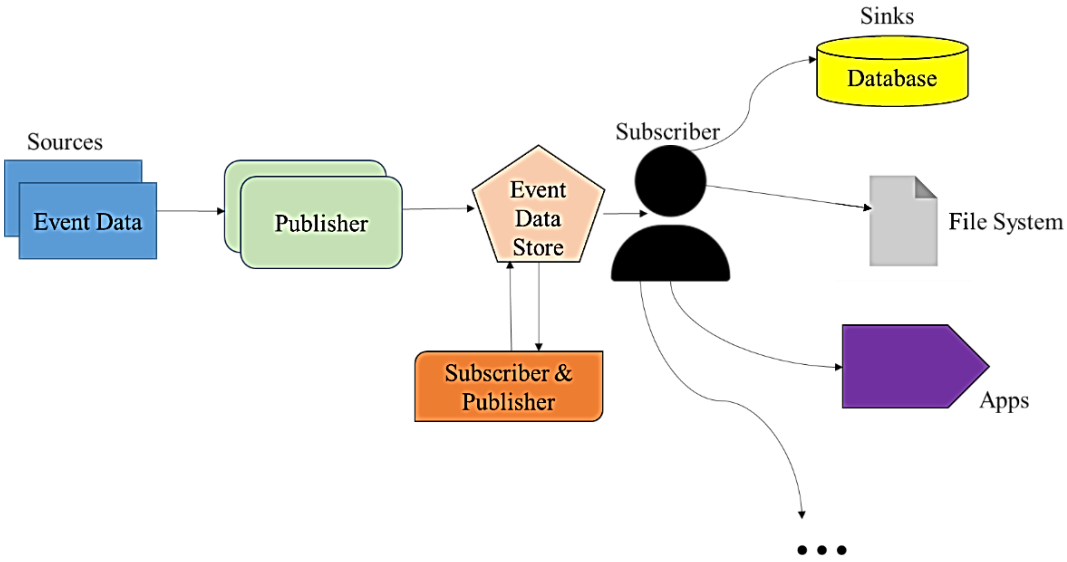


Figure.6 Event-Driven Architecture

Event-driven architectures for transaction management in microservices offer several advantages, including scalability, real-time processing, fault tolerance, and the decoupling of services. These architectures allow for asynchronous communication, enabling the system to scale more easily and handle large transaction volumes efficiently. Additionally, the loose coupling of services ensures that failures in one micro service do not impact the others, enhancing system resilience. However, there are also notable disadvantages. Managing data consistency across distributed services can be complex, especially when eventual consistency is used. Event ordering issues can arise, making it difficult to maintain the correct sequence of transactions. Debugging and monitoring become more challenging due to the asynchronous nature of events, and there may be additional latency introduced during event handling. Furthermore, the system may incur overhead due to the need to manage and process a high volume of events [32].

4.5 Service-Oriented Architecture (SOA)

The SOA has integrated to the modular, reusable and communicable services on the network. In the field of transactional applications, the SOA has enhanced the potentials, scalability and incorporation by permitting the services such as payment processing for operating individually. It has utilized the standard criteria's for the unified systems interactions with the enhancements in performance and the process maintenance in the difficult transactions.

In the industrial 4.0 era, the digital thread has permitted the unified data incorporation through the life cycle. The IoS (Internet of Services) model has assisted the difficult data analysis and has generated the incorporated part of smart products data. Though, the issues such as big data heterogeneity, data dominance and the admission control has been tackled specifically in the data flow through borders of industry 4.0 and the integration of smart cities.

The study [33] has developed the multi-tier SOA which has created for tackling the issues in the MICS (Made in Italy – Circular and Sustainable) has increased the business and has been sponsored by the next generation EU. Additionally, the SOA has faced several difficulties such as utilization of architectural metrics, performance via load testing [34]. Additionally, the SOA has increased the enterprise IT structure with the integration of business and technical operations to the individual, free couple operations. The study [35] has examined the growth of SOA challenges such as innovations in service infrastructures, growth of modernized applications like mobile cloud and health care which has required the interpretability.

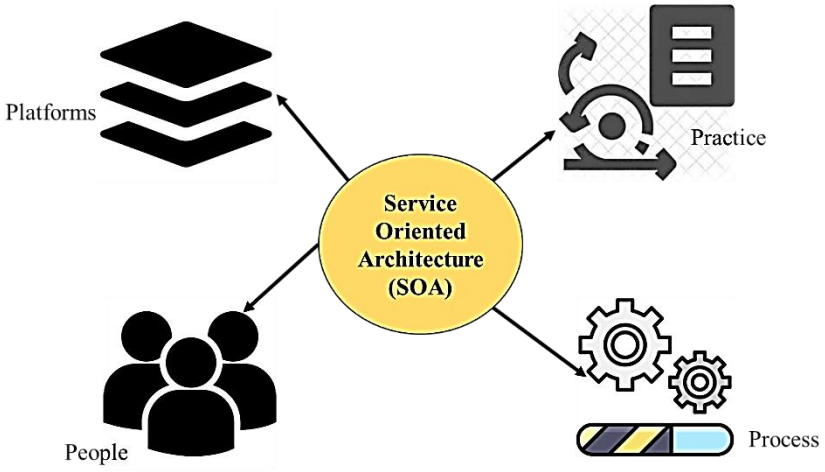


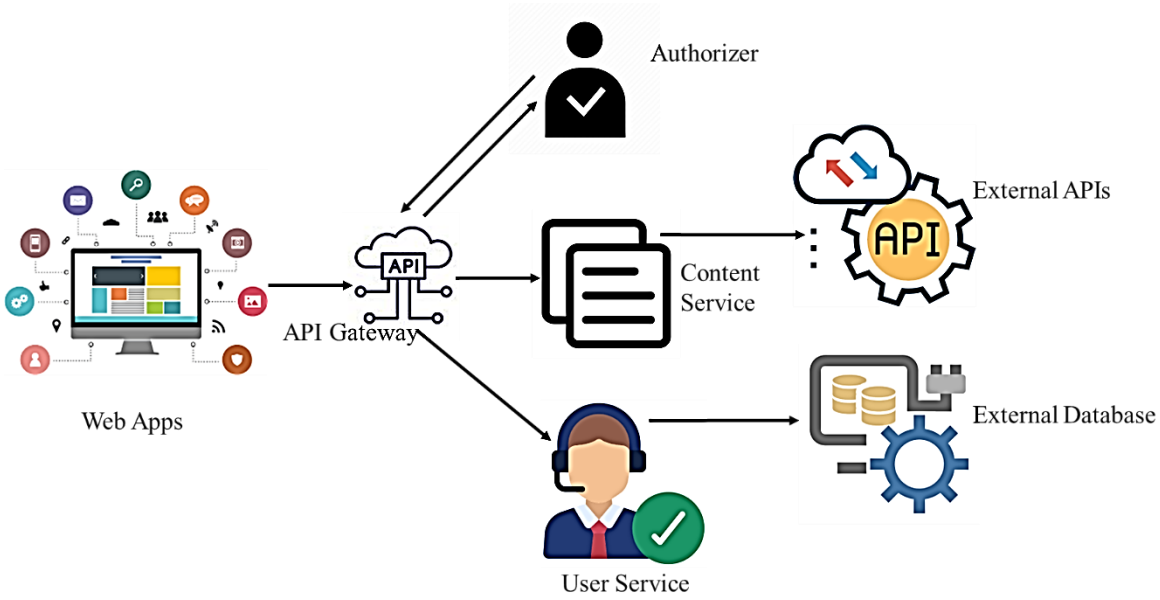
Figure.7 Service Oriented Architecture

The study [36] has assisted the SOA with the incorporation of numerous systems via well-defined APIs. The architecture has been built as seven service parts such as business applications, micro-services, databases, integration software's, infrastructures services , shared services and the user interfaces with the development of modularity and scalability. The utilization of API initial method has assured the casual interaction among the services where the micro-services and the cloud –based design has increased the potential and reuse which has been considered as the main factors of SOA.

4.6 Server less Architecture

The server-less architecture in the transactional applications has extracted the structural maintenance with the utilization of cloud systems where the coders has focused on the code. Each of the transaction has activated the small, displaced functions which has scaled automatically on demands. It has eliminated the requirement for managing the servers, reduction of operational costs. It has reduced the workloads by permitting the effective scaling and payments per usage models. Additionally, it has provided higher potential and rapid growth which has been made adaptable for the event-driven transactions and micro services.

Accordingly, the server less computing specifically the FaaS (Function-as-a-Service) has powered the positioning and scalability for the stateless function which has struggled with the stateful concepts. The methods such as Microsoft, Orleans and the azure durable functions has depended upon the external databases for the maintenance of state whereas the apache flink's state fun has incorporate the maintenance of state in the data flow and has provided the once processing models. The study [37] has developed the programming model for the transaction arrangement of the stateful server less functions, which has assisted the distribution systems in series and are ultimately reliable workflows. In the rapid development of financial sector, the requirement for ultra-low latency answers are essential significantly. On the other hand, the server-less computing has provided scalability, cost effectiveness and the decreased operational overhead. The study [38] has examined the incorporation of those trends, pointing towards the viability and enhancements in the server less architectures for the ultra-low latency presentations in the finance sector. Various issues has been examined including the cold starts, multi-tenancy and the difficulties of distributed system and has assessed the enhancement methods through the functions, platforms, architectures and the structures



.

Figure.8 Server less Architecture

Accordingly, the server less platforms such as AWS Lambda and the azure functions has been known for the management of stateless applications which has assisted the database-backed applications by permitting the functions for executing in the database transactions. Additionally, it has generated advantages such as time travel corrections and the consistent execution with the one semantics whereas the server less architectures which has been utilized for several operations such as image processing by expanding them for managing the stateful implementations associating with the databases such as postgres which has increased the utilization in e-commerce, banking and online reservations. Moreover, it has been one of the key advantages for permitting the server less architectures for the maintenance of backend systems for the modernized web servers [39].

4.7 Distributed Architecture

The distributed architectures in the field of transactional applications has classified the system into numerous inter-associated components through different locations which has increased the scalability, fault acceptance and the performance. Additionally, it has enabled the concurrent transaction process with higher possibilities and consistency during the assistance of real time transaction and effective resource consumption. Moreover, it has been considered as crucial in the maintenance of difficult and highly demandable transactions.

The distributed transaction has assured the consistency with the trade-offs such as enhanced delay which has been demonstrated via several contrasts among the monolithic and micro service systems. In order to tackle the above difficulties, the study [40] has suggested a decision making model for assisting the software engineers and architects for influencing the exact usage of distributed transactions which has involved the consistencies needs, performance applications and the adaptations of patterns such as saga for the event reliability. Additionally, it has aimed for streamlining the decision making process by assuring that the distributed transactions are integrated cautiously in the micro service architectures. The DAs (Distributed Architectures) differentiates with the monolithic architectures in the distribution on the process of applications through numerous individual services which each of the service is self-independent containing the crucial components for the process involving the individual databases with the deletion of mono sources of truth and permitting the services for utilizing the different technologies according to the requirements. Additionally, the communication among the services which has required that each service knows how to recruit contact with others without depending on the centralized orchestrator, which has resulted in coupling [41].

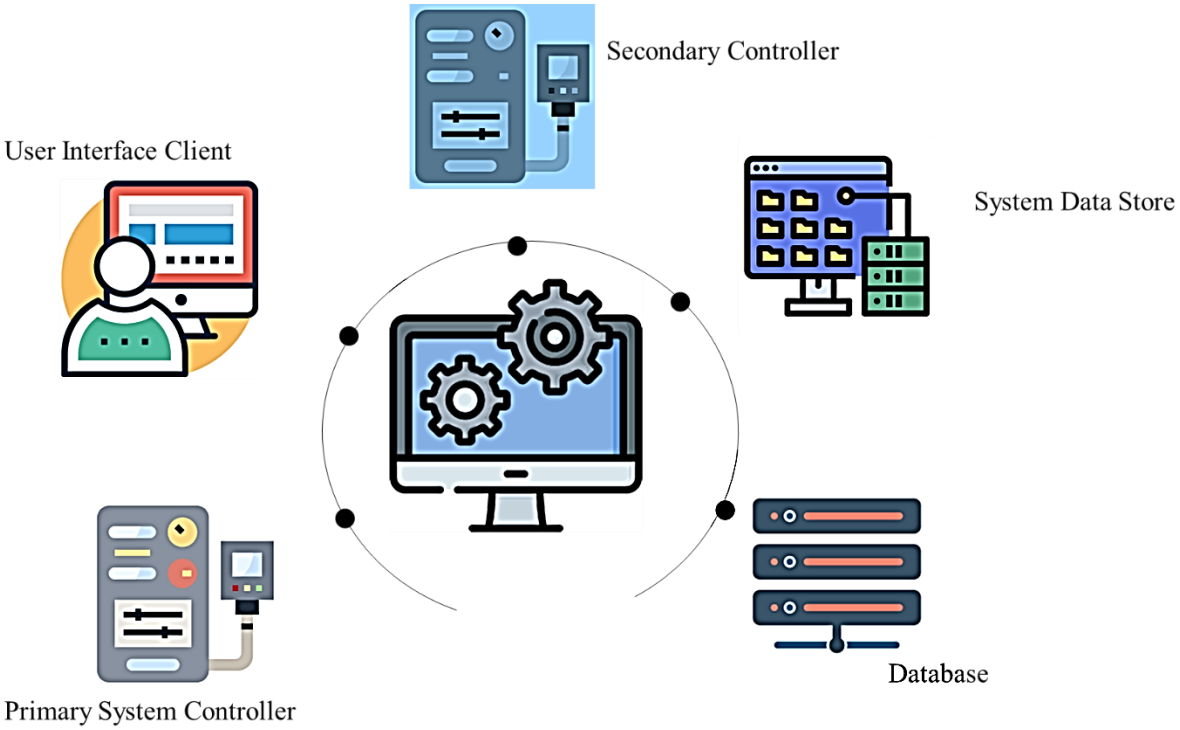


Figure.9 Distributed Architecture

Specifically, the cloud computing has demonstrated the crucial increase in the growth of distributed systems. It has permitted the demand availability of the computing resources on the internet which has generated scalability and potential which has not been attained. Additionally, this model has transferred the growth of additional models in software architectures specifically the micro-services architectures, which has classified the applications into small, individual components which has been developed, deployed and scaled with ease [42].

4.8 Micro-services Architecture

It has highlighted the modularity with the disintegration of the application into the individual services which has been developed, tested and deployed individually with the development of reusability. For instance, the user authentication service has been reused through numerous applications without alterations. Moreover, it has decreased the duplicate efforts and has permitted the teams for influencing the prevailing services for the rapid construction of additional features [43]. One of the crucial advantage of micro service has involved the capability of deploying the services individually which has resulted in the generation of additional features, bug fixes and the updates by not cooperating with several teams in the complete application. As a result, the deployment routines are small and the risk of introducing the issues in the larger systems has been reduced [44]. Additionally, the utilization of micro services has simplified the slower conversion from the legacy models. Moreover, the

organizations has increasingly replaced the monolithic components with the micro services which has decreased the risk and difficulties which has been related to the larger system renovations. Hence, it has permitted the quick adaptation of modernized technologies and observations [45].

Accordingly, it has generated a potential model for the constructing the scalable, manageable and robust applications. With the decomposition of services, it has deliberated the clearer boundaries and has enabled the individual deployment where the micro services has tackled the challenges of monolithic architectures and has provided multiple advantages. The potential for the selection of better technologies for each of the service has raised the innovation and has increased the growth of companies, resulting the micro-services as the immediate cause for modernized application growth [46]. Subsequently, the design patterns has occupied a significant part in the efficient incorporation of micro services architectures. The frequent pattern such as API gateways, SR (Service Registry) and discoveries has generated the initial abilities for the maintenance of communications and service possibilities. Moreover, the modern patterns such as saga and CQRS has tackled the difficulties such as distributed transactions and the performance enhancements. With the cautious selection and the implementation of design patterns resulted in the potential, accessible and resistant micro services based models [47].

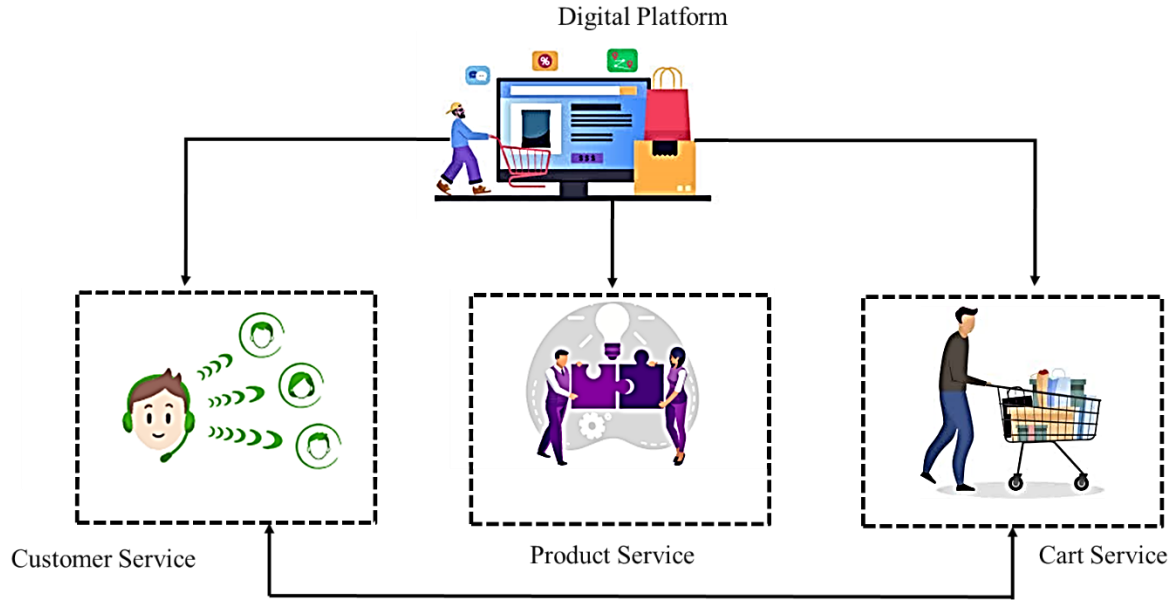


Figure.10 Micro-services Architecture

Subsequently, it has tackled the issues of permitting the individual deployment and service scaling where each of the service has been organized individually with the decrease in the difficulty of downtime and permitting common apprises. The CI and CD of the pipelines could be integrated for the automation of deployment operations which has assured the rapid and reliable outcomes [48]. Accordingly, the study [49] has demonstrated the MSA (Micro Service Architecture) as the process of segmenting the larger application into the smaller services which can be built, tested and has deployed independently. This relates to several benefits like fault isolation enhancement, scalability, enhanced development agility and so on. Micro services are fine-grain and work independent of one another, which means that different teams can work on each micro service depending on its complexity [50].

Table.2 depicts the differences between monolithic and micro service architecture in which the micro service architecture has obtained higher scalability, rapid development, deployment, fault isolation and technology stack.

Table.2 Differences between Monolithic and Micro service Architectures [51]

Attributes	**Monolithic Architecture**	**Micro service Architecture**
Scalability	The complete application has to be scaled	Individual scaling's

Rapid Development	Slow in larger applications	Fast due to the working of independent teams on systems
Deployment	Slow and difficult as there is a single unit	The services are independently deployed
Fault Isolation	The entire system is influenced during failures	Failures are isolated for particular services
Technology Stack	Restricted for single technology stack	Every single service has own stack which has enabled the flexibility

5. Evaluation of Transactional Architectures

The evaluation of transactional architectures mainly focuses on the differences among the various architectures depending upon the attributes such as scalability, development speed, fault isolation along with the deployment complexity and the fault isolation. The conventional architectures such as monolithic and client-server has faced challenges in the scalability and fault isolation, whereas the modernized architectures such as micro services and the event driven architectures has provided high flexibility, fault tolerance and scalability.

Table.3 Difference among Various Architectures

Architecture	Scalability	Development Speed	Deployment Complexity	Fault Isolation
Monolithic	Difficult to scale individual components	Fast initially, slows down	Simple	Failure in one affects entire system
Client-Server	Limited by server capacity	Moderate	Moderate	Server failure affects all clients
Layered (N-Tier)	Improved, but complex to scale	Moderate	Moderate	Isolation between layers, not complete
Event-Driven	Highly scalable via asynchronous events	Moderate to Fast	Can be complex	Components decoupled, isolates faults
Service-Oriented (SOA)	Scalable through reusable services	Moderate	Moderate to Complex	Service failures isolated
Server less	Highly scalable, auto-scaling	Fast	Simplified infrastructure	Isolated to function executions
Distributed	Highly scalable, load distributed	Complex	Complex	Fault-tolerant through redundancy
Micro services	Highly scalable, service-level scaling	Fast, independent teams	Complex	Isolated to individual services

Table.3 deliberates the overall attributes of the architectures where the micro service has offered highly scalable, service level scaling. Rapid independent teams, complex deployment and isolated to independent services.

Table.4 Difference among Various Architectures

	Monolithic	Micro services	Event-Driven	Server less	Distributed
Scalability	Low	High	High	High	High
Performance	High	Medium	Medium	High	Medium
Fault Tolerance	Low	High	High	High	High

Deployment Complexity	Low	High	Medium	Low	High
Consistency	Strong	Eventual	Eventual	Eventual	Eventual
Technology Flexibility	Low	High	Medium	Medium	High

Table.4 deliberates the range of attributes such as scalability, performance, fault, tolerance, deployment complexity, consistency and technology flexibility where the micro services architecture has obtained better ranges.

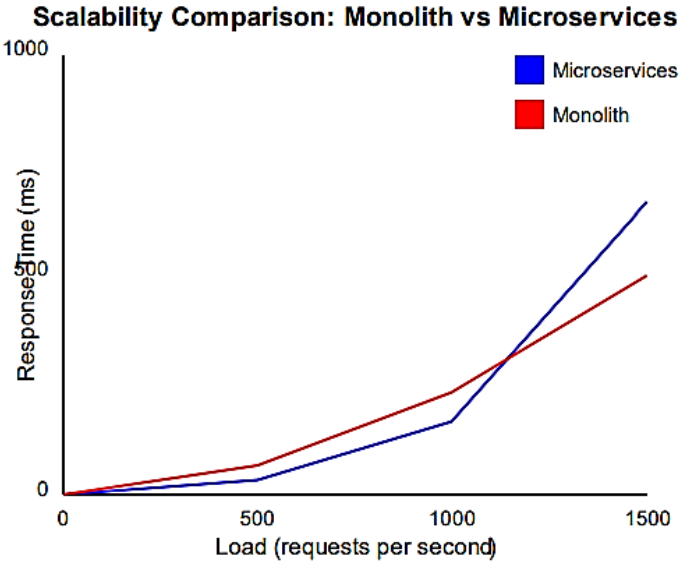


Figure. 11 Performance comparison of monolithic and micro service architectures under increasing load [52]

Figure.11 deliberates the scalability comparison of the monolithic and micro service architecture where the micro service has obtained better scalability.

6. Challenges in Implementing Transactional Architectures

The integration of transactional architectures has demonstrated several issues that the organizations has to tackle for the assurance of success [53]. One of the main issues is the maintenance of data consistency [54] through the distributed systems specifically in the failures of network. In addition with, the scalability [55] has been the main challenge where the architecture has to maintain the enhancing transactions without the degradation on performance. Subsequently, the latency [56] has been the vital issue in larger systems, where the small delay in the transactions has resulted in the deprived experience of users. The fault tolerance is considered as the crucial in the assurance that the transactions are not missed during the failures in systems, whereas its integration is complex [57]. Also, the distributed transactions frequently faced issues in obtaining the consistency through numerous databases which has made the architecture difficult for designing and managing. The requirement for the real time processing in some of the systems needs an effective resource maintaining for the avoidance of overloading the structure. Other challenge is the assurance of security in the transactions for protecting the sensitive data from the breaches and unrecognized access, which influences the growth of systems [58].

The compliances with the industry acts such as GDPR, PCI-DSS [59] has been incorporated into the architectures without impacting the performance. Also, the transactions rollback mechanisms has been cautiously framed for handling the issues with the assurance of data integrity. The incorporation of transactional architecture with the legacy has demonstrated the compatibility and transformation issues [60]. Hence, the maintenance of higher availability and the reduction of downtime has been essential as the system outcomes has impacted the transaction and business process [61, 62].

7. Future Recommendations in Transactional Applications

The evolving trends and technologies in the transactional applications are demonstrating the innovations and effectiveness. Additionally, the blockchain technologies are gaining attentions for their assurance on transparency and immutability in the transactions by providing the distributed resolutions for secured transactions. The AI and ML are utilized for forecasting the frauds, automatic transaction process and increased decision-making. In addition with, the cloud based transactional systems are growing familiar for their scalability, flexibility and the cost effectiveness. Also, the edge computing enhances the real time transactions for extracting the data processing nearer to the source by decreasing the latency. The micro services architecture permits the modularity, permitting the ease in scaling's and the management of transactional systems. The evolution of 5G systems is also increasing the speeds of transactions and consistency specifically in mobile phones and IoT based transactions. However, the enhancements in the quantum computing embraces the flexibility for developing the privacy and power generation for difficult transactions in future.

8. Conclusion

The evolution of transactional systems from monolithic to microservices architectures marks a significant advancement in scalability, flexibility, and fault isolation—key qualities for modern applications in e-commerce, banking, and online reservations. Microservices address the limitations of monolithic architectures by decomposing applications into independently deployable services, facilitating faster development cycles, enabling technology diversity, and enhancing resilience. While microservices introduce challenges in deployment, consistency management, and inter-service communication, their advantages—such as service-level scaling and fault isolation—make them the preferred choice for organizations seeking greater agility and scalability. A comparative evaluation of various architectures highlights the dominance of microservices in handling the demands of high-transaction environments. Despite the complexities involved in their implementation, the shift toward microservices is driven by the need for scalable, manageable, and robust applications, with the support of emerging technologies like blockchain, AI, and cloud computing. The ability to independently deploy and scale services minimizes downtime and facilitates continuous updates, cementing microservices as a cornerstone of modern application development. Furthermore, the adoption of established design patterns further enhances the accessibility, resilience, and effectiveness of microservices-based models, solidifying their crucial role in transforming transactional systems.

9. Declaration

- **Conflict of Interest:** The author reports that there is no conflict of Interest.
- **Funding**: None
- **Acknowledgement**: None

Reference

[1] L. Albshaier, S. Almarri, and M. J. C. Hafizur Rahman, "A review of blockchain's role in E-Commerce transactions: Open challenges, and future research directions," *Computers* vol. 13, no. 1, p. 27, 2024.

[2] K. B. Raj *et al.*, "Optimizing Financial Transactions and Processes Through the Power of Distributed Systems," *Meta Heuristic Algorithms for Advanced Distributed Systems* pp. 289-303, 2024.

[3] A. Hafid, S. Bahri, S. N. Marzuki, M. Muis, and R. J. S. J. H. K. d. H. I. Idayanti, "The Application of Khiyar Principles to E-Commerce Transaction: The Islamic Economy Perspective," *Samarah: Jurnal Hukum Keluarga dan Hukum Islam* vol. 8, no. 1, pp. 403-420, 2024.

[4] M. Işkın, C. Prentice, N. Eker, Ü. J. E. J. o. T. Şengel, Hospitality, and Recreation, "Understanding the Effects of Reservation Systems and Online Transaction Capacity on the Competitiveness of Travel Agencies," *European Journal of Tourism, Hospitality Recreation* vol. 14, no. 1, pp. 55-70, 2024.

[5] S. Selvarajan and H. Mouratidis, "A quantum trust and consultative transaction-based blockchain cybersecurity model for healthcare systems," *Scientific Reports*, vol. 13, no. 1, p. 7107, 2023.

[6] D. C. Hidayat, I. K. J. Atmaja, and I. B. G. J. J. G. Sarasvananda, "Analysis and Comparison of Micro Frontend and Monolithic Architecture for Web Applications," *Jurnal Galaksi* vol. 1, no. 2, pp. 92-100, 2024.

[7] M. Felisberto, "The trade-offs between Monolithic vs. Distributed Architectures," *arXiv preprint arXiv:2405.03619*, 2024.
[8] K. Barde, "Modular monoliths: Revolutionizing software architecture for efficient payment systems in fintech," *International Journal of Computer Trends and Technology*, vol. 71, no. 10, pp. 20-27, 2023.
[9] H. Samoylenko and A. Selivanova, "Features of microservices architecture in e-commerce systems," *Math. Mach. Syst*, vol. 3, pp. 51-58, 2023.
[10] O. Bodemer, "Blockchain Enterprise Architecture: Monolith or Microservices in the Financial Industries," *Authorea Preprints*, 2023.
[11] V. Patil and R. Shyamasundar, "A Decoupling Mechanism for Transaction Privacy," in *International Conference on Information Systems Security*, 2025, pp. 359-379: Springer.
[12] M. T. González-Aparicio, M. Younas, J. Tuya, and R. Casado, "A transaction platform for microservices-based big data systems," *Simulation Modelling Practice and Theory*, vol. 123, p. 102709, 2023.
[13] A. Kamisetty, D. Narsina, M. Rodriguez, S. Kothapalli, and J. C. S. Gummadi, "Microservices vs. Monoliths: Comparative Analysis for Scalable Software Architecture Design," *Engineering International*, vol. 11, no. 2, pp. 99-112, 2023.
[14] R. Ouyang *et al.*, "A Microservice and Serverless Architecture for Secure IoT System," *Sensors*, vol. 23, no. 10, p. 4868, 2023.
[15] C. Oyeniran, A. O. Adewusi, A. G. Adeleke, L. A. Akwawa, and C. F. Azubuko, "Microservices architecture in cloud-native applications: Design patterns and scalability," *Computer Science & IT Research Journal*, vol. 5, no. 9, pp. 2107-2124, 2024.
[16] Z. Li, H. Yu, G. Fan, and J. Zhang, "Cost-efficient fault-tolerant workflow scheduling for deadline-constrained microservice-based applications in clouds," *IEEE Transactions on Network and Service Management*, vol. 20, no. 3, pp. 3220-3232, 2023.
[17] G. C. J. G. A. R. Oguta and Reviews, "Securing the virtual marketplace: Navigating the landscape of security and privacy challenges in E-Commerce," *GSC Advanced Research Reviews* vol. 18, no. 1, pp. 084-117, 2024.
[18] V. M. Reddy, L. N. J. I. J. o. A. E. T. Nalla, and Innovations, "Real-time Data Processing in E-commerce: Challenges and Solutions," *International Journal of Advanced Engineering Technologies Innovations* vol. 1, no. 3, pp. 297-325, 2024.
[19] N. HIDAYAH *et al.*, "ENHANCING E-COMMERCE DATA ARCHITECTURES: SOPHISTICATED FRAMEWORKS FOR PRECISE ANALYTICS AND INFORMED DECISION-MAKING," *Journal Gate*
[20] S. Sekar *et al.*, "Autonomous transaction model for e-commerce management using blockchain technology," *International Journal of Information Technology Web Engineering* vol. 17, no. 1, pp. 1-14, 2022.
[21] S. Sethi, S. J. J. o. C. Panda, and Communications, "Transforming Digital Experiences: The Evolution of Digital Experience Platforms (DXPs) from Monoliths to Microservices: A Practical Guide," *Journal of Computer Communications* vol. 12, no. 2, pp. 142-155, 2024.
[22] G. Blinowski, A. Ojdowska, and A. J. I. A. Przybyłek, "Monolithic vs. microservice architecture: A performance and scalability evaluation," *IEEE Access* vol. 10, pp. 20357-20374, 2022.
[23] D. Faustino, N. Gonçalves, M. Portela, and A. R. J. P. E. Silva, "Stepwise migration of a monolith to a microservice architecture: Performance and migration effort evaluation," *Performance Evaluation* vol. 164, p. 102411, 2024.
[24] N. C. Mendonça, C. Box, C. Manolache, and L. J. I. s. Ryan, "The monolith strikes back: Why istio migrated from microservices to a monolithic architecture," *IEEE software* vol. 38, no. 5, pp. 17-22, 2021.
[25] A. M. Saucedo, G. Rodríguez, F. G. Rocha, R. P. J. I. dos Santos, and S. Technology, "Migration of monolithic systems to microservices: A systematic mapping study," *Information Software Technology* vol. 177, p. 107590, 2025.
[26] M. G. M. Nyabuto, M. V. Mony, and S. Mbugua, "Architectural Review of Client-Server Models," *IJSRET*, 2023.
[27] D. J. J. o. C. N. Hindarto, Architecture and H. P. Computing, "Application of customer service enterprise architecture in the transportation industry," *Journal of Computer Networks, Architecture High Performance Computing* vol. 5, no. 2, pp. 682-692, 2023.

[28] S. Thakare and A. W. J. a. p. a. Kiwelekar, "Redefining measures of Layered Architecture," *arXiv preprint arXiv:.03079* 2021.

[29] Y. Zhang, W. Zhou, Y. Ren, S. Li, G. Li, and G. J. a. p. a. Yu, "Towards Transaction as a Service," *arXiv preprint arXiv:.07874* 2023.

[30] S. A. J. I. J. o. E. T. i. C. S. Fathima and I. Technology, "Design Patterns for Scalable Microservices in Banking and Insurance Systems: Insights and Innovations," *International Journal of Emerging Trends in Computer Science Information Technology* vol. 1, no. 01, pp. 1-11, 2025.

[31] P. J. J. o. T. I. Badgujar, "Implementing Event-Driven Architectures for Real-Time Insights," *Journal of Technological Innovations* vol. 5, no. 1, 2024.

[32] P. S. Yadav, "DESIGN AND EVALUATION OF EVENT-DRIVEN ARCHITECTURES FOR TRANSACTION MANAGEMENT IN MICROSERVICES," *International Journal of Core Engineering & Management,* pp. Volume-6, Issue-07, 2020 ISSN No: 2348-9510, 2020

[33] D. Bianchini and M. J. T. Garda, "A Service-Oriented Architecture for implementing Digital Threads in Smart Products Design," *Traffic* vol. 1, no. S2, p. S3.

[34] V. Raj, R. J. I. J. o. C. N. Sadam, and D. Systems, "Performance and complexity comparison of service oriented architecture and microservices architecture," *International Journal of Communication Networks Distributed Systems* vol. 27, no. 1, pp. 100-117, 2021.

[35] S. J. A. a. S. Ahamad, "Contemporary Research Challenges and Applications of Service Oriented Architecture," *Available at SSRN 3948566* 2021.

[36] L. Chamari, E. Petrova, and P. J. I. A. Pauwels, "An end-to-end implementation of a service-oriented architecture for data-driven smart buildings," *IEEE Access* 2023.

[37] M. de Heus, K. Psarakis, M. Fragkoulis, and A. J. I. S. Katsifodimos, "Transactions across serverless functions leveraging stateful dataflows," *Information Systems* vol. 108, p. 102015, 2022.

[38] P. S. J. E. J. o. A. i. E. Yadav and Technology, "Optimizing Serverless Architectures for Ultra-Low Latency in Financial Applications," *European Journal of Advances in Engineering Technology* vol. 11, no. 3, pp. 146-157, 2024.

[39] Q. Li and P. J. C. o. t. A. Kraft, "Transactions and Serverless are Made for Each Other," *Communications of the ACM* vol. 67, no. 12, pp. 52-56, 2024.

[40] A. Bashtovyi and A. Fechan, "DISTRIBUTED TRANSACTIONS IN MICROSERVICE ARCHITECTURE: INFORMED DECISION-MAKING STRATEGIES," 2024.

[41] M. J. a. p. a. Felisberto, "The trade-offs between Monolithic vs. Distributed Architectures," *arXiv preprint arXiv:.03619* 2024.

[42] B. Cloud and S. Lyu, "Practical Rust Web Projects," *Journal of Sustainable Technologies and Infrastructure Planning* 2023.

[43] K. Cannon *et al.*, "GstLAL: A software framework for gravitational wave discovery," *SoftwareX* vol. 14, p. 100680, 2021.

[44] B. Sejdiu, F. Ismaili, and L. J. C. Ahmedi, "IoTSAS: An integrated system for real-time semantic annotation and interpretation of IoT sensor stream data," *Computers,* vol. 10, no. 10, p. 127, 2021.

[45] D. R. Apolinário and B. B. J. J. o. t. B. C. S. de França, "A method for monitoring the coupling evolution of microservice-based architectures," *Journal of the Brazilian Computer Society* vol. 27, no. 1, p. 17, 2021.

[46] H. F. O. Rocha, *Practical event-driven microservices architecture: building sustainable and highly scalable event-driven microservices*. Springer, 2021.

[47] N. Zhou *et al.*, "Container orchestration on HPC systems through Kubernetes," *Journal of Cloud Computing* vol. 10, pp. 1-14, 2021.

[48] Y. J. E. J. o. A. i. E. Jani and Technology, "Spring boot for microservices: Patterns, challenges, and best practices," *European Journal of Advances in Engineering Technology* vol. 7, no. 7, pp. 73-78, 2020.

[49] C. P. Singh, "From Monoliths to Microservices: Transition Strategies and Success Metrics in Modern Software Development," *Journal of Advances in Developmental Research (IJAIDR),* 2022.
[50] H. Calderón-Gómez *et al.*, "Evaluating service-oriented and microservice architecture patterns to deploy ehealth applications in cloud computing environment," *Applied Sciences* vol. 11, no. 10, p. 4350, 2021.
[51] I. Ortiz and S. González, "Overcoming Challenges in Microservice Architectures."
[52] H. R. V. M. Vamsi Krishna Thatikonda, "Microservices vs. Monoliths: Choosing the Right Architecture for Your Project " *International Journal of Software Computing and Testing* vol. Volume 10, Issue 2, pp. ISSN: 2456-2351, 2024.
[53] E. Solberg, "The transition from monolithic architecture to microservice architecture: A case study of a large Scandinavian financial institution," 2022.
[54] S. Sethi and S. Panda, "Transforming Digital Experiences: The Evolution of Digital Experience Platforms (DXPs) from Monoliths to Microservices: A Practical Guide," *Journal of Computer and Communications,* vol. 12, no. 2, pp. 142-155, 2024.
[55] G. Blinowski, A. Ojdowska, and A. Przybyłek, "Monolithic vs. microservice architecture: A performance and scalability evaluation," *IEEE access,* vol. 10, pp. 20357-20374, 2022.
[56] R. Manchana, "Balancing Agility and Operational Overhead: Monolith Decomposition Strategies for Microservices and Microapps with Event-Driven Architectures," *North American Journal of Engineering Research,* vol. 2, no. 2, 2021.
[57] M. Kaloudis, "Evolving Software Architectures from Monolithic Systems to Resilient Microservices: Best Practices, Challenges and Future Trends," *International Journal of Advanced Computer Science & Applications,* vol. 15, no. 9, 2024.
[58] H. Knoche and W. J. a. p. a. Hasselbring, "Fast and Efficient What-If Analyses of Invocation Overhead and Transactional Boundaries to Support the Migration to Microservices," 2025.
[59] S. C. Rajesh and E. A. Jain, "Integrating Security and Compliance in Distributed Microservices Architecture," 2024.
[60] L. Regander and C. O'Driscoll, "Challenges and Considerations of Architectural Patterns for Payment Solutions," ed, 2024.
[61] K.-H. HSU, Y.-Y. CHEN, A. J. J. o. I. S. WEIAN HOU, and Engineering, "MAT-Go: A Study on Automated Transformation of Monolithic Go Application Systems Into Microservice Architecture," vol. 41, no. 1, 2025.
[62] A. Katal, P. Prasanna, R. Birla, and Kunal, "Evolution from Monolithic to Microservices Architecture: A New Era in Software Architecture," in *Advancements in Optimization and Nature-Inspired Computing for Solutions in Contemporary Engineering Challenges*: Springer, 2025, pp. 235-279.