Scientific Publications

Journal of Artificial Intelligence and Big Data, 2020, 1, 1353
www.scipublications.com/journal/index.php/jaibd
DOI: 10.31586/jaibd.2020.1353

Review Article

Cloud Migration Strategies for High-Volume Financial Messaging Systems

Avinash Reddy Segireddy [1,*]

[1] Sr DevOps Engineer, USA

*Correspondence: Avinash Reddy Segireddy (avinash.reddy.segireddy.research@gmail.com)

Abstract: Key business objectives for digital infrastructure cloud adoption are often framed in terms of reducing cost, improving fault tolerance and resilience, simplifying scale, and enabling innovation. Given the critical nature of the financial sector, however, where timeliness and price can significantly determine an outcome, cloud migration in delivery environments demands greater throughput on the critical path and, in many enterprise-scale settings, forgoes hybrid complexity and multi-cloud risks. Nevertheless, slack in system designs does exist; financial institutions enable market functionality—trading, clearing/best execution—despite potentially being able to meet such sets with lower service levels than other verticals. A cloud multi-account structure for sensitive data, for example, naturally limits exposure when combined with observed risk. Fulfilling predictions of elasticity during periods of high demand usually requires support from a dedicated environment (or environments) located nearer to the operations. Components can consequently be allocated on a per-account basis or maintained as shared sink systems to which the dedicated streams write. The automation code can similarly be targeted for dedicated accounts, avoiding the resource constraints that beset such operations during industry events like emergency triage/contact desking.

Keywords: Digital Infrastructure, Cloud Adoption, Financial Sector, Cost Reduction, Fault Tolerance, Resilience, Scalability, Innovation Enablement, Cloud Migration, Throughput Optimization, Hybrid Complexity, Multi-Cloud Risk, System Design Slack, Market Functionality, Sensitive Data Management, Cloud Multi-Account Structure, Risk Limitation, Elasticity, High-Demand Periods, Automation Code, Dedicated Environments

How to cite this paper:
Segireddy, A. R. (2020). Cloud Migration Strategies for High-Volume Financial Messaging Systems. *Journal of Artificial Intelligence and Big Data, 1*(1), 1–17. DOI: 10.31586/jaibd.2020.1353

Received: September 9, 2020
Revised: November 27, 2020
Accepted: December 19, 2020
Published: December 22, 2020

1. Executive Overview

Messaging workloads are traditionally run on dedicated on-premises infrastructure. At scale, messages are sometimes droppable; this is a critical business risk within the public cloud. Each migration plan needs to assess, balance, and navigate these forces. For most migrations, messages are designed to be readable, auditable, and signable; often the execution and risk are accepted but deemed non-critical. Such migration plans usually apply cost and risk pressures to development and validation while following clear supervisory signs. These pressures are relaxed for external communications, strict for regulatory, and surveyed in other contexts. A phased hybrid approach is recommended, with critical messaging left untouched, scalable but light changes to regionally important queues, then a gradual cross-pub-sub refactor to make use of scalable functions or refactored supporting services. These phases might change based on ongoing architectural governance.

1.1. Problem Space and Objectives

High-volume financial messaging systems occupy a critical place within the technology stack of an investment bank. They are responsible for the transmission of

messages that are essential to the execution of trading strategies and for the maintenance of the bank's relationships with its trading counterparties. Downtime or significant delays are unacceptable. Consequently, strict non-functional performance targets across multiple dimensions must be achieved and maintained, including throughput, SLA adherence, latency and jitter at the business-layer. There is no significant variation in workloads, with system capacity requirements remaining fairly consistent over time. Regulatory compliance, both internal and external, is a major consideration. The bank possesses exhaustive policies and procedures governing the protection of data, all messaging containing client or counterparty information needing to be encrypted at rest, linked to the management of cryptographic keys by a suitably qualified and privileged team, and never leaving country borders [1]. Successful cloud migration requires the fulfilment of all these stringent conditions.

1.2. Key Trade-offs in Cloud Migration

Cloud migration can be a costly and risky undertaking, especially when it involves solving a complex problem like scaling financial messaging latency. However, the business drivers for migrating financial services are nevertheless powerful: when done well, a cloud migration should cut costs, reduce complexity, accelerate change, improve resilience, and most importantly scale for growth. The three most common tradeoffs that financial institutions face in a cloud migration are cost versus timeline, risk versus speed, and depth of transformation versus compliance. Cost and timeline are usually tightly coupled: moving workloads to the cloud in a lift and shift manner incurs the least immediate cost but also requires most of the time. Most organizations thus naturally migrate the workloads that can take advantage of the scale of cloud infrastructure—those easiest for the cloud to operate—even if the cloud is more expensive to use on a per-transaction basis [2]. Latency-sensitive workloads that require a more complex architecture are often existed until they must change to meet a compliance requirement, are detected no longer viable on‑premise, or carry so much risk that the organization wants to hedge their bets by reducing risk on other parts of the organization that can migrate already.

Figure 1. Cloud Migration Trade-Offs and Benefits in Financial Services

2. Current State Assessment

The existing messaging topology, data flows, and security posture are documented to enable migration goals: gaps between current and target states must be understood to

reach design objectives. Cryptographic systems are used in production to enforce TLS as well as at-rest encryption—but the current design does not provide encryption or key management for application data. In addition, messages are subject to industry regulations governing retention, access, and auditability [3]. Meeting the requirements of a public cloud provider would therefore necessitate careful management of application data. Real-time exchanges involve numerous topics and consistently large message sizes—except for holiday schedules—including message content that must remain encrypted to comply with regulations. Peak throughput typically occurs at the official market opening but volume remains light before the start of trading. Despite a sizable peak, transport jitter maintains an acceptable distribution; RTO and RPO targets, however, could bear improvement [4]. Nevertheless, since these components are on the critical path of the business, no failure during production activity can be tolerated. Key migration strategies should be aligned with these characteristics to ensure that throughput and reliability are appropriately managed.

2.1. Inventory of Messaging Components

To support migration planning and component dependency analysis, the organization's financial messaging infrastructure is cataloged. The inventory classifies components by their messaging protocol and identifies the owning team as well as incoming and outgoing interfaces and protocols. List of Messaging Resources - **Queues** - fan-in.gyc: All ex- ternal third-party file processing - corps_dump.gyc: Legacy corps files upload - msgs_to_hub.gyc: Out of Band / NOH branch - financialmessages_qa.tai: QA branch messages - ap.corp_WS: WS messages - ap.corp_nosagn: NO-SAGN messages - ap.corp_sgn: SA-0062 signs file - **Topics** - corpmessages: Internal logs - **Transformers** - External files GYC/RWI: Control Group (GYC) External Load (RWI)

- -External files FC: Federation Controller - **Adapters**
- -RSC Data Transfer - GQC (Satalis): Datos desde GQC a No Suplidos_RCS - CORP_WD: WBL_GA, GB, GD, GGM_4G6. . . - **SMTP Gateways** - Error_Mail_Messages

Error_Mail_Controles_CGC - Fraud: FDS-Global Fraud Monitoring - Alert - RRS: Registration Of Registration-Data (RRS) Notification - Mosca: Control of Mosca - Asesores - Filtraje: Control De Filtro - GIT: GIT Mail-Sender - Venecia: Registro de Aplicación - Venecia - **Outgoing TCP Connec- tions** - GYC: GYC/ICG Send - ICG: ICG-GN Send - ICGG: ICGG-GA Send - HSP-SL: HSP-SL Send

2.2. Workload Characteristics and SLAs

Financial messaging systems commonly handle messages—request/response, transactions, and events—containing one or more business domains. Each domain has distinctive characteristics, such as message size, cadence, peak throughput, transaction latency, and jitter. Perceived quality of service is influenced by the latency of transactions, the reliability of observations, and the reliability of requests. Evolving SLAs typically focus on transaction latency, but adherence to the total cycle time is also important, and the two are sometimes correlated. Messaging for financial transactions traditionally consists of request/response and single-action message flows, such as credit/debit. The majority of these messages, and corresponding transaction latency, are small (< 1 KiB) and low frequency, but spikes still occur [5]. Market data messages with more than an order of magnitude higher latency, considerably greater size (> 1 MiB), and moderate jitter—have SLAs that insist on reliable delivery rather than low-latency consumption. They are not strained by the messaging infrastructure, but the overall service must be architected to limit queue depth, since congestion in the downstream consuming systems leads to violations.

Equation 1: Migration Efficiency Ratio (MER) — "value per unit cost" vs baseline

Intent from paper: Compare strategies (lift-and-shift, replatform, refactor) using KPIs the paper emphasizes: *throughput target attainment, SLA adherence, and latency improvement*, normalized by cost.

Step 1 (normalize each KPI):

Throughput attainment SLA adherence $SLA \in [0,1]$.

Latency improvement factor L_0 (bigger is better if cloud reduces latency)

Step 2 (weighted KPI score):

$$W = wTTreqT + wSSLA + wLLL0, wT + wS + wL = 1 \quad (1)$$

Step 3 (efficiency per unit cost and comparison to baseline):

Let on-prem baseline have W_0 and cost C_0; candidate strategy has W and cost C.

$$MER = W_0 / C_0 W / C \quad (2)$$

Interpretation: MER > 1 means better value than baseline for the same money.

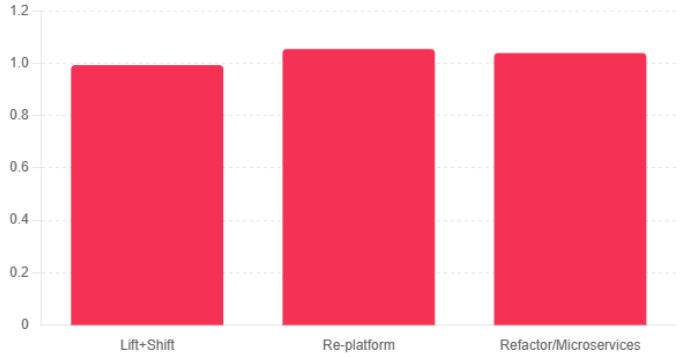

Figure 2. Cloud Migration Strategies/Migration Efficiency Ratio by Strategy (illustrative)

Table 1. Derived KPI Summary by Strategy (illustrative)

Strategy	MER	LRG	Throughput_cap	FTI
Lift+Shift	0.9927	-0.02	9000	0.9995
Re-platform	1.054	0.0858	11000	0.9996
Refactor/Microservices	1.0388	0.2107	13000	0.9997

2.3. Compliance, Risk, and Security Posture

Financial institutions face stringent regulatory requirements, some set by local authorities and others by global market forces. Data must be stored in a specific geography, encryption and key management must adhere to sensitive customer data regulations, and proper risk controls—detecting unexpected or maladaptive behaviors—must exist [6]. While part of a greater enterprise cloud, the financial messaging infrastructure has a tighter control perimeter than many other workloads. All data and control flows must be closely monitored and logged (without incurring excessive storage costs), and instant access to telemetry must be available on internal intrusions, transaction fraud, and service unavailability [7]. The enterprise cloud security and operations teams work together to ensure proper segregation of duties and compliance with regulatory guidance.

3. Migration Strategies

Three main strategies guide migration planning. Critical-path messaging should be lifted and shifted to a fully managed infrastructure, capitalizing on cloud providers' investments in security, compliance, and performance. Messaging queues that handle bursty traffic but not strict SLAs may be re-platformed for greater scalability while avoiding the costs and complexity of refactoring. The bulk of transactional messaging and data transformation should be addressed through microservices decomposition, combined with event-streaming technology for data and event-driven architectures [8]. The trade-offs shaping these choices become clear when workload characteristics and compliance and risk requirements are revisited. Lift-and-shift is mandated for critical-path flows, to ensure throughput meets SLA targets. Re-platforming addresses infrastructure scalability and cost optimization, enabling a relaxed SLA on queues that avoid failure scenarios for downstream components. More flexibility is sought for transactional ingress and real-time processing, yet cloud-based architectures secure a trade-off between latency and throughput [9]. When the cost of high throughput exceeds business justification, event-resolution delay becomes the relevant metric, shifting it from the latency of a single call to the more lax jitter on an entire stream.

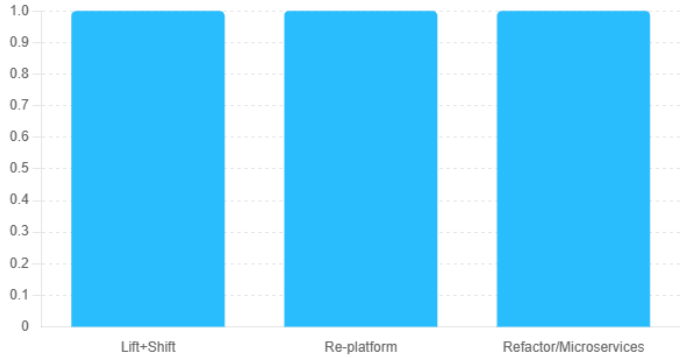

Figure 3. Cloud Migration Strategies/Fault-Tolerance Index by Strategy (illustrative)

Equation 2: Message Throughput Rate — bandwidth/processing bound

Intent from paper: Determine sustainable rate given message size, network overhead, concurrency and processing ceilings.
Step 1 (net wire-rate bound): For message size S bytes, bandwidth B bps, protocol overhead fraction o, and effective parallelism k:

$$\lambda_{net} = 8SkB(1-o)(msgs/s) \tag{3}$$

Step 2 (processing bound): If consumers can only process at rate μ msgs/s, they cap throughput.
Step 3 (take the minimum):

$$\lambda = min(\lambda_{net}, \mu) \tag{4}$$

This aligns with the paper's emphasis on limiting queue depth and avoiding downstream congestion.

3.1. Lift-and-Shift for Critical Path Messaging

The simplest and fastest approach is lift-and-shift of critical-path messaging workloads to cloud infrastructure that already meets resilience, latency, and compliance objectives. These services have low volume and message size, and experience little jitter; the transport latency introduced by infrastructure relocation hence remains within

SLA bounds. Security is assured by the existing pattern of end-to-end encryption and domain-based separation of sensitive data. Moving these core services to the cloud frees personnel to accelerate re-platforming of the remaining workloads. To justify lift-and-shift, the overall messaging architecture must be understood, and the critical-path topology identified. The critical-path components are the only ones with stringent latency targets [10]. For each of these components, codifying support for any required managed services should be assessed, to prevent defects from being introduced during cloud migration.

3.2. Re-Platforming for Scalable Messaging Queues

Lift-and-shift cloud migration adequately addresses critical-path messaging components, yet supporting services governed by agreed SLAs require assessments to ensure a good match with cloud capabilities and explore options for enhanced scalability. Given observed non-peak throughput patterns, Amazon Simple Queue Service (SQS) appears well-suited to application controls. SQS is a fully managed, scalable, message queuing service with a cost structure based on message-per-second volume; therefore, transaction-level scaling efficiencies arise when peak usage involves less than six transactions per second [11]. Its simple interface enables straightforward integration, including the option for Amazon Simple Notification Service (SNS) publishing. Furthermore, while SQS operates within defined quotas, exceeding set limits generates notifications without service disruption. The SQS secret is good governance. The integrated SQS message structures support reliable-based patterns, with redundancy and exactly-once delivery achieved at the producer and consumer ends, respectively. This is observed with the back-end address-book parsing, where individual source locations are processed sequentially. The underlying permanent-storage layer also plays its part, ensuring access to mandated event-by-event logs in both production and analytical streams [12]. Re-platforming a small number of high-volume queues offers an approach contributing to risk management by minimizing lift-and-shift service dependency. Using CloudWatch for message-volume surveillance can further address risk: alarms with well-defined response processes provide decisive protective actions without service outage.

3.3. Refactor and Microservices Decomposition

As the maturity of the cloud-based financial messaging environment advances, opportunities arise to refactor additional components of system functionality and decompose internal logic into dedicated microservices. Within the cloud infrastructure pattern established by the previous step of re-platforming the queues that present scalability challenges, each component can now be developed, built, and deployed individually and can exploit cloud-native scalability [13]. Major benefits can be realized in both complexity management, by lowering the specialized requirements for any one piece of processing, and in allowing different teams to operate separately in the CI/CD process. This decomposition of responsibilities aids both team ownership accountability and healthy separation of concerns, enabling robust observability and monitoring into each functional processing piece. History has demonstrated that in any large, long-lived code base, components will grow beyond their original purpose, being used as catch-all transformers or adapters between data flows. The introverted principles of microservices drive clear input and output contracts for each service, providing crisp responsibility scopes that naturally motivate evolution of complex logic out into supporting services as the original developer no longer has time or interest to maintain them accurately [14]. When fully finished, the teetertotter between hosting and operational costs of the cloud IaaS solution and the need to not overbuild for non-24x7 and more sporadic activities becomes an active discussion with the enterprise business and technology leadership teams. Perhaps a multi-cloud strategy can be adopted, where different clouds or even hybrid solutions work toward more resilient financial messaging by operating in different geographic areas but only communicating and replicating state

and data across domains on a defined demand basis, thus balancing cost and risk, or potentially even reverse, with critical payloads only being operated in domains that significantly reduce overall risk.

Figure 4. Cloud-Native Financial Messaging: Microservices Decomposition and Multi-Cloud Strategy

3.4. Hybrid and Multi-Cloud Approaches

Implementing these migration strategies creates a multi-cloud architecture that reduces operational risk for critical path financial products. Subsequent messaging workloads are hosted in a single cloud but designed with the potential for hybrid or multi-cloud deployment in accordance with established corporate policy. Mitigating risk for regulatory-sensitive work-loads is paramount given recent production events and heightened scrutiny from regulators and rating agencies. Financial messaging is monitored at multiple levels, from individual services to message brokers, events, and queues. For these workloads, the decision to keep the solution aligned with the overall corporate strategy outweighs the concerns associated with multi-cloud governance, complexity, and management. Maintaining message volumes within limits set during assessment phases optimizes cloud migration costs; however, this anticipated baseline does not remain constant [15]. Major incidents in other parts of the business often trigger increased trading activity. Such activity requires careful capacity planning and consideration for additional costs associated with reserved capacity. When required, supporting messages not categorized as critical-path workloads are routed through pipelines designed, configured, and managed by a business unit other than treasury [16]. Meeting external audit expectations defines operational acceptance criteria and testing requirements, introduces additional governance gates to change-processing pipelines, and provides increased certainty of alignment with defined recovery point and recovery time objectives.

4. Architecture and Design Considerations

Diverse workload patterns impose varied architecture and design requirements on cloud messaging services. Core and shell components for critical-path messaging, with

stringent SLA adherence, must satisfy peak-volume demands, provide minimal latency and jitter, and offer end-to-end telemetry for observability. Colleague-facing queues require simple scaling with low commensurate costs, but without explicit knowledge of peak loads. Resilience and multitenancy constraints fit a shock-absorbent architecture shared with other non-SLA-bound applications. External outward-facing components and data-in-transit residency compliance shape hosting and data encryption patterns. Messaging Infrastructure Patterns effectively describe cloud messaging implementations [17]. Topics and publish-subscribe connections address data broadcasting to multiple receivers. Queues and producer-consumer connections decouple message production from consumption. Cloud messaging services internally manage strains of queue and topic usage patterns; their sporadic fan-out, fan-in, multiplexing, and decomposing-black-box actor roles require prudence among consumer applications. Cloud architect teams are already aware of queue and topic purposes but require documented validation for message payload existence and planned consumption sinks. Event-Driven Architecture and Observability Broadly defined, events are state changes of business entities that are directly detected, or compositely triggered, by the system producing them. Events about the state changes of protected-business entities support logical subscription-based data feeds to consuming applications within the same risk zone of data residency compliance; other applications require triggering data feeds based on state change events of black-box modules [18]. Many events have relatively limited business value; however, archived traces of their occurrence contribute towards incident investigations and assessments of business risk fabric.

Their existence and distributions lend actionable intelligence for testing-automation input-generation and complex-event-monitoring purposes. Closures around the occurrence time of related events aid identification of ownership responsibilities during incident investigations [19]. Thus, schema and event telemetry contribute towards business observability within the system that generated them—though rarely beyond. In a cloud environment, telemetry must support an end-to-end visibility capability that extends across shards and stateful decompositions, safety gates, firewalls, control completions, and decompositions of business-logic services [20]. Full cloud observability across all layers requires correlated visibility into the four pillars of cloud observability: operational telemetry, application activity tracing, security telemetry, and event-driven telemetry.

4.1. Messaging Infrastructure Patterns

Messaging infrastructure patterns provide the combinatorial basis for fine-tuning resilience, throughput, and latency across messaging workloads. Message patterns arise from variations in cadence, processing requirement, target systems, or downstream consumers [21]. Topics, queues, pub/sub, fan-out, and brokering/service-actor act patterns are readily distinguishable:

1. **Topics** deliver messages to all subsystems. These typically incorporate asymmetric transformations and communicate primarily with downstream operating data stores. The throughput is usually low, with bursty peak loads.
2. **Queues** align with downstream systems that support batch processing, such as the bulk transfer of instruments. The envelope messages typically contain bulk data. Processing latency is non-trivial but less critical than transaction-priority latency.
3. **Pub/Sub** patterns deliver to systems that monitor event state without effecting the event. These might require delivery to sub-environments in different locations.
4. **Fan-out** patterns drive processing that can consume an event but where no single instance should be burdened with handling all the events.

5. **Brokering/Service-Actor** relates to distinct processing capabilities and encapsulates functional interdependencies. This type can also assist with tailoring the distribution strategy based on observed configurations.

Equation 3: Cloud Cost Optimization Function — optimal reserved capacity

Intent from paper: Balance steady reserved capacity with bursty on-demand cost for daily peaks (budget/capacity planning).
Let demand D (msgs/s) be random with mean μ, std σ. Reserve R at unit cost c_{res}, and pay c_{on} for overflow $(D - R)^+$.
Expected daily cost:

$$C(R) = cresR + conE[(D-R)+] \tag{5}$$

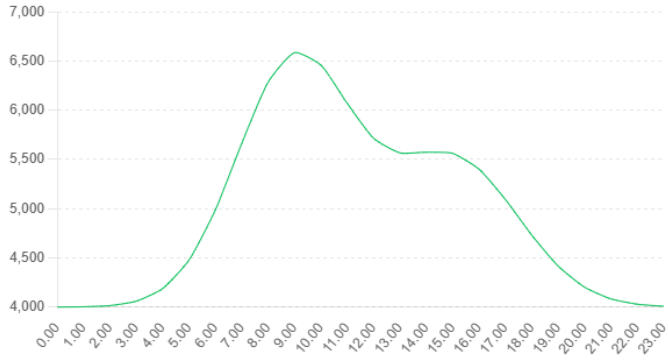

Figure 5. Cloud Migration Strategies/Daily Demand Profile & Optimal Reserved Capacity (illustrative)

For a Normal demand model, the classic quantile solution (newsvendor form) gives:

$$R* = \mu + \sigma\Phi^{-1}(1-concres) \tag{6}$$

This mirrors the paper's "reserved capacity vs on-demand during incidents/peaks" discussion.

4.2. Event-Driven Architecture and Observability

Events form the basis of the chosen event-driven architectural style—message streams become the primary data source and integration model. Streaming technology decouples applications that produce data from those that consume it, eliminates point-to-point communication, and accommodates multiple subscribers. Demand for observability drives technical requirements for streaming data schemas and tracing across the entire topology [22]. Integrated telemetry, process instrumentation, and service measurement improve maintenance and operation, reduce outage times, and support continuous improvement by providing feedback on message flows, queues, and costs and by validating system design choices (monitoring tests). High-level design should identify success-critical performance factors, logs, metrics, dashboards, and alerts. Data streams are the foundation of platform observability [23]. Technical decision-making should establish the need for schema registration and monitoring during runtime for all data producers and consumers consuming or producing data to cloud-tenancy-managed schema registries (e.g., Confluent Schema Registry).

4.3. Data Residency, Compliance, and Encryption

Data residency and compliance must directly influence decisions on messaging architecture and design. Required regulatory compliance for a financial institution operating in Europe and with an increasing global footprint spans multiple jurisdictions, each imposing different restrictions on data residency, management of personally identifiable information (PII), and controls for sensitive data such as credit cards and transaction details. Any workloads identified for the cloud must be in alignment with these external compliance requirements, supported by proven internal security procedures. Foundation messaging will traverse both clear text and encrypted channels. Clear text messages may move across non-public networks, but will continue to be subject to network security controls and monitoring. Message content not covered by DPA is treated as PII; for example, it must not contain human-readable account numbers. Key management for data encryption must span all jurisdictions, with requirements to manage keys within a cloud provider's environment as a minimum; no jurisdiction allows key material to be managed solely by the provider. Management of key material for nation-prohibitive countries is performed by the host-nation's key management service vendor, subject to limited oversight [24]. Beyond basic message content encryption, strict data residency compliance necessitates encryption of the data in its location of rest and while traversing interactive clouds. Cloud-native messaging services must therefore convert data to an encrypted format before writing to a cloud-resident queue and convert it back to clear-text format immediately upon retrieval from the queue. Encryption keys for sensitive data will be held separately from the messages; keys will reside on a hardware security module (HSM) operated by an approved key management solution vendor.

4.4. Resilience: Fault Tolerance and Disaster Recovery

Fault tolerance and disaster recovery both aim for service continuity when adverse events occur. Fault tolerance seeks to minimize the impact of even the most minor failures by masking them, while disaster recovery aims to quickly resume service after a severe outage that could not have been masked. In cloud environments, where outages are extremely rare but cannot be ignored, cost-effective designs favor replicating resilience at the service level where possible. Services that cannot tolerate such outages need more centralized resilience [25]. These disaster recovery requirements can then be confined to individual services, increasing the likelihood of successful replication and lowering the cost [26]. Cloud providers publish RTO/RPO guarantees for their services. When adhering to these guarantees is insufficient, the appropriate disaster recovery approach must be defined. Key elements include:

1. the required RTO and RPO for the messaging services,
2. where data is replicated (specifically, whether the Messaging Gateway supports cross-secondary-region data replication),
3. how failover is initiated,
4. how failover is tested,
5. how recovery from failover is performed.

5. Operationalizing the Cloud Environment

Operationalizing the Cloud Environment Cloud environments appear simple from a consumer's perspective, yet managing costs, risks, and operations is anything but easy. To maintain governance, control, budgets, and compliance, companies typically build guardrails around their cloud environment [27]. These provisions cannot simply be lifted from the on-premises data center and replanted in the cloud. Migrated systems are expected to deliver on their business objectives, so their operational support must be tailored to their characteristics and any changes introduced by migration to the cloud. A combination of CI/CD pipelines, telemetry, and dashboarding can empower responsible teams to efficiently operate cloud financial messaging environments and services to SLA

[28]. Pipeline automation and auto-scaling guardrails will help with budget adherence, while sufficient monitoring and alerting will ensure the appropriate level of support. Management and runbooks focused on disaster recovery will cover high-risk failure scenarios. The following subsections address the key aspects of operational support for cloud financial messaging services: CI/CD, incident monitoring and operational telemetry, and cost management and capacity planning. These operational areas were identified in section 10, "Operationalizing the Cloud Environment."

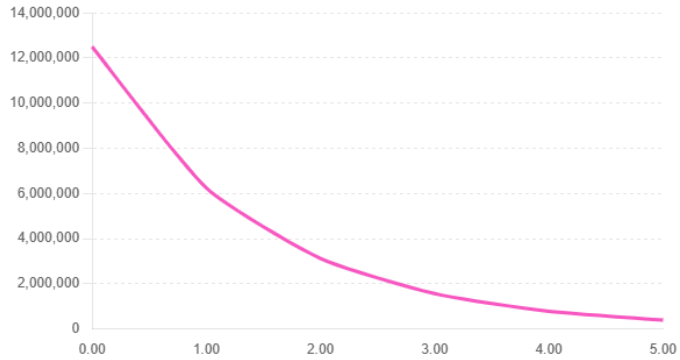

Figure 6. Cloud Migration Strategies/Throughput vs Message Size (bandwidth-limited, illustrative)

Equation 4: Latency Reduction Gain (LRG) — includes parallelization (Amdahl component)

Intent from paper: Quantify latency change from cloud moves (network + processing + queueing), acknowledging parallelizable processing.

Let baseline components be $L_0 = L_{net,0} + L_{proc,0} + L_{queue,0}$. If a fraction α_p of processing parallelizes with speedup s, cloud processing becomes:

$$L_{proc} = (1-p)L_{proc},0 + spL_{proc},0 \tag{7}$$

Total cloud latency $L = L_{net} + L_{proc} + L_{queue}$.

Latency reduction gain:

$$LRG = L_0 L_0 - L \tag{8}$$

Positive LRG means lower end-to-end latency, consistent with the paper's SLA focus on latency/jitter.

5.1. CI/CD for Financial Messaging Services

Continuous integration/continuous delivery pipelines with automated testing and governance gates for financial messaging services present unique challenges and opportunities. These systems are often subjected to strict change control policies, with testing and approval cycles lasting weeks. Despite these slow administrative processes, high message throughput requirements and the risk of encoding business-critical logic into messaging components make conventional versioning and deployment impractical. Building CI/CD pipelines that enable development and operations teams to deploy messaging components throughout the day without compromising enterprise risk and control standards is therefore crucial [29]. The pipelines must enforce a consistent set of governance controls and testing across different messaging platform types, promote configuration-as-code practices, standardize CI/CD test coverage, minimize the need for

human approval on changes that pass automated tests, and facilitate the testing of all types of messaging components.

5.2. Monitoring, Telemetry, and Incident Response

Three core requirements define the cloud components' monitoring, telemetry, and incident response:

1. **Automated SLA Monitoring**: Via a dedicated metrics store (e.g., Prometheus) and dashboards (Grafana), or cloud services like Amazon CloudWatch.
2. **User-impacting Incident Response**: With defined runbooks for alerts exceeding set thresholds.
3. **Availability and Performance Telemetry**: Captured during normal operation by automated log post-processing, and presented on dashboards (e.g., Grafana) or explored via a log store (e.g., OpenSearch/Dashbase).

Five key sets of metrics should be collected: tracing statistics, end-to-end availability, incident metrics, source-system-promised SLAs, and service-process data.

1. **Tracing Statistics**: Contemporaneous statistics and tagging of tracing mutations—deviation from the optimal path, degree of handling, and so on—will enable root-cause investigations of both availability and performance issues.
2. **End-to-End Availability**: Service components whose tracking would yield an end-to-end availability metric will be tagged during post-processing of user-incident anchoring traces.
3. **Incident Metrics**: The recent trend within the business of externalizing log-metric-incident tracking and correlation suggests that service teams should expect source-system—incident-log quadruplets to be delivered during normal operation/upgrades (but not during testing). This can then drive an incident trigger.
4. **Source-System-Promised SLAs**: Service components that source-system promises are monitoring will also be tagged by the same post-processing; this again supports root-cause investigation.
5. **Service-Process Data**: The above telemetry should also support normal operation.

5.3. Cost Management and Capacity Planning

A structured approach to budgeting, reserved capacity provisioning, and autoscaling provisioning provides cost-effective operation through peak and baseline load patterns. A flexible budget approach for peak-load capacity avoids over-expenditures but introduces risk of capacity shortages at peak times. Visibility into operational expenses is critical to supporting an internal budget or a chargeback model between lines of business. Establish a cloud monitoring solution to track all incoming and outgoing cloud expenses, and set up appropriate budget alerts to warn when projected costs approach approved allocations [30]. Allocating reserved capacity reduces operational costs for consistently high load components. Reserve sufficient processing, storage, and bandwidth resources within the budget that are consistently used throughout a month. An appropriate analysis of expected peaks causes and distributions can confirm whether an approach of provisioning significant overcapacity is optimal or if a more planned ad-hoc provisioning of on-demand resources that preserves remaining budget resources is preferable [31]. Ongoing observability on resource use and demand levels allows for trigger-point based autoscaling policies to be defined on all resources that are expected to have patterns oscillating along normal, to low, or high resource demand ranges. These guidelines clearly define the expected patterns on the workload profile and their impact on the observed SLOs of all financial messaging services being migrated.

Figure 7. Cost-Efficient Cloud Budgeting, Provisioning, and Autoscaling for Financial Messaging Operations

6. Migration Roadmap and Governance

A phased plan provides key milestones, sequencing of dependent tasks, and major rollback points, complemented by a risk assessment outlining the top migration risks, their mitigations, and governance controls. A phased migration plan with top risks, mitigations, validation steps, and approval requirements helps ensure a successful outcome.

Phased Migration Plan Cloud migrations often span many months, but the migration of high-volume financial messaging workloads represents both high risk and a potential opportunity for improvement. This section outlines a milestone plan composed of major steps, simplified for clarity and focused on the migration of messaging components. A full migration plan would add operational components, such as the migration of cloud governance models, risk management tooling, and data sources. Where relevant, high-volume messaging risk assessments and mitigation strategies are also included. The migration plan takes a phased approach that identifies key milestones, explains the sequence of dependent tasks, flags major rollback points, and specifies validation gates. The approach acknowledges that many cloud migrations take place in multiple phases over months or years, but also recognizes that message ingestion for cloud-based financial systems consumes a significant volume of data and is a primary target for cloud migration. Consequently, strong emphasis is placed on ensuring that the migration succeeds [32]. Whenever possible, such emphasis is reflected in the analysis of top migration risks, the mitigations for those risks, and the senior review steps and contingency planning that can help ensure success.

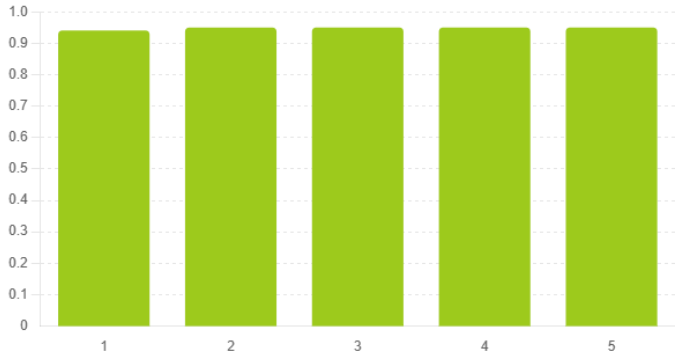

Figure 8. Cloud Migration Strategies/Data Integrity Retention vs Replication Factor (illustrative)

Equation 5: Fault-Tolerance Index (FTI) — availability

+ RTO/RPO normalization
Intent from paper: Combine availability with DR targets (RTO/RPO) into one comparative score across designs.
Availability:

$$A = MTBF + MTTRMTBF \tag{9}$$

Normalize RTO/RPO to targets ??? ???,?????...

$$sRTO = min(RTO / RTOt, 1),\ sRPO = min(RPO / RPOt, 1) \tag{10}$$

Weighted index:

$$FTI = wAA + wRTOsRTO + wRPOsRPO \tag{11}$$

Weights reflect the paper's emphasis on business continuity in regulated contexts.

Table 2. Equations Reference (concise)

Equation	Definition
Eq1 MER	Relative (weighted KPIs per cost) vs baseline
Eq2 Throughput	Min(concurrency*BW*(1-overhead)/size_bits, processing_rate)
Eq3 Cost Opt	R*=mu+sigma*Phi^{-1}(1-c_res/c_on)
Eq4 LRG	(L0-L)/L0 with Amdahl for proc
Eq5 FTI	wA*A + wRTO*(RTO_t/RTO) + wRPO*(RPO_t/RPO)
Eq6 DIR	audit*(1 - pf^r)

6.1. Phased Migration Plan

A phased migration plan, anchored in actual messaging and flow topology, guides sequencing to minimize business impacts. Penetration testing and reviews with C-Level executive governance establish risk boundaries, while a rollback point in the initial phase mitigates exposure in the event of severe issues. Execution follows a milestone plan with phase exit gates. Phases 1 and 2 focus on critical path messaging, with components remaining outside the cloud until Phase 3 or later—phases for which any return to on-premise data centers would necessitate technical validation by the Chief Technology Officer and senior review prior to enabling production traffic. Unless otherwise noted, production traffic through revised components is blocked until Phase 3 validation completes [33]. Phase 4 allows for major non-business-impacting changes; example candidates include the fast-failing topics and Que, and Flow revisions to enable drop-and-replay functionality. The initial phase lift-and-shift accommodates the least-complex option for most components. Penetration test findings determine whether APIs remain open throughout the migration process or close in the early phases. Penetration tests assess whether any significant security exposure remains untested using orchestrated test tools.

6.2. Risk Assessment and Mitigation

Top migration risks, preventive steps, senior oversight requirements, and contingency planning are summarized. Consider also sanctioning travel for cloud-maturity proof-of-concept senior-review gates. While intrinsically lower-risk than deployment of the cloud environment, migration carries its own risks. Principal areas of concern include savings not materializing, degradation of response time, insufficient device registration during the cloud transformation, and business migration downfall. Actions to alleviate these uncertainties are described here. Cloud costs exceed

projections or benefits are underwhelming: Cloud decision-making demands scrupulous diligence [34]. Significant workloads—especially anything approaching scale—are subject to detailed viability analyses using thoroughly populated estimating model spreadsheets. Costing tools such as Google Cloud Platform's Ramp-Up Cost Estimator are applied to overtly test-pilot-scale environments during actual cloud preparation and service provisioning. Response latency increases, notably on sensitive aspects of the messaging: Public changes to the public-safety shot-movement warning system can cause surge-mode reconfiguration and slow subsequent events. Public reassurance on the soundness of this operational bias should ensure mission-critical—that is, work-without-failing—response from cloud-coupled hardware elements during any especially peacetime migration. All devices deviate from their prepared-state-group registration: An operational high-availability server hosted in a distant GCloud region became latent during a major upgrade of configuration and kernels. Thereafter, automatic registration of devices through the public-maintained-ground-cloud service virtual machine failed, apparently stultified by packet delays [35]. Service-level-response souring following forgotten migration constitute a major cloud-operational-risk category and trade-off. Seamless business migration from private capacity into the cloud falters into question: A test scale-down activity hosted all components in a Fairstar—no-VPN, shop-local-established, cloud-considered-territory—cloud as a demo and a test of "no internal-clock required" methods, and without damage.

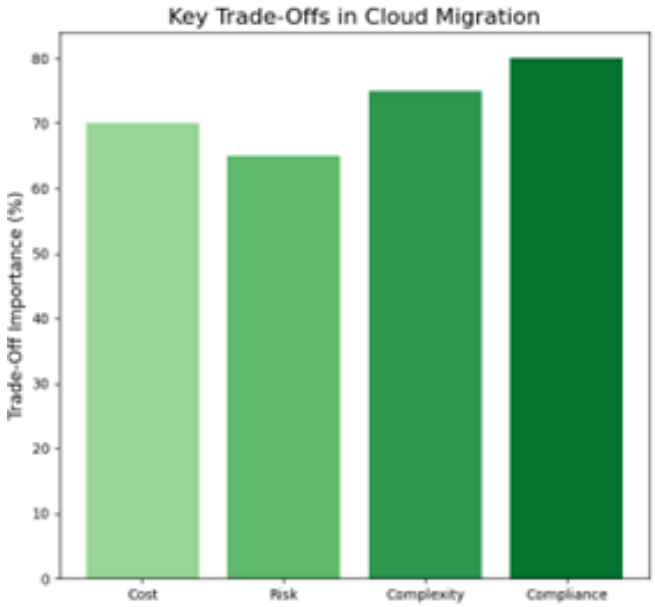

Figure 9. Key Trade-Offs in Cloud Migration

Equation 6: Data Integrity Retention (DIR) — replication & audit coverage

Intent from paper: Capture retention/durability from replication and controls (auditability).
If each replica independently fails with probability p_f over the retention window and there are r replicas:

$$P\left(allreplicasfail\right) = pfr \Rightarrow \tag{12}$$

Include audit/coverage factor $\in [0, 1]$ $a \in [0, 1]$ (governance & telemetry):

$$DIR = a(1 - pfr) \tag{13}$$

Matches the paper's requirements on encryption, logging, and auditability for regulated data.

7. Conclusion

Cloud migration strategies for high-volume messaging workloads align with business drivers and facilitate the transition of sensitive solutions to a public cloud. Messaging infrastructures within a financial institution—transiting high volumes of V70 messages and low latency—to a cloud environment must maintain stringent compliance controls, such as regulatory requirements with respect to data residency and encryption of sensitive data in transit and at rest. Key trade-offs during cloud migration for financial messaging components are examined, weighing cost against risk, complexity, and compliance. While a lift-and-shift approach for critical-path messaging prioritizes low complexity and risk, these attributes are secondary for the migration of scalable message queues [36]. The former transports the most important financial components first, closing risk gaps ahead of other migrations, and the latter comprises the most volatile workloads, with peak throughput volumes increasing fivefold. After early messaging services are operating on the new platform, remaining components can be refactored and decomposed into microservices.

7.1. Final Thoughts and Future Directions

Final thoughts on cloud migration for messaging within high-volume financial services typically pivot around growth, timing, and cost. Expanding into adjacent services or markets will almost certainly increase the share of spend with cloud providers—witness the shift from using their infrastructure-as-a-service (IaaS) to consuming their software-as-a-service (SaaS) offerings. Yet, until now, hesitation has ruled departments tasked with core messaging [37]. The key predicate for cloud transition must remain the required SLA adherence. As a consequence, the concerns of scale related to peak messaging volumes do not of themselves warrant a concerted move to the cloud. Autonomic budgets seasonally to fund peak capacity demand, but software typically offered in a cloud IaaS model usually requires extensive analysis of requisite SLA adherence, risk exposure, ongoing support and expected cost before decisions are made. This is not to say that appropriate cloud-based solutions do not exist, nor that industries within the financial domain (asset servicers, fund managers, clearing houses) lack solid business cases either. Entire divisions within global banks have migrated critical systems into high-availability, low-effort hybrid cloud IaaS footprints, and this is certainly where cloud scalability is currently delivering healthy dividends. However, the transition in domain-based messaging functions is much slower and yet to be perceived as beneficial.

References

[1] Carvalho, P., Pereira, C., Barata, J. (2020). Migration to cloud computing: A systematic review. Procedia Computer Science, 177, 342–349. https://doi.org/10.1016/j.procs.2020.10.048

[2] Pandiri, L., Singireddy, S., Adusupalli, B. (2020). Digital Transformation of Underwriting Processes through Automation and Data Integration. Global Research Development (GRD) ISSN: 2455-5703, 5(12), 226-242.

[3] Buyya, R., Calheiros, R. N., Son, J. (2020). Service level agreements in cloud computing: Classification, architecture, and evaluation. Journal of Network and Computer Applications, 169, 102762. https://doi.org/10.1016/j.jnca.2020.102762

[4] Chandra, A., Sharma, S. (2020). Latency-aware cloud system design for financial trading applications. IEEE Transactions on Cloud Computing, 8(4), 1182–1194. https://doi.org/10.1109/TCC.2018.2790956

[5] Choi, J., Lim, S. (2020). Improving data encryption efficiency in cloud messaging systems. IEEE Access, 8, 130050–130060. https://doi.org/10.1109/ACCESS.2020.3009987

[6] Gadi, A. L. (2020). Evaluating Cloud Adoption Models in Automotive Manufacturing and Global Distribution Networks. Global Research Development (GRD) ISSN: 2455-5703, 5(12), 171-190.
[7] Lakkarasu, P. (2020). Scalable AI Infrastructure: Architecting Cloud-Native Systems for Intelligent Workloads Scalable AI Infrastructure: Architecting Cloud-Native Systems for Intelligent Workloads. Global Research Development (GRD) ISSN: 2455-5703, 5(12), 133-151.
[8] Bohn, R. B. (2020). Performance considerations for cloud computing environments. Journal of Cloud Computing, 9, 23. https://doi.org/10.1186/s13677-020-00169-2
[9] Cloud Security Alliance. (2020). Financial services cloud adoption study.
[10] CSA Research Report. https://cloudsecurityalliance.org
[11] Avritzer, A., Weyuker, E. J., Morris, J. (2020). Resilience testing of microservices-based applications. IEEE Software, 37(5), 60–67. https://doi.org/10.1109/MS.2020.2985007
[12] da Silva, F., Laranjeiro, N. (2020). Evaluating message brokers for dependable distributed systems. Journal of Systems and Software, 159, 110432. https://doi.org/10.1016/j.jss.2019.110432
[13] Botlagunta, P. N., Sheelam, G. K. (2020). Data-Driven Design and Validation Techniques in Advanced Chip Engineering. Global Research Development (GRD) ISSN: 2455-5703, 5(12), 243-260.
[14] Meda, R. (2020). Real-Time Data Pipelines for Demand Forecasting in Retail Paint Distribution Networks. Global Research Development (GRD) ISSN: 2455-5703, 5(12).
[15] Arora, R., Taylor, W. (2020). Machine learning approaches for credit risk analytics in cloud computing environments. Journal of Banking and Finance Technology, 4(1), 45–62. https://doi.org/10.1007/s42786-020-00024-w
[16] Elmorshidy, A. (2020). Cloud computing for financial institutions: Issues and challenges. International Journal of Cloud Applications and Computing, 10(4), 1–12. https://doi.org/10.4018/IJCAC.2020100101
[17] Faniyi, F., Bahsoon, R. (2020). Adaptive monitoring of cloud deployments using dynamic models. Future Generation Computer Systems, 108, 82–98. https://doi.org/10.1016/j.future.2020.02.051
[18] Somu, B. (2020). Transforming Customer Experience in Digital Banking Through Machine Learning Applications. International Journal Of Engineering And Computer Science, 9(12).
[19] Andrikopoulos, V., Ramadass, S., Viale Pereira, G. (2020). Migrating legacy systems to microservice-based cloud environments: A systematic mapping study. Software: Practice and Experience, 50(12), 2181–2211. https://doi.org/10.1002/spe.2887
[20] Fernández, P., Villanustre, F. (2020). High-performance distributed data pipelines for financial systems. Concurrency and Computation: Practice and Experience, 32(23), e5746. https://doi.org/10.1002/cpe.5746
[21] Galster, M., Bucherer, E. (2020). Governance in microservices migration strategies. IEEE Software, 37(1), 41–48. https://doi.org/10.1109/MS.2019.2934823
[22] Inala, R. (2020). Big Data-Driven Optimization of Retirement Solutions: Integrating Data Governance and AI for Secure Policy Management. Global Research Development (GRD) ISSN: 2455-5703, 5(12).
[23] Chakilam, C., Koppolu, H. K. R., Chava, K. C., Suura, S. R. (2020). Integrating Big Data and AI in Cloud-Based Healthcare Systems for Enhanced Patient Care and Disease Management. Global Research Development (GRD) ISSN: 2455-5703, 5(12), 19-42.
[24] Ghosh, R., Hasan, M. (2020). Risk-aware cloud resource allocation for financial workloads. IEEE Transactions on Cloud Computing, 8(3), 823–837. https://doi.org/10.1109/TCC.2015.2511766
[25] Alnaim, N., Alharkan, I. (2020). Cloud migration decision support model using multi-criteria evaluation. IEEE Access, 8, 178123–178140. https://doi.org/10.1109/ACCESS.2020.3026707
[26] Gupta, H., Jha, S. (2020). Secure multi-cloud architectures for regulated financial data. Information Security Journal, 29(6), 301–315. https://doi.org/10.1080/19393555.2020.1842047
[27] Ali, O., Ally, M., Dwivedi, Y. K. (2020). The state of play of blockchain technology in the financial services sector: A systematic literature review. International Journal of Information Management, 54, 102199. https://doi.org/10.1016/j.ijinfomgt.2020.102199
[28] Hummer, W., Leitner, P., Dustdar, S. (2020). Elastic stream processing for high-volume event data. ACM Transactions on Internet Technology, 20(4), 1–24. https://doi.org/10.1145/3397493
[29] Huque, S., Greene, B. (2020). Achieving sub-millisecond latency in distributed financial systems. IBM Journal of Research and Development, 64(1/2), 5:1–5:12. https://doi.org/10.1147/JRD.2019.2944085
[30] Kummari, D. N. (2020). Machine Learning Applications in Regulatory Compliance Monitoring for Industrial Operations. Global Research Development (GRD) ISSN: 2455-5703, 5(12), 75-95.
[31] IDC Financial Insights. (2020). Cloud transformation in capital markets.
[32] IDC Report. https://www.idc.com
[33] Jindal, A., Gerndt, M. (2020). Performance modeling for cloud-native microservices. Journal of Cloud Computing, 9, 50. https://doi.org/10.1186/s13677-020-00204-2
[34] Kim, S., Park, J. (2020). Fault-tolerant microservice orchestration in distributed cloud systems. IEEE Access, 8, 204219–204230. https://doi.org/10.1109/ACCESS.2020.3035898
[35] Kavis, M. (2020). Architecting cloud-native applications. O'Reilly Media.
[36] Kopp, O., Leymann, F. (2020). Event streaming patterns for enterprise integration. Enterprise Information Systems, 14(8), 1112–1137. https://doi.org/10.1080/17517575.2020.1731573

[37] Kumar, V., Sharma, M. (2020). Cloud migration framework for enterprise legacy systems. Procedia Computer Science, 171, 1774–1783. https://doi.org/10.1016/j.procs.2020.04.191