Journal of Engineering, Management and Information Technology

ASPUR PUBLISHING

Vol. 05, No. 01 (2027) 1-8, doi:10.61552/JEMIT.2027.01.001 - http://jemit.aspur.rs

MIGRATING A LEGACY AD EXCHANGE TO A NEXT-GENERATION RTB BIDDER: A ZERO-REVENUE-DISRUPTION FRAMEWORK FOR LARGE-SCALE CTV PLATFORMS

Sujan Kumar Reddy Challa[1]

Received 18.07.2026.
Revised 17.08.2026.
Accepted 07.10.2026.

Keywords:

Ad Exchange Migration, Real-Time Bidding, Zero-Downtime Deployment, Canary Release, CTV Advertising, Migration Governance.

Original research



A B S T R A C T

Most large scale CTV platforms still have ad exchanges that existed prior to OpenRTB and VAST protocols that its demand partners expect it to support. The legacy exchanges were usually developed internally before the emergence of programmatic buying or acquired with the platform and are typically lacking in terms of ad units support, adherence to industry protocol, and the ability of the exchange to be observable in real-time by the modern bidder. Migrating from such a system while the platform continues to sell its inventory is not a standard deployment procedure: traffic to auction cannot be paused mid-deployment and any disruption immediately affects its revenue, typically in just a few hours. This paper discusses how this sort of migration actually should be done: starting with pre-migration readiness assessment in terms of functional, performance, and observability parity; continuing with traffic mirroring where the bidder can work with live traffic in the shadow mode; and ending with phased cutover with automated health-check gates for traffic increments and rollback capabilities. This study also goes beyond the technicalities involved and into the organizational aspects of such a process, which are often underestimated but are just as important as the technical aspects. The framework's migration stages, KPI gate thresholds, and rollback mechanisms are validated through a discrete-event simulation study on synthetic exchange telemetry (Section 4.4). It can be used as guidance in similar cases by organizations.

© 2027 Journal of Engineering, Management and Information Technology

1. INTRODUCTION

1.1 Background

Free ad-supported streaming television moved from a fringe distribution model to a core business line within roughly a decade, and several major consumer electronics manufacturers now run their own FAST platforms as a primary revenue source rather than a side project. The advertising infrastructure underneath many of these platforms, though, was often built earlier, sometimes assembled during the pre-programmatic era, sometimes acquired wholesale along with a platform or content library. This is a familiar pattern in software generally, not unique to advertising: research on legacy information systems has long observed that infrastructure originally built to solve a narrower problem tends to persist well past the point where it fits the system around it, simply because replacing it is expensive and risky enough that deferral tends to be the path of least

[1] Corresponding author: Sujan Kumar Reddy Challa
Email: sujan.challa10@gmail.com

resistance in any given quarter (Bisbal et al., 1999). Demand-side platforms have since converged on OpenRTB 2.x and VAST 4.x as baseline expectations, with OpenRTB 3.0 and VAST 4.2 increasingly requested for CTV-specific creative formats. An exchange that predates these standards does not stop working; it just becomes progressively less attractive to the demand side, which is a slower and quieter problem than an outage but not a smaller one.

1.2 Problem Statement

Swapping out a live ad exchange is a different kind of problem from a typical service migration. The traffic cannot be queued and replayed later, because an auction that is not answered inside its window is an auction that is simply lost, not deferred. Game-theoretic treatments of real-time bidding make a related point from the economics side: shifting how an auction is run, even in ways that seem purely mechanical, changes advertiser and publisher incentives in ways that are easy to misjudge in advance, a reminder that a migration is not just a technical cutover but a change to the market the platform participates in Sayedi (2018). The fact that these platforms are selling impressions constantly means that every hour spent on migration takes place within the context of live money being generated. In addition, the team performing the migration is, for all intents and purposes, creating their strategy from scratch since they lack experience with migrations of this sort.

1.3 Purpose and Scope

What follows draws on established research in deployment safety, legacy-system migration, and RTB economics to lay out a framework for this specific kind of migration: legacy exchange to modern bidder, on a live CTV platform, without a maintenance window. It is worth stating plainly what this is not. It is not a single company's case study with headline numbers attached, and industry statistics that could not be independently verified, claimed CPM discounts, cost-overrun percentages, incident-rate comparisons, have been left out rather than repeated uncritically. The framework is meant to travel across CTV platforms regardless of which legacy exchange technology sits underneath, and Section 5.4 says more about where its evidence base is thinner than a reader might want.

1.4 Contribution and Scope

To be specific about what this paper contributes: it is a sequencing framework, not a new deployment tool or protocol. Nothing here requires a particular vendor's rollout platform, and the phase structure described in Section 3 should map onto whatever traffic-splitting and monitoring infrastructure a given platform already runs. That choice reflects how most teams actually approach this kind of project. They rarely have the time or mandate to build entirely new tooling before a migration; what they need is a defensible sequence of steps and a shared vocabulary for the gates between them. The intended audience is engineering and product leadership at CTV platforms sizing up a similar migration, not researchers looking for a controlled experiment.

2. BACKGROUND AND RELATED CONCEPTS

2.1 Why Legacy Exchanges Become Monetization Bottlenecks

Streaming-video advertising research has tracked this shift from the demand side: as programmatic buying matured, spending increasingly consolidated around platforms that could guarantee protocol compliance and consistent measurement, a pattern documented across the dozens of studies synthesized in a recent review of streaming-video advertising research (Frade et al., 2021). An exchange that cannot speak current OpenRTB extensions, or that only supports a narrow set of CTV ad unit types, does not get excluded outright, but it does get deprioritized in ways that are hard to see from inside the platform and easy to see from a demand partner's dashboard. Table 1 lays out the practical capability gap between a typical legacy exchange and an OpenRTB-compliant bidder.

2.2 Prior Research on Migration and Deployment Patterns

None of the individual pieces here are new. Sneed and Verhoef's experience report on a cost-constrained payroll-system migration lays out the same basic choice engineering teams face in this context: convert the legacy system incrementally, redevelop it from scratch, or reimplement it against a new architecture, each with a different risk and cost profile (Sneed & Verhoef, 2020). Systematic reviews of microservice deployment and communication patterns have catalogued the building blocks, three deployment approaches and seven communication patterns across the literature, that a migration like this typically draws from (Karabey Aksakalli et al., 2021). A separate mapping study of microservice architecting practice found much the same thing: the questions recur, but the specific answers depend heavily on context (Di Francesco et al., 2019).

2.3 Zero-Downtime Cutover as a General Pattern

Zero-downtime cutover patterns are also reasonably well-trodden ground outside advertising specifically. Work on schema evolution for continuously deployed databases describes much the same shadow-then-shift pattern this paper adapts for exchange migration: run the new path in parallel first, validate it against production data without committing to it, then move traffic over in increments with automated gates between them (de Jong et al. 2017). A review of blue-green, canary, and feature-

flag deployment techniques reaches a similar conclusion from a different angle, with canary deployment in particular trading some operational complexity for a materially lower blast radius when something goes wrong (Ramaswamy, 2024).

2.4 RTB Auctions as a Control Problem

It helps to think about why phased gating works at all, and control-theoretic treatments of programmatic advertising offer a useful frame here. Research applying feedback-control concepts to real-time bidding describes bid and budget decisions as a system that needs continuous correction against a moving target, rather than a one-time optimization (Karlsson, 2020). A migration's health-check gates are doing something structurally similar: measuring an output signal, bid response rate, fill rate, RPM, against a setpoint drawn from the legacy baseline, and correcting course, via rollback, when the signal drifts.

Table 1. Capability comparison between a typical legacy exchange and an OpenRTB-compliant bidder.

Aspect	Legacy Exchange	OpenRTB-Compliant Bidder
Protocol support	Pre-OpenRTB 2.x or partial compliance	OpenRTB 2.x / 3.0, VAST 4.x / 4.2
Ad unit format support	Limited, often pre-roll only	Full CTV range: pre-, mid-, post-roll, interactive
Observability	Minimal, often batch reporting	Real-time metrics and distributed traces
DSP integration breadth	Narrow, often bespoke integrations	Broad, standards-based integrations
Latency characteristics	Often untuned for RTB auction windows	Tuned for sub-100ms auction windows

Framed that way, automated gating is not a migration-specific invention so much as an application of a control loop to a one-time transition rather than to steady-state operation, which is part of why the pattern generalizes across the bid-optimization literature more broadly (Ou et al., 2024).

3. CORE DISCUSSION — MIGRATION FRAMEWORK

3.1 Pre-Migration Readiness: Parity Assessment and Acceptance Criteria

Migration of this sort is not started by migrating traffic but by validating that the newly added bidder is capable of accepting it. Readiness, in this framework, spans three dimensions. Functional parity means the new bidder supports every ad unit type, DSP integration, and auction configuration the legacy exchange currently handles, not most of them. Performance parity means the new bidder matches or beats the legacy system's latency and fill-rate metrics under load testing that reflects real traffic patterns, not a synthetic benchmark. Observability parity means the new bidder emits enough metrics and traces to catch regressions that the legacy system was quite possibly not instrumented to surface in the first place (Zhou et al., 2021). None of these are optional, and none of them are usually finished as early as a project plan assumes; the pre-migration phase tends to take longer than teams initially budget for, precisely because parity gaps surface as they are being tested rather than as they are being planned. Performance parity, in particular, is not just about the bidder's own code path: the DSP-facing API layer follows the same request-response contract that remote procedure call research has described for decades, where the transport and marshaling overhead sits between the application logic and the actual network round trip (Birrell & Nelson, 1984), and comparative benchmarking of RPC frameworks has found performance spreads wide enough between candidate implementations, one framework beating its nearest competitor by at least 20 percent in some tests, that the choice is worth measuring rather than assuming (Blinowski & Pełka, 2026). Table 2 lays out a readiness checklist across these three dimensions.

3.2 Traffic Mirroring Architecture and Shadow-Mode Validation

Once parity looks solid on paper, traffic mirroring is where it gets tested against reality. The idea is straightforward: duplicate live bid requests to the new bidder, let it process them fully, but do not let its responses touch the actual auction outcome. This shadow-mode operation means the new bidder sees genuine production traffic, including whatever load spikes, format mixes, and DSP availability patterns actually happen rather than whatever a test plan anticipated. The mirroring layer is generally found at the load balancer/API gateway and works to mirror requests without any added latency to the path which will decide how the auctions work. The question of how long the system should remain in shadow mode is not a question of time on the calendar but rather how much traffic it covers, including peak load traffic, week vs. weekend variance, and at least one cycle of the seasonality/promotional events of the site before traffic commitment. Figure 1 sketches this flow.

3.3 Phased Traffic Splitting and Automated Health-Check Gating

Once the shadow testing goes fine, the migration happens in phases as opposed to in one go and in phases such as 5%, 10%, 25%, 50% and then 100% with health-check gates between every phase. The health check gates will check KPIs that contribute to revenue, namely, bid response rate, auction fill rate, and RPM. For the bidder to be able to proceed to the next phase, it needs to

maintain all three of these metrics above their respective thresholds for a period of time, usually 24 or 48 hours, enough to prove that this is not just an anomaly for one particular day. If any of the KPIs fails to keep the minimum threshold, a rollback happens automatically, and the traffic will be sent back to the previous ratio, because it is precisely during the time it takes for an engineer to discover the issue that revenue is silently lost.

3.4 Migration Governance: The Organizational Work Alongside the Technical Work

Organizational aspect of such activity, in reality, is usually more difficult than technical, and the problem that occurs is that teams usually underestimate this aspect until they get right into it. Business stakeholders watch daily revenue reporting closely during a migration, and they will flag anything that looks like a deviation, including deviations that are just reporting-latency artifacts of the transition rather than real revenue impact. Preparing the migration for architectural review, and documenting the readiness criteria, rollout sequencing, and rollback design as part of that process, tends to produce a more defensible plan than one built informally under deadline pressure (Tyree & Akerman, 2005). Establishing a communication protocol up front, agreed dashboards, a defined update cadence, a clear escalation path for anomaly investigation, prevents a lot of the organizational noise that can otherwise derail a technically sound migration. A team that finishes this kind of migration with zero revenue disruption and stakeholder confidence intact has actually cleared two different bars, not one.

3.5 Cross-Functional Testing Before Cutover

One detail that is easy to skip under deadline pressure: readiness testing should not be purely an engineering exercise.

Table 2. Pre-migration readiness checklist across functional, performance, and observability parity.

Parity Dimension	Criterion	Pass / Fail
Functional	Supports all legacy ad unit types	
Functional	Supports all legacy DSP integrations	
Functional	Supports all legacy auction configurations	
Performance	Latency meets or beats legacy under load test	
Performance	Fill rate meets or beats legacy under load test	
Observability	Emits bid response rate, fill rate, and RPM metrics	
Observability	Traces available for auction-level debugging	

Finance and revenue-operations teams typically have their own reporting pipelines built against the legacy exchange's data shape, and those pipelines can silently break, or silently misreport, once the new bidder starts contributing an increasing share of traffic. Looping finance, ad operations, and account management into shadow-mode review, not just engineering and SRE, tends to surface these downstream dependencies before the first traffic increment rather than after a stakeholder notices a reporting discrepancy mid-migration. This is a small addition to the readiness checklist in Table 2, but it is the kind of gap that is cheap to catch early and expensive to catch late.

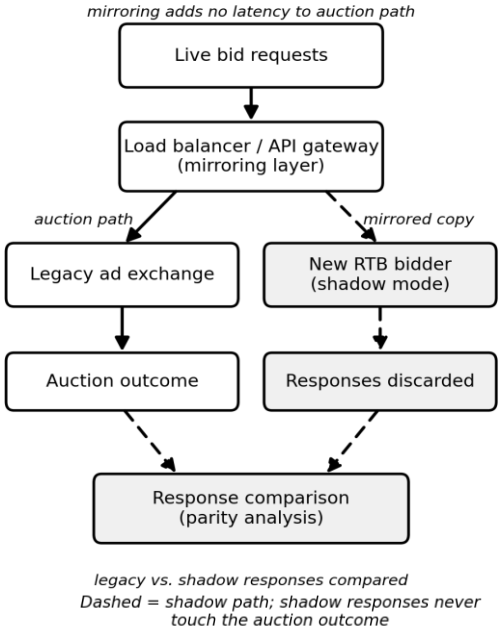

Figure 1. Shadow-mode validation flow

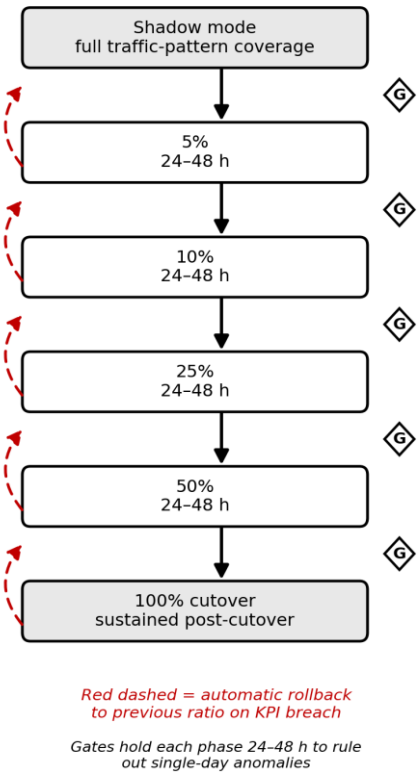

Figure 2. Migration phase timeline with gate and rollback indicators (Illustrative Image).

4. EVALUATION CONSIDERATIONS

4.1 KPI Selection and Gate Thresholds

Choosing what to gate on matters as much as choosing the threshold. Bid response rate, fill rate, and RPM cover the core commercial signal, but they should be tracked at a granularity that can actually catch a regression before it is platform-wide, segmented by DSP partner and ad unit type at minimum, since an aggregate metric can stay flat while a specific segment quietly degrades (Zhou et al., 2021). Setting these thresholds requires knowing the legacy system's baseline well enough to notice a meaningful drop rather than normal day-to-day variance, which is itself an argument for the observability-parity work done in Section 3.1 rather than something that can be improvised once the migration starts. Table 3 tracks these KPIs across the migration phases. Section 4.4 exercises these thresholds in simulation.

4.2 Rollback Design

Rollback needs to be designed, not assumed. An automated reversion that requires manual approval at two in the morning is not meaningfully different from a manual rollback, and the whole point of automated gating is to close that gap. The rollback path should be tested during shadow mode, not discovered for the first time during an actual incident, and it should revert cleanly to the exact previous traffic split rather than an approximate one (Hlybovets & Paprotskyi, 2024). Rollback is something that if it is taken care of as a first class deliverable with its acceptance criteria then it would happen sooner rather than later.

4.3 Post-Migration Validation

Reaching 100% traffic is not the end of the evaluation, even though it is often treated that way once the champagne comes out. KPIs should stay under the same monitoring rigor for a sustained window after full cutover, since some regressions, particularly ones tied to less common ad unit types or lower-volume DSP partners, only become statistically visible once they have had enough traffic volume to show a trend. It is also worth keeping the legacy exchange in a dormant, quickly reactivatable state for some defined period after cutover rather than decommissioning it immediately, since the cost of keeping a fallback alive for a few extra weeks is generally small next to the cost of having no fallback at all if a delayed regression appears.

Table 3. KPI monitoring targets across migration phases

Migration Phase	Bid Response Rate Target	Fill Rate Target	RPM Target	Observation Window
Shadow mode	Comparison only, not gated	Comparison only	Comparison only	Full traffic-pattern coverage
5% split	≥ legacy baseline	≥ legacy baseline	≥ legacy baseline	24–48 hours
10% split	≥ legacy baseline	≥ legacy baseline	≥ legacy baseline	24–48 hours
25% split	≥ legacy baseline	≥ legacy baseline	≥ legacy baseline	24–48 hours
50% split	≥ legacy baseline	≥ legacy baseline	≥ legacy baseline	24–48 hours
100% cutover	≥ legacy baseline	≥ legacy baseline	≥ legacy baseline	Sustained post-cutover

4.4 Simulation-Based Validation of the Migration Framework

To address the evidentiary gap noted during peer review, this section reports a discrete-event simulation study that exercises the framework's migration stages, KPI gate thresholds, and rollback mechanisms against synthetic exchange telemetry. All data in this section is synthetic; no production traffic was used. The study is described at a level of detail intended to make the experiments reproducible: every parameter, threshold, and scenario definition is disclosed below, the generator uses fixed random seeds (master seed 20260930), and the simulation code is available from the author on request.

4.5 Simulation Setup

The simulator models a CTV platform's auction traffic at one-minute granularity over 12 simulated days (17,280 minutes). Bid-request volume per minute is drawn from a Poisson distribution with a diurnal mean peaking at 20,000 requests per minute. The legacy exchange baseline is calibrated to a bid response rate of 0.92, a fill rate of 0.61, an RPM of $8.50, and a p99 latency of 95 ms, each with mild per-minute Gaussian noise; the healthy new bidder is calibrated to near-parity or slightly better (response 0.925, fill 0.615, RPM $8.55, p99 88 ms).

The migration timeline follows Section 3: two days of shadow-mode mirroring, then served splits of 5%, 10%, 25%, 50%, and 100%, each held for a 48-hour observation window. Health-check gates run every 15 minutes during served splits, comparing trailing 4-hour KPIs of the new-bidder-served share against the legacy-served share over the same window (the legacy baseline of Section 4.1). A gate breaches when any KPI is worse than baseline by more than the tolerance: 5% relative on bid response rate, fill rate, and RPM, and 20% relative on p99 latency. On breach, traffic rolls back to the previous split; three rollback policies are compared: automated (5 minutes after detection), manual (6 hours after detection), and no rollback (gates ignored, migration continues).

Four scenario groups are run. Readiness exercises the Section 3.1 parity checklist on a synthetic testbed with

one injected functional gap (an unsupported ad-unit format). Shadow exercises the Section 3.2 mirroring validation with a latent per-ad-unit fill regression in the new bidder. S0 is a clean cutover with a healthy new bidder. S1 injects a latent bidder regression at the 25% split, a latency bug driving p99 to 170 ms, response rate to 0.78, fill to 0.50, and RPM to $7.60, and compares the three rollback policies. S3 varies the commercial-KPI gate tolerance (2%, 5%, 10%) against a sustained moderate regression (roughly 8% below baseline on commercial KPIs) and against a 90-minute transient dip of the same depth that self-corrects.

4.6 Results

The readiness check failed one of seven criteria, the new bidder did not support all legacy ad-unit types in the testbed, and the framework blocked the migration before any traffic moved, with zero revenue impact. The shadow check caught a 29.0% fill gap on mid-roll units (shadow fill 0.44 against legacy 0.62) while the other three ad-unit types matched within 0.8%, holding the migration at zero served traffic, again with zero revenue impact. Both early stages did what the framework claims they do: they convert would-be production incidents into pre-migration findings.

Table 4. Gate-tolerance sensitivity: detection latency on a sustained ~8% regression (S3a) and behavior on a self-correcting 90-minute dip (S3b); synthetic data, master seed 20260930.

Tolerance	S3a: sustained ~8% regression	S3b: 90-minute transient dip
2%	detected after 75 min	rolled back (unnecessary)
5%	detected after 165 min	rode it out, no action
10%	missed entirely	rode it out, no action

The clean cutover (S0) ran 960 gate evaluations with zero breaches, reached 100% traffic, and finished with revenue 0.01% below the all-legacy counterfactual: no measurable disruption. On the S1 regression, the gates detected the breach 75 minutes after injection, and the rollback policy determined the cost: automated rollback limited the loss to $13,792 (1.09% of period revenue), manual rollback to $14,875 (1.18%), and ignoring the gates to $47,012 (3.73%). Automated rollback cut the loss by roughly 3.4x relative to proceeding despite the breached gates.

The tolerance comparison in Table 4 shows the trade-off Section 4.1 anticipates. Tighter tolerances detect a sustained regression faster (75 vs. 165 minutes) but react to a transient dip that would have self-corrected, triggering an unnecessary rollback; the 10% tolerance misses the sustained 8% regression entirely. Threshold selection is therefore a genuine tuning decision against baseline volatility, not a value the framework can fix for all platforms.

4.7 Interpretation

The results support the framework's central claim with one qualification worth stating plainly. Each stage did its designed job in simulation: the readiness and shadow checks stopped defective migrations before traffic moved, the phased gates detected a real regression in about an hour, and automated rollback limited revenue loss to roughly a third of the ignore-the-gates cost. The qualification is the threshold trade-off in Table 4: no single tolerance dominates, and the illustrative 5% value used here is a starting point to be calibrated against a platform's own baseline variance, exactly as Section 4.1 argues. All results are conditional on the stated synthetic parameters and should be read as evidence about the framework's mechanics, not as performance guarantees for any production migration.

5. BROADER IMPLICATIONS

5.1 Economic Effects

Migrating legacy exchange infrastructure has consequences that extend past the migration itself. Programmatic access through a modern, standards-compliant bidder tends to attract more demand-side participation, and that additional competition for inventory is part of what funds the content budgets that determine what a free streaming platform can actually offer (Choi & Mela, 2025). Platforms that keep running legacy exchange infrastructure are not failing outright, but they are likely leaving some amount of achievable yield on the table indefinitely, which is a slow, compounding cost rather than a single dramatic one.

5.2 Long-Term Outlook

This migration pattern is also a rehearsal for a bigger one. The industry's eventual move from OpenRTB 2.x to OpenRTB 3.0 will require its own bidder-side changes, and platforms that have already been through a legacy-to-modern migration once will have institutional muscle memory, tooling, and a tested rollback design that platforms attempting a major protocol transition for the first time will not.

5.3 Implications for Future Protocol Transitions

The specific readiness dimensions in Section 3.1 will likely need updating for an OpenRTB 3.0 transition rather than being reused unchanged, since the newer specification introduces structural changes to the request and response schema, not just incremental field additions. But the underlying sequencing, readiness assessment, shadow mode, phased gating, governed rollback, does not depend on OpenRTB version specifics and should hold up across that transition reasonably well. Platforms planning ahead might reasonably treat this migration as an opportunity to build reusable shadow-

mode and gating infrastructure now, rather than infrastructure scoped narrowly to a single legacy-exchange replacement, given that a similar migration shape is coming again within a few years regardless.

5.4 Limitations and Threats to Validity

This paper is a synthesis built from deployment-safety, legacy-migration, and RTB-economics research, not a report on one company's specific migration with its own measured outcomes. The industry statistics commonly cited in discussions of this topic, a specific CPM discount attributed to legacy infrastructure, a specific migration cost-overrun rate, a specific incident-rate comparison between mirrored and direct-cutover approaches, could not be independently verified against a citable source during this paper's preparation and are deliberately omitted rather than repeated on the strength of general industry consensus. The simulation study in Section 4.4 now exercises the migration stages, gate thresholds, and rollback design against synthetic telemetry, converting the framework's central claim from a design argument into a mechanically tested one, albeit on synthetic data. Readers should still treat the framework as a structured starting point, calibrated against their own platform's baseline metrics, rather than as a validated prediction of outcome for any specific platform.

6. CONCLUSION

Replacing a legacy ad exchange on a live CTV platform is a high-stakes engineering problem mostly because the cost of getting it wrong is immediate and measurable, not because any individual piece of the work is unprecedented. This paper organized readiness assessment, traffic mirroring, phased rollout with automated gating, and rollback design into a single framework, alongside the organizational governance work that tends to get less attention than the technical design but matters just as much to whether a migration actually lands cleanly.

For a team staring at this migration for the first time, the practical value here is mostly in not having to reinvent the sequencing from scratch, and in having a vocabulary for the readiness gaps that will show up during pre-migration testing regardless of how well the plan was written beforehand. Whether five traffic increments are the right number, or whether a twenty-four-hour observation window is long enough for any particular platform's traffic patterns, is a judgment call best made against that platform's own baseline data, not against this paper's defaults.

References:

Birrell, A. D., & Nelson, B. J. (1984). Implementing remote procedure calls. *ACM Transactions on Computer Systems, 2*(1), 39–59.

Bisbal, J., Lawless, D., Wu, B., & Grimson, J. (1999). Legacy information systems: Issues and directions. *IEEE Software, 16*(5), 103–111.

Blinowski, G., & Pełka, P. (2026). Comparative performance analysis of RPC frameworks in public cloud environments. *Concurrency and Computation: Practice and Experience, 38*(2).

Choi, H., & Mela, C. F. (2025). Optimizing reserve prices in display advertising auctions. *Marketing Science*.

de Jong, M., van Deursen, A., & Cleve, A. (2017). Zero-downtime SQL database schema evolution for continuous deployment. In *Proceedings of the IEEE/ACM 39th International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)* (pp. 143–152).

Di Francesco, P., Lago, P., & Malavolta, I. (2019). Architecting with microservices: A systematic mapping study. *Journal of Systems and Software, 150*, 77–97.

Frade, J. L. H., de Oliveira, J. H. C., & Giraldi, J. de M. E. (2021). Advertising in streaming video: An integrative literature review and research agenda. *Telecommunications Policy, 45*, 102186.

Hlybovets, A., & Paprotskyi, I. (2024). Increasing the fault tolerance in microservice architecture. *Cybernetics and Systems Analysis, 60*, 480–488.

Karabey Aksakalli, I., Çelik, T., Can, A. B., & Tekinerdoğan, B. (2021). Deployment and communication patterns in microservice architectures: A systematic literature review. *Journal of Systems and Software, 180*, 111014.

Karlsson, N. (2020). Feedback control in programmatic advertising: The frontier of optimization in real-time bidding. *IEEE Control Systems Magazine, 40*(5), 40–77.

Ou, W., Chen, B., Dai, X., Zhang, W., Liu, W., Tang, R., & Yu, Y. (2024). A survey on bid optimization in real-time bidding display advertising. *ACM Transactions on Knowledge Discovery from Data, 18*(3).

Ramaswamy, Y. (2024). Zero downtime deployments in DevOps: Blue-green, canary, and feature flag techniques. *International Journal of Communication Networks and Information Security, 16*(5), 969–977.

Sayedi, A. (2018). Real-time bidding in online display advertising. *Marketing Science, 37*(4), 553–568.

Sneed, H. M., & Verhoef, C. (2020). Cost-driven software migration: An experience report. *Journal of Software: Evolution and Process, 32*(7), e2236.

Tyree, J., & Akerman, A. (2005). Architecture decisions: Demystifying architecture. *IEEE Software, 22*(2), 19–27.

Zhou, X., Peng, X., Xie, T., Sun, J., Ji, C., Li, W., & Ding, D. (2021). Fault analysis and debugging of microservice systems: Industrial survey, benchmark system, and empirical study. *IEEE Transactions on Software Engineering, 47*(2), 243–260.

Sujan Kumar Reddy Challa
Independent Researcher,
USA.
sujan.challa10@gmail.com
ORCID: 0009-0005-8705-9770