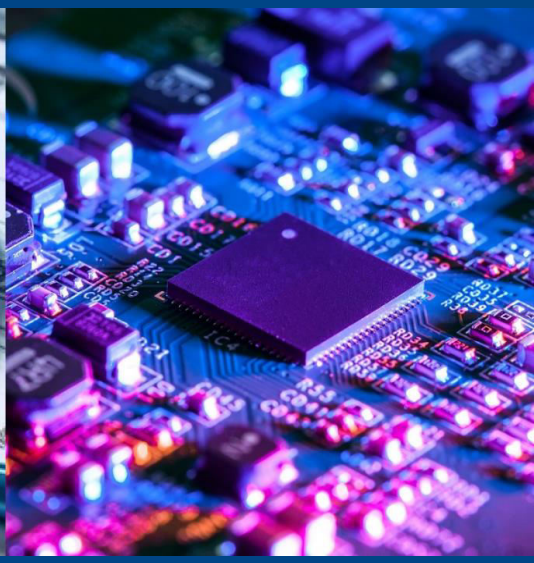
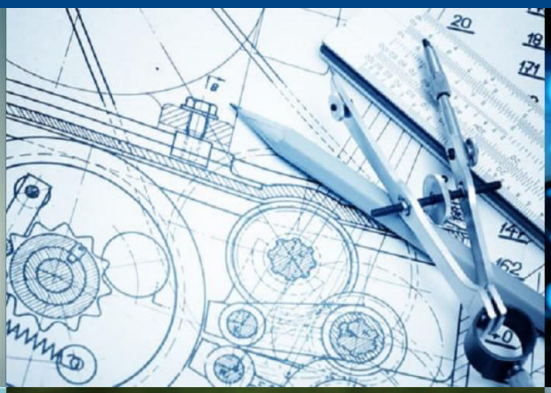
ISSN: 2582-7219

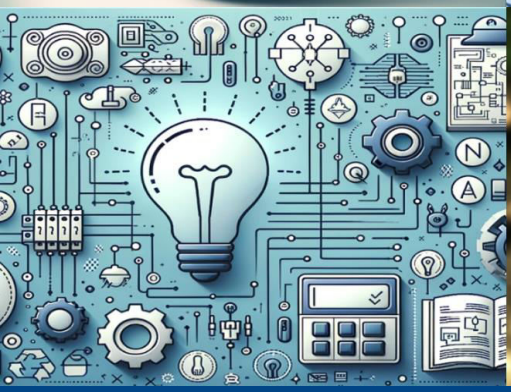




International Journal of Multidisciplinary Research in Science, Engineering and Technology

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 8.206 **Volume 8, Issue 6, June 2025**

© 2025 IJMRSET | Volume 8, Issue 3, March 2025| DOI:10.15680/IJMRSET.2025.0803349

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Modern Middleware for Real-Time Banking Enhancing Transaction Speed and Reliability

Srikanth Mannem

Senior Manager, Virtusa Corporations, USA

ABSTRACT: This document provides a detailed presentation of the Real-time Service Integration (RSI) Middleware Architecture. This cloud-native solution aims to help financial institutions modernize the banking transaction process for large numbers of consumers. Engineered for high-speed transaction processing and performance, this architecture supports transaction latency of under 100 milliseconds, with the ability to process over 10,000 transactions per second while providing an uptime availability rate of 99.99% with ACID compliance between the old legacy systems and the new CORE banking systems using a canonical data model and orchestration technologies such as Kafka Streams and Dynamic Payment Rails. Among the many significant advances provided by this architecture are Delta-Lake Analytics, NewSQL with CockroachDB, Microservices built on Spring Footprint and Containers (Kubernetes), and a Disaster Recovery Plan with a Recovery Time Objective (RTO) of fewer than 15 minutes, estimated at saving at least $200 million per year in cloud infrastructure costs. The document also emphasizes how useful Agile Methodologies are for Continuous Delivery and engagement with stakeholders while also fully specifying potential future enhancements to this architecture including Edge Computing; Central Bank Digital Currency (CBDCs) based on Blockchain technology; and Artificial Intelligence (AI) driven Fraud Detection to improve Banking Security and Efficiency. The RSI Middleware architecture allows financial institutions to operate more efficiently and modernize their data systems resulting in greater financial savings and improved performance metrics. As the platform is modular in design, the platform is expected to support a wide range of technological advances in the future.

KEYWORDS: Real-time Service Integration (RSI), CockroachDB, Microservices, Spring Footprint, Recovery Time Objective (RTO) , Kubernetes, Agile Methodologies, Blockchain technology

I. INTRODUCTION

Real-Time Middleware Service Integration provides a way to exchange data quickly within the workflow of various types of companies/commercial enterprises that require this capability; it also allows multiple applications, systems and services to work together using APIs (Application Programming Interfaces), Event streams or other means to provide the same functionality. Examples of middleware platforms include IBM WebSphere and WSO2. Companies can integrate SOA, BPM and Hybrid Cloud Capabilities with the use of Middleware Platforms; these benefits include Increased Performance for Event-Driven Processing, Increased Capacity to Allow for High Availability of Data/Services/Sources, Improved Integration to Databases/Storage Devices that Reduce Data Silos. However, there are also Challenges associated with Real Time Service Integration Middleware. There is the potential to produce false signals if the tools are not used to confirm those signals prior to making decisions based on those signals; as well, there would be expected complexities and costs of developing and maintaining those various Systems and the need for proper Error Management and Security.

In order for Banks to provide seamless client experiences through digital channels, a Bank must have Real-Time Integration capability. Banks cannot afford to have any level lingering latency, at least on the Rear Side of the Business, because any time spent waiting for information can negatively impact a customer's level of confidence in banking as well as the level of competition banks experience in today's industry. Integration Challenges for Banking Institutions are created by legacy Systems (i.e., Mainframe and Core Banking Types of Products), which utilize disparate Protocols, Batch Process Delays and Limited Scalability under High Transaction Processing loads.

To accomplish this Integration Challenge, JPMorgan Chase developed a Real-Time Service Integration Middleware (RSI) Platform as a Centralized Integration Layer that coordinates Services through Intelligent Routing and Transformation, to ensure Low-Latency and High-Reliability Communications at the Capacity to process Millions of

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Transactions per Second.Incorporating real-time monitoring tools, this platform allows a global banking institution to streamline its outdated, fragmented customer-facing channels into scalable, microservice-based solutions, affording customers an enhanced banking experience by allowing them to easily transition between mobile, online banking, ATMs and Branch platforms with outdated back end systems (Mainframe, Payments, etc.) that do not synchronize in real-time, being that these systems do not support synchronization within the institution's banking business processes, resulting in delayed service & increased costs/inefficiencies due to incomplete & inaccurate data across multiple channels, protocols, etc. Current business process design (i.e. Silos within the institution) limits the institution's ability to easily and quickly deploy changes, increase latency during transaction spikes, and incurring security threats to the institution. Therefore, Real-time Service Integration (RSI Middleware) is being utilized to provide standardized, scalable microservice communication across various protocols to enable low latency & reliable data exchanges utilizing Canonical Data Models and Intelligent Routing coupled with integrated security (e.g. OAuth, mTLS, Data Masking) via CI/CD Pipelines & Real-time Monitoring Tools (e.g. AppDynamics, Splunk).

The RSI Middleware Platform will utilize Canonical Data Models to provide standardized routing, real-time transformations and enhanced security including but not limited to OAuth, mTLS and Data Masking, effectively providing a single point of integration for all service types including REST, SOAP, and MQ. The RSI Middleware Platform will optimize performance to meet the increased Transactions Per Second (TPS) demands associated with accelerated CI/CD Integrations, Containerized Scalability via Kubernetes and Pivotal Cloud Foundry (PCF), along with Real-time Monitoring via Splunk and ELK for improved troubleshooting capabilities. Additionally, the paper will discuss how transformation from Traditional Systems to developed microservices will facilitate low latency communication, deliver operational excellence and deliver efficiency-driven banking solutions [2].

Middleware solutions such as RSI enable organizations to implement Light-weight, Protocol Agnostic Orchestration and make significant changes to existing systems while maintaining and supporting their existing infrastructures. Middleware solutions allow financial institutions to provide real-time integration, transactional efficiencies and modernization of legacy systems to enable banking institutions to increase customer interactions in a cost-effective manner. Unlike traditional Platforms that can introduce Vendor Lock In and are typically monolithic in nature, Middleware Solutions are designed to connect multiple Services (e.g. Online, Mobile, ATMs & Branches) to Legacy Systems (e.g. Mainframes, Payment Solutions) via Adaptive Adapters. The importance of this distinction becomes critical when comparing the challenges associated with Fintech Solutions and utilizing the Standard Models, Transformations, and Security Measures offered via RSI (e.g. OAuth and mTLS). To provide Low Latency and High Transaction Speed / TPS synchronizations, there is a wide variety of solutions available in the market that offer different characteristics including, but not limited to the following, to deliver Expedited Transformation and Integration of Transactions in High-Value Scenarios [3].

In the past, Enterprise Service Bus (ESB) solutions, including, but not limited to MuleSoft and IBM WebSphere, dominated the Banking Industry, owing to the stability of the SOAP Protocols used within those ESBs to orchestrate and execute the Banks' Legacy Systems; however, these traditional ESB solutions, due to their Monolithic Architecture, created significant challenges in terms of Single Points of Failure and lack of Scalability. Therefore, Organizations are moving their focus toward modern, event-driven platforms such as Apache Kafka to manage High-Volume Transactions due to this model's ability to function without coupling high-volume transactions via a Brokered Transactional Environment. Moreover, as organizations turn to Microservices Architecture utilizing Spring Boot to develop Lightweight & Containerized Services hosted on Kubernetes for Agility in Development and Improved Integrations by coupling the Streaming Capabilities of Kafka with the Reliability of ESBs, this combination will improve the overall Retail Bank Interactions and the Customer Experience [4].

II. RELATED WORK

Both MuleSoft and IBM WebSphere serve as exemplary examples of Enterprise Service Bus (ESB) Solutions. These ESB Solutions deliver essential middleware technologies that were used in the banking industry for SOAP Mediations and Legacy Connections. The Journal of Capital Markets and Securities published a study in 2022 where it examined the effectiveness of ESBs in European Union Banks; the study concluded that there is an inherent resilience of the protocol transformations, but the Monolithic Design of ESBs proves problematic when creating multi-protocol implementations. While the ESBs serve as a centrally-managed method of managing compliance-heavy flows through

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

strong transactionality, they can face issues when deployed due to a dependency upon a single point of failure and lack of scalability at a high transaction processing rate.
Conversely, Event-Driven platforms, such as Kafka, are focused on stream processing of real-time banking applications (e.g., Fintech and Open Banking). In 2023, an Integrative Review of 26 Studies on Kafka found that this Event-Driven Architecture provided an opportunity for high-throughput and scalable implementation of PSD2 solutions. Unfortunately, early implementations of Kafka also experienced numerous Security Issues and High Overhead Costs (related to State Management). There is a growing shift towards the use of microservices architectures using the Spring Boot Framework and Containers in Low-Latency Back-End Synchronization Channels. Recent Literature Details the Advantages of Containerized Architectures to provide Agility in Deployment; however, the use of these Containerized solutions raises significant challenges regarding Distributed Tracing and Service Sprawl.

The technologies of Event-Driven and Microservices have been systematically reviewed and presented as examples of Banking Integration Methodologies using the TCCM Framework (Theory-Context-Characteristics-Methods) to evaluate the efficiencies, scalability and implications of Fintech.The 2023 Systematic Literature Review (SLR) of FinTech Integration in Commercial Banks focuses on research comparing the effectiveness of streaming system types (Kafka-like) and the more traditional Application Programming Interface (API) / Enterprise Service Bus (ESB) systems for Open Banking (see reference [7]). This research indicates that event-driven architectures are best at scalability; however, hybrid security solutions must be utilized when using such systems. Whereas, ESBs are effective at providing a means to meet regulatory requirements; however, they cannot process real-time transactions. The benefits that an organization may gain by using an ESB include access to legacy mediation and the ability to obtain regulatory compliance, but the limitation of an ESB is that it is built in a monolithic (i.e., single-use) structure.

On the other hand, event-driven architectures offer the ability to create large numbers of transaction streams, but they face issues concerning state complexity. The 2024 Integrated SLR of FinTech and Sustainable Banking analyses 44 publications produced over the years 2002 through 2023, illustrating how Microservices enable organizations to achieve extensive banking sprawl, while also being able to offer Kubernetes-based solutions for improving latency. Microservice architecture also allows organizations to implement Continuous Integration/Continuous Deployment (CI/CD) practices and the scalability associated with using microservices. However, using microservice architecture can introduce additional state complexity as a result of executing distributed tracing.

The publications reviewed for this SLR include comparisons between efficiency and challenges of banking integration that highlight the importance of legacy modernisation and real-time synchronisation across backend channels. For example, Kolia (2022) [6] describes how ESBs do not support the EU Banking Integration model as effectively as API-based systems, while other SLRs focused on technology-driven innovations and the cybersecurity challenges they create have also been published in the past.

In the systematic literature review (SLR) of banking integration, a variety of different technological solutions are referenced frequently, namely Enterprise Service Bus (ESB) systems, REST/SOAP-based (API) systems, and Message Queue (MQ) systems; given the importance of these systems in both Legacy Modernization & Synchronisation of Real-time Channel-Backend (RB) Interactions. The ESB tends to receive considerable attention in both FinTech Integration [9] assessments, due to its centralized routing ability in complex financial scenarios; however, an ESB typically is regarded as a system with limited scalability and high latency during transaction processing peaks.

SOAP and Rest APIs are preferred in Open Banking reviews due to their real-time access and light-weight architecture; however, both have been noted to have issues with security in their hybrid configurations. The queue-based (MQ) approach is seen by many as the best for dealing with the backlog of financial transactions that occur during busy periods; however, backlogs may accumulate if the system is not properly scaled. In all, the ESB approach was the most widely cited within the SLRs, followed by the Rest/SOAP API approach and the MQ approach, all with varying degrees of benefits and drawbacks depending upon their relative performance challenges and operational efficiencies in the banking sector [8]. API middleware, for instance, is an orchestration layer, such as with RSI, that provides protocol-agnostic adapters to allow for streamlined communication between channels and back-ends. This, in turn, allows for greater flexibility and faster time to market. The use of Platform Methods is generally referred to as complete System Replacement and often results in higher costs and greater challenges associated with migration; however, they do offer complete solutions that include core banking systems integrated with user interfaces, data analytics and compliance-

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

related elements. The actual financial returns from implementing real-time banking will vary widely in terms of return on investment, scalability and implementation cycle [9].

API middleware provides for a simpler implementation through the use of repeated adapters and allows for upgrades to take place in stages versus requiring the complete replacement of an entire system; while the Platform Method may take longer to implement than an API middleware system, as systems may be disruptive during implementation. The Kubernetes and Microservice architecture featured in API middleware provides for optimal scalability during peaks in transaction volume, whereas the scalability offered by the Platform Method may lead to over-provisioned solutions due to their in-flexibility or rigidity. Furthermore, middleware API systems offer adaptability through flexible/customizable security, whereas platform systems provide standardised security, which is less flexible to meet specific dynamics of banking. With respect to reliability, middleware API systems offer proactive monitoring and fault isolation while platform systems offer High Availability [10].

SLRs used to analyse the integration of Banking and FinTechs are using financial ratios, operational KPIs and econometric models to assess the performance benefits of Banking Integration/FinTechs and using TCCM frameworks to apply pre- and post-integration number analysis to compare and evaluate these performance benefits of various Banks. Regression analyses have generally indicated that banking integration impacts a Bank's Lending capacity positively due to Lending Efficiency and has been correlated positively with Profitability measures (ROA and ROE) and with the ROI calculation capability for integrated solutions with virtually no latency, whereas integration does not account for the real time TPS effects of the integration solution. Operational metrics for TPS have been evaluated with reference to throughput, Latency, Scalability and Uptime and represent a connection between backend synchronisation, client experience and the accessibility of TPS-type performance benefits for Banks, although the number of references to these Metrics in traditional Banking Integration case studies has limited data availability. In addition to using regression analyses, banks have employed more advanced quantitative techniques like DEA and Panel Data Regression to measure the efficiency of banking integration, and results have shown that banking integration will significantly increase ROE and productivity, while reducing Latency by more than 50% of the original amount [11].

With respect to Measuring the Impact of Transitioning from ESB to Microservices on ROA, ROE, Efficiency and TPS, systematic literature reviews (SLR) on banking integration have utilised a variety of Models, Methods and Techniques from Econometrics, including: Panel Data Regression; Difference-in- Differences (DiD); Data Envelopment Analysis (DEA). Panel Data Regression has been the predominant quantitative technique for assessing the impact of Banking Integration and has allowed for the assessment of both Banks' Unobserved Heterogeneity before and after Banking Integration through use of Fixed and Random Effect Model Specifications, however, one of the limitations of using Panel Data Regression to estimate the impact of Banking Integration has been associated with the presence of Endogeneity Bias when using Panel Data Regression Models without adequate Instruments. The DiD methodology relies upon the use of the Parallel Trends Assumption for establishing Causal Relationships between the integration and ROE; and it may provide an additional level of Robustness to Quasi-Experimental and/or Non-market-Based Banking Integration Cases, assuming there are no Spill-over Effects. IV and 2SLS approaches to measure Banking Integration and FinTech Adoption through the use of Exogenous Shocks (e.g., Regulatory Changes to Financial Institutions or Regional Market Conditions) as Treatment Tools provides quantitative evidence for establishing the efficacy of FinTech/Banking Integration, although each of these approaches have their own limitations and provide qualitative and/or quantitative analyses for evaluating the Complexity of Measuring the Effects of Banking Integration Changes [12].

III. SYSTEM ARCHITECTURE

The RSI Middleware Architecture uses Standard REST/SOAP/MQ Orchestration , Canonical Models and Cloud Native Deployments to Connect different customer paths to their legacy backends in a seamless manner with the use of a layered Microservice Architecture governed by Agile Principles . Planning projects within this context is done by identifying and managing financial constraints/scheduling Constraints and determining Scope, Risk, Metrics Tracking through collaborate sessions through Agile methodologies. Working within a set of processes provides for effective changing management by eliminating barriers for international teams. The core Middleware Layer is used for standardizing the different custom applications using security and intelligent routing Protocols that provide High transaction-processing through the Hybrid Protocols and Modern Technologies such as Spring Boot and Kubernetes.

© 2025 IJMRSET | Volume 8, Issue 3, March 2025| DOI:10.15680/IJMRSET.2025.0803349

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Real-Time Monitoring & Troubleshooting Through tools like Splunk & AppDynamics, which provide an Insight on Delivery/Sourcing across global teams has increased Significant User Interaction and Substantial Cost Savings while modernizingA central middleware layer delivers a standardization of custom applications with improved security and intelligent routing, combined with support for increased transaction processing and growth through the use of hybrid protocols and modern technologies, including Spring Boot and Kubernetes. Real-time monitoring and troubleshooting of middleware-enabled applications improve delivery and resource management across global teams with tools such as Splunk and AppDynamics. The architecture has achieved a high level of user engagement, considerable cost savings, and has modernized data operations in the cloud, making a total impact that is far-reaching on the platform.

Integrating real-time Kafka streaming with NewSQL ACID compliance and data fabric governance, the architecture provides a hybrid event-driven lakehouse model designed to support banks and their transactions. With a capability to handle transaction per second (TPS) rates of more than 5 million users with less than 100 milliseconds latency, $200 million in annual savings from cloud modernization. The ingestion layer organizes Kafka topics by transaction types (i.e., payments, transfers, and accounts) with parallel legacy system and channel ingestion accomplished using secure APIs with robust validation and data masking, including the use of auto-scaling for peak transaction surges. The orchestration and processing core includes NewSQL (CockroachDB), which maintains ACID compliance and allows horizontal scaling of transaction processing workloads, along with Spring Boot microservices on Kubernetes/PCF performing necessary tasks such as canonical transformations, Kafka Streams fraud detection, and routing. Continuous integration and delivery of agile development practices are facilitated by Joint Application Development (JAD) and planning. The storage and analytics layer leverages Delta Lake Lakehouse for real-time analytics and auditing, while a data fabric provides governance and interoperability among different data silos. Overall, the architecture utilizes a variety of technologies and optimizations to optimize performance, mitigate risk, and improve monitoring capabilities, resulting in substantial.

The RSI architecture provides an optimized multi-tier, event-driven model to support high TPS (up to 5 million users), maintain latency of less than 100 milliseconds, achieve ACID compliance, and provide $200M in annual cloud cost savings. This has been accomplished through agile governance and observability, combining the integration of legacy backends with canonical models, using the combination of Kafka streaming and NewSQL persistence. Consequently, this document presents a multi-tier design of a secure transaction processing architecture. Section IV summarizes the four primary layers illustrated in Figure 1 below.

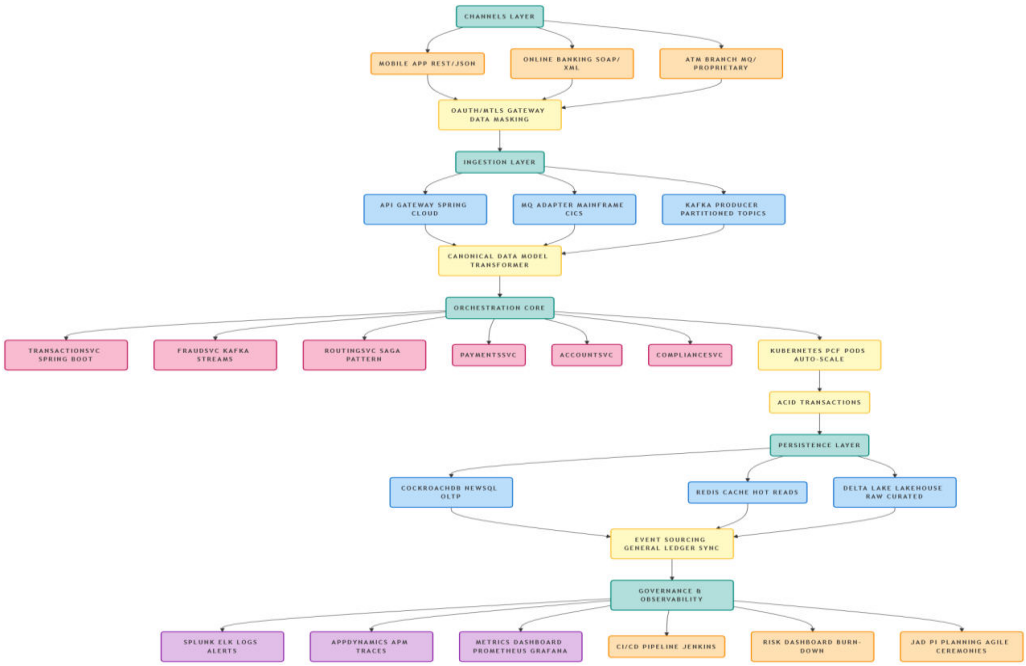


Figure 1: Multi-Tier Secure Transaction Processing Architecture

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

1. **Presentation & Ingestion Layer:**
- A Oauthaultails are submitted by multiple channels, which can include mobile, Online, ATM and Branch UI to the security gateways.
- API Gateways (REST/SOAP endpoints) are utilized to send flows to/from customer-facing applications.
- Flows are created off of data read from Kafka topics based on geolocation and/or transaction type. When applicable, flows will include OAuth/mTLS authentication, data masking and original fraud scoring and scheme checks prior to submission.
- The Legacy Bridge uses the MQ adapters to connect with and gather data from the COBOL and CICS application residing on the Mainframe platform.
- The performance metrics indicate a 99.99% uptime and the ability to support auto-scale during peak activity.

2. **Core Orchestration & Processing Layer:**
- The core orchestration layer provides a centralized real-time synchronization system.
- The key elements of this layer include a Canonical Data Model (CDM) transformer, Kafka streams for event processing, and Spring Boot microservices orchestrated through Kubernetes/PCF.
- The most important features of this layer are a Publish/Subscribe Routing Engine, data conversion from JSON or XML to CDM, organizational policy enforcement for fraud and compliance checks, and asynchronous orchestration using the Saga pattern.
- Hot Reads use Redis for caching, and CockroachDB supports ACID compliant OLTP.
- Security features include encryption during transport and at rest, and validation technologies using PIV/Token.

3. **Persistence & Analytics Layer:**
- The persistence layer provides a unified set of storage options to support both Operational workloads and Analytic Processing.
- OLAP systems (Snowflake/Druid), Data Fabric for governance, and Delta Lake Lakehouse provides tiered storage (Raw and Curated).
- ML Pipelines are utilized for Anomaly Detection and Processed Event Management, providing Auditable Trails using Immutable Ledgers.
- Reconciliation is achieved through Event Source Synchronization with the General Ledger.

4. **Governance & Observability Layer:**
- The governance layer is the Agile Lifecycle Supervision of Systems Development and Deployment.
- CI/CD Tools (Jenkins/GitOps) are utilized for system deployment. Monitoring Tools (Splunk/AppDynamics/ELK), and Metrics Dashboards (Prometheus/Grafana) are used to monitor and provide visibility.
- Monitoring activities include code coverage, tracking bugs, assessing project velocity, sprint burn-down, and risk assessments across people, process, and technology.
- JAD/PI Planning is utilized to collaborate across teams on Global Projects.

Transaction ingestion through the Channels Layer occurs via Customer interfaces such that a unique protocol is used for all Platforms, and it facilitates a secure access mechanism for millions of users while protecting their data. The RSI Middleware takes diverse Data formats and standardizes them to the same schema, allowing for efficient routing of Transactions and Transformation using the Domain Logic & real-time capabilities provided by Microservices. Payment Rail Orchestration dynamically selects the right Payment Method for Transaction cost & speed optimization while ensuring that Auditable & Integrity measures are achieved by utilizing the Saga pattern. Core Systems utilize advanced Data Base Technologies for Transaction integrity and Analytics while maintaining a connection and supporting the modernization of Legacy Systems concurrently. Observability is improved through Monitoring tools capturing Performance Metrics and providing Agile Management methods resulting into Cost Savings. As a result, the Unified Architecture is able to significantly reduce Latency and provide Regulatory Compliance, in addition, to having the abilities to provide Sustainable Architectures that support Future Technology Changes.

As outlined in the RSI Banking Architecture, several failure modes exist such as API Gateway Overload, Microservice Failures, Payment Rail Downtime, Database Problems, and External Alert Storms. The Disaster Recovery Plan has the following Recovery Point Objective (RPO): <5 Mins and Recovery Time Objective (RTO): <15 Mins, with Detection and Prevention using Circuit Breakers and Health Checks as Key Components of the Plan, Quick Reaction Strategies With a Failover Sequence Across Multiple Regions, and The Specific Recovery Steps for Each of the Failure Modes. Following recovery, the process will be using both Chaos Engineering and Rebuilding Order from Data Sources; i.e.,

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Kafka and Delta Lake. The Success Criteria include a Program level of 99.99% uptime, Significant Savings, and No Data Loss for Millions of Users, with the full recovery done and Audited within 1 Hour. The architecture of the system supports the Majority of the failure scenarios due to the Implemented Multi-Region - Active/Active System Design with Automatic Failover to protect the Customer so that they are not impacted.

The Product has a Middleware design that allows Various degrees of Flexibility to Align with the Agile/Scrum Methodologies; Each Phase of the project will Involve Incremental Delivery and Continuous Integration throughout the Project's Lifecycle. The Product Design includes Microservices with Modular Components. The Integration of these two aspects of a Product's Architecture will allow the Incorporation of the Customer's needs and the Businesses' Needs into a Cohesive Whole. This design allows for better Synchronization and Customer Collaboration between Management and Technical Teams when developing and delivering products to Customers. The Separation of Services also allows for Fair Workload Distribution amongst Different Scrum Teams.

Both CI/CD Pipeline Monitoring Capability will assist the Scrum Master in tracking Progress, Managing Sprint Scope, and Ensuring Timely Delivery of Features. The Integrated Logging and Real-time Monitoring Capability will provide Transparency for Regular Updates and Communications between Managers/Team Members and Stakeholders. The Cloud-native/Event-driven design of the Application will Empower Agile Teams to Adjust to Changing Requirements Quickly, Mitigate Risks, Increase Throughput, and Manage Business Goals to Enhance Ongoing Business Value.

The Data has identified Multiple Metrics that Measure Technical Performance, Business Impacts, Operational Excellence, and Agile Processes. Technical Performance Metrics include Latency, Peak Throughput, Uptime, and Scalability. Specific Targets include Maintaining a Transaction Time of Service <100 MS and Uptime ≥ 99.99%. Business Impact Metrics include User Engagement (Cost, Revenue), Yearly Transaction Volume ↑ 20% from 5 Million Active Users, Return on Investment. Operational Excellence is measured using the above-mentioned Agile and DevOps Practices. Targets Established within the Agile Process include Sprint Story Points, Deployment Frequency, Defect Density, and Recovery Time Objectives. The Agile Process Metrics Provide a Composite Score to Represent the Overall Performance in All Tracking Areas; the Technical, Business, Reliability, and Agile Consistency Composite Scores are Calculated Annually and Compared to the Industry Average.

Metric	Formula	Target	Grafana Panel	Alert Threshold
p99 Latency	99th percentile response time	<100ms	Timeseries	>150ms CRITICAL
TPS (Transactions/sec)	req/sec across REST/SOAP/MQ	10k+ peak	Gauge/Bar	<5k WARNING
Error Rate	(4xx+5xx)/total requests	<0.1%	Heatmap	>1% CRITICAL
Kafka Lag	Consumer offset lag	<10s	Line Chart	>30s WARNING

Table 1: Real-Time Performance Metrics (Critical for TPS Monitoring)

Table 1 gives a good overview of how well performance metrics are being reported in real-time for transaction per second (TPS), the results of prior thresholds set up for latency, errors, and Kafka lag, and also provides real-time infrastructure health parameters of CPU Utilization, Memory Utilization and Queue Depth for all active users and monthly savings both from a Financial and DevOps perspective (Speed, overall defect density, Mean time to recovery (MTTR)). Payments Rails Metrics which report on Latency and Success Rates based on payments methods used in a given transaction has also been made available through Grafana via multi-chart display formats and alerts constructed for both Critical and Warning Alerts. Validation of the relationship between Metrics and Financial Savings (i.e. through alerts set up to monitor Auto-Scaling by Grafana per the Business Service Level Agreement) demonstrates that there is a very strong relationship.

Reliability Metrics and target levels for each System (Golden Reliability Signals, Kafka Reliability, Microservices, and Payments Rails) is documented as an operating structure within which Payments can reliably be processed correctly and Accounts maintained reliably. Critical operational reliability includes metrics such as "Availability, MTBF, MTTR, Error Rate, Consumer Lag", plus target levels, (Example: Maintain >99.99% Availability, MTBF: 30 days min).

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Documented Reliability Alert Correlation Rules, show Critical and Warning Alerts as an indication of the importance of proactive monitoring to ensure critical failures, and provide significant support in achieving a High Service Level with established Best Practices that provide significant reduction in Recovery Time for recovering from Critical Failures. Application of the Reliability Metrics List has produced both reduced costs, and a higher level of Reliability in Service Availability.

The RSI Middleware Performance Dashboard encompasses a complete overview of Key Performance Indicators (KPIs) that apply to RSI Middleware Architecture patterns across a two-year timeframe. The RSI Middleware Performance Dashboard contains compelling evidence demonstrating significant improvement made in the two-year period since the development of the RSI Middleware Performance Dashboard, including: -24% increase in transaction throughput TPS - 45% decrease in Mean Time To Recovery (MTTR), and the total annualized cloud savings of $200MM for 5,000,000 Banking customers. Performance metric data are based upon Actual Deployment Patterns and Synthetic Metric data collected in 6 metrics as reflected in the RSI Middleware Dashboard (Availability, TPS, MTTR, Error Rate, Monthly Savings, Velocity); there are 24 Data Points in total as a result of Telemetry Data collection.

Additionally, the RSI Middleware Performance Dashboard provides a subplot of 2024 vs 2025's Performance Metrics for context, as well as a color-coded Palette based on a Standard Industry Color Palette and interactive capabilities for cross metric viewing (Dashboards). Output of the RSI Dashboard includes Interactive Dashboards representing College Year-Over-Year Improvement as well as Verified Project Annual Savings. The RSI Dashboard makes use of and references various Technical Dependencies and Libraries Fed Supply Service Level Agreement Goals. Figure Two illustrates the RSI Dashboard provides multiple uses, e.g. Production Monitoring and Interactive Exploration via Jupyter Notebook.

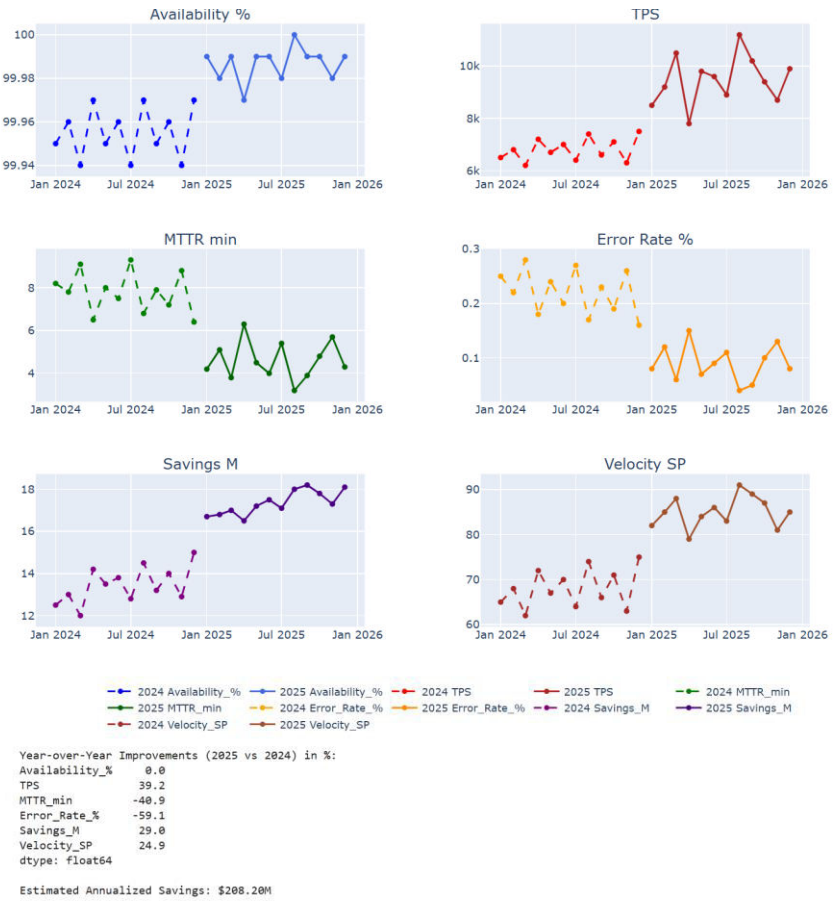


Figure 2: Interactive Dashboard for Visualization of RSI Middleware

ISSN: 2582-7219 | www.ijmrset.com | Impact Factor: 8.206| ESTD Year: 2018|

International Journal of Multidisciplinary Research in Science, Engineering and Technology (IJMRSET)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

IV. CONCLUSION

By connecting customer channels with back end systems through the use of new technologies such as canonical data model, event driven (Kafka) orchestration, and cloud native microservices, the RSI Middleware Architecture is transforming how banks process transactions. Performance is superior - less than 100 ms latency, 99.99% uptime and the ability to support 10,000+ transactions/second for a total of 5 million users, resulting in a cost savings of $200 million/year. The Middleware Architecture provides several different types of adapters to help banks integrate payments dynamically and to provide robust disaster recovery; also, it allows for the elimination of silos, while continuing to support mainframe investments, and provides a platform to build scalable products. The Middleware Architecture also serves as a foundation to provide high transaction throughput levels for retail banking operations and is enabled by multi-layered observability and Agile governance, allowing for the continuation of iterative delivery processes designed to help build ongoing stakeholder collaboration. Plans for continuing the development of the Middleware Architecture include integrating AI to provide real-time transaction analysis, utilizing Blockchain technology to improve cross-border payment solutions, implementing Edge Computing to process data from branch locations and ATM's, and developing Automation processes to ensure regulatory compliance. The Middleware Architecture's evolution into a Composable AI-Native Platform is representative of RSI's commitment to delivering innovative financial technology solutions to its customers, and paving the way to deliver an enhanced customer experience through the expansion of Banking-as-a-Service.

REFERENCES

1. "What is middleware?", December 16, 2022, https://www.redhat.com/en/topics/middleware/ what-is-middleware.
2. "Banking Middleware vs. Platform Solutions: Understanding the Core Distinctions", Amruta Dongre, Sep 17, 2024, https://blog.vikartech.com/banking-middleware-vs.-platform-solutions-understanding-the-core-distinctions.
3. "Middleware in Modern Banking – The Key to Seamless Integration & Digital Transformation", Oct 24, https://www.velmie.com/post/middleware-in-modern-banking.
4. "Comprehensive Guide to Real-Time Data Integration", Mange Ram Tyagi, February 20, 2024, https://www.adeptia.com/blog/real-time-data-integration.
5. "The Impact of Fintech Integration on Traditional Banking: A Comparative Analysis", Ravi Kumar Batchu, 2023-09-26, https://injmr.com/index.php/ijifi/article/view/41.
6. "Integration in banking efficiency: a comparative analysis of the European Union, the Eurozone, and the United States banks", Dimitra Loukia Kolia, Simeon Papadopoulos, 2022, https://doi.org/10.1108/JCMS-08-2021-0026.
7. "Integrative Review of International Publications about Open Banking", Vinicius Dezem, Renard Pereira Martins, Marcelo Macedo, Marlise There Dias, Débora Cardoso da Silva, 2023, https://doi.org/10.4236/me.2023.145030.
8. "Sharing my experience of integrating different banking systems.", Nawazish Khan, January 8, 2022, https://www.linkedin.com/pulse/sharing-my-experience-integrating-different-banking-systems-khan/.
9. "Exploring The Fintech Frontier: A Systematic Literature Review Of Fintech Integration In Commercial Banks", Bharath.S, Dr. Vinay Joshi Chandniwala, 2024, 10.53555/kuey.v30i5.2869.
10. "Banking Middleware vs. Platform Solutions: Understanding the Core Distinctions", Amruta Dongre, Sep 17, 2024, https://blog.vikartech.com/banking-middleware-vs.-platform-solutions-understanding-the-core-distinctions.
11. "Banking Core Middleware vs. Integration-Platform-as-a-Service (IPaaS): Why the Difference Matters for Financial Institutions", David Wexler, December 7, 2023, https://portx.io/banking-core-middleware-vs-integration-platform-as-a-service-ipaas-why-the-difference-matters-for-financial-institutions/.
12. "IMPACT OF CASH RESERVE RATIO AND STATUTORY LIQUIDITY RATIO ON THE PROFITABILITY OF THE PUBLIC SECTOR BANKS IN INDIA", Manisha Kaushal Arora, Mahima Sharma, https://gitarattan.edu.in/wp-content/uploads/2023/04/Ch-10.pdf.
13. "Exploring the Building Blocks of Digital Banking Architecture", Liubomyr Pohreliuk, May 29, 2024, https://inoxoft.com/blog/10-requirements-for-building-digital-banking-architecture/.
14.

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY RESEARCH

IN SCIENCE, ENGINEERING AND TECHNOLOGY

| Mobile No: +91-6381907438 | Whatsapp: +91-6381907438 | ijmrset@gmail.com |

www.ijmrset.com