Journal of Information Systems Engineering and Management
2026, 11(2s) e-ISSN: 2468-4376 https://www.jisem-journal.com/ **Research Article**

Expanding Ad Inventory Through New Surface Monetization: A Technical Analysis of Discovery Surface and Creator Content Surface Implementation

Rishi Kanth Alapati
University of Southern California, Los Angeles, USA

ARTICLE INFO

Received: 03 Feb 2026

Revised: 08 Feb 2026

ABSTRACT

This article examines the technical implementation of advertisement integration into previously unmonetized surfaces within a major social media platform ecosystem. It details the engineering approaches, system modifications, and performance optimization techniques employed to monetize these surfaces while maintaining positive user engagement metrics successfully. The article investigates how the engineering team extended the advertisement rendering system to accommodate diverse creative formats tailored to unique display requirements, implemented backend eligibility framework extensions for surfacespecific rule implementation, and established comprehensive real-time monitoring capabilities. Additionally, the article explores the sophisticated revenue attribution mechanisms developed for content creator monetization, including creator revenue sharing and eligibility determination algorithms. Through cross-functional technical integration across finance systems, organic surface protection mechanisms, and infrastructure scaling solutions, the implementation achieved successful monetization expansion without compromising user experience quality, demonstrating how careful technical planning, crossfunctional collaboration, and data-driven decision-making can balance revenue objectives with platform integrity.

Keywords: Monetization Architecture, Cross-Platform Rendering, Event-Driven Eligibility, Real-Time Analytics Pipeline, Creator Revenue Attribution

1. Introduction

The dilemma that social media sites face is always in balancing monetization objectives and maintenance of user experience. Placing advertisements on previously ad-free surfaces is a titanic technical and product problem, requiring a high level of engineering solutions to maintain the integrity of the platform and increase revenue potential. This article discusses a successful case study of a monetization initiative on a major social platform's previously unmonetized discovery and creator content surfaces.

Modern studies in digital experience optimization identify that successful monetization strategies for platforms need to balance the conflict between metrics of revenue generation and user satisfaction carefully. Successful implementations generally adopt a gradual deployment approach where new ad surfaces are released with conservative initial frequency caps that incrementally move up as user adaptation is realized [1]. The platform's monetization strategy for the new surfaces is a case in point, starting with impression caps some 60% below normal feed placements in early rollout.

Technical sophistication in enabling monetization on previously advertisement-free surfaces demands advanced backend architecture updates. Fast ad delivery necessitates solid event processing capacity that can tolerate high traffic levels while still meeting stringent latency needs [2]. The implementation solution involved using a distributed event processing architecture with the scale to cope with the sheer impression volume driven by the new surface interactions.

The platform's engineers understood that various creative formats would behave differently on the newly monetized surfaces. The team had performed a large-scale multivariate test across cohorts of users, examining how different creative dimensions influenced key metrics. These tests indicated that some formats registered significantly higher engagement levels in certain contexts - square formats working very well in grid-based layouts and vertical rectangles showing better performance in scrolling feed contexts.

For creator content monetization, the platform applied a complex revenue-sharing system that necessitated changes in core attribution systems. The rollout drew from learnings from platforms that had earlier launched creator monetization mechanisms, creating dynamic revenue ratio determination mechanisms based on numerous variables such as creator tier, engagement, and content category.

The planned depreciation of the underperforming 2×2 video format to the improved 1×2 format illustrates the data-driven optimization methodology that defined the rollout. Following thorough A/B testing that demonstrated higher scroll-through rates on the vertical format, the engineering team orchestrated a planned rollout away from the larger format, orchestrated to mitigate the impact on active campaigns.

2. System Architecture Changes

2.1 Creative Format Support

The engineering group expanded the ad rendering system to meet new creative format specs specific to the individual display needs of the target surfaces. Three formats could be used in the deployment: 1x1 (square), 1x2 (vertical rectangle), and 2x2 (big square). Cross-platform rendering technologies played a very significant role in supporting the presentation of advertisements in various device ecosystems. The team utilized responsive design concepts that dynamically modified creative components according to screen limitations while upholding brand consistency and messaging readability [3].

Technical implementation involved a component-based design that kept presentation logic distinct from business rules, supporting effective format adjustment without the replication of core functionality. This was in line with contemporary cross-platform development principles that focus on code reusability and maintenance effectiveness. The rendering pipeline utilized device-specific optimizations for both iOS and Android platforms, taking into consideration platform-specific rendering performance and behaviors. Implementation issues were overcoming fragmentation across generations of devices while maintaining backward compatibility with previous platform releases that were still actively being used [3].

2.2 Backend Eligibility Framework Extensions

The advertisement eligibility system also went through radical architecture transformations to enable surface-specific rule execution. Placement-specific eligibility evaluation modules were designed by the engineering team that worked within an event-driven architecture to allow real-time processing of targeting attributes and contextual signals. This supported loosely coupled system components that evolved independently while upholding system cohesion through standardized event contracts [4].

Surface-directed frequency capping policies were enforced with an event mesh topology that allowed the sharing of information among distributed system elements. This provided for uniform policy enforcement regardless of the extent of user interactions over multiple services and application domains. Event-driven implementation offered inherent scalability benefits by enabling the system to manage variable traffic patterns in an efficient manner while upholding strict latency guarantees. Optimizations in terms of performance involved the enforcement of event filtering at the source to avoid unnecessary network overhead and processing [4].

The use of new metadata fields for surface compatibility flagging utilized event-carried state transfer patterns to preserve contextual data during the advertisement selection process. The implementation used a publish-subscribe communication pattern in which targeting decisions involved real-time signals from diverse system components. The event-based process allowed for swift response to shifting

Journal of Information Systems Engineering and Management
2026, 11(2s) e-ISSN: 2468-4376 https://www.jisem-journal.com/ **Research Article**

circumstances, as the system was able to dynamically adapt eligibility rules in response to performance metrics and system health indicators [4].

Feature	Format Support	Eligibility Framework
Key Technologies	Cross-platform rendering, Responsive design	Event-driven architecture, Event mesh topology
Implementation Approach	Component-based design, Presentation/business logic separation	Placement-specific modules, Publish-subscribe model
Optimization Techniques	Device-specific rendering, Platform adaptations	Source-level event filtering, Realtime signal processing
Compatibility Solutions	Device fragmentation management, Backward compatibility	Standardized event contracts, Contextual data preservation
Scalability Methods	Code reusability patterns, Maintenance efficiency	Variable traffic handling, Strict latency maintenance

Table 1: Cross-Platform Technologies: Development Approaches, Frameworks, and Implementation Strategies [3, 4]

3. Data-Driven Performance Optimization

3.1 Real-Time Metrics Pipeline

A central aspect of the deployment was the creation of exhaustive performance tracking functionality using a sophisticated real-time metrics pipeline. The unified system tracked user engagement on organic content as well as ads, processing event volumes scaling with usage patterns on the platforms. The pipeline design used the Lambda Architecture pattern, blending batch processing for historical insights with stream processing for real-time understanding. This hybrid solution supported both indepth retrospective analysis and real-time anomaly detection capabilities, important for ensuring experience quality throughout monetization deployment [5].

The event collection layer utilized a distributed ingestion framework with buffer elements that avoided data loss during fluctuations in downstream processing. The events that were gathered were gradually augmented by contextual data as they left the pipeline, and multidimensional analysis was made possible by the appended contextual data. The real-time analytics module employed window-based aggregations over a variety of time horizons, which facilitated the detection of abrupt changes and ramp-like experience degradation patterns. The multi-tiered method of monitoring established a broad analysis platform that facilitated both tactical interventions and strategic optimization decisions [6].

Cohort comparison capability was built using an experiment-specific testing framework that automatically split users into groups based on exposure to various creative formats and surface arrangements. The system continually tests key metrics such as user retention rates, session depth measurements, scroll-through percentages, engagement metrics by creative type, and revenue performance by placement. Statistical significance testing was automatically performed on metric changes to avoid incorrect conclusions due to random variations or sampling errors [5].

3.2 Format Optimization Decision Process

The analysis of the data exposed vast performance differences between creative formats across surfaces and user segments. The engineering team put in place a standardized experimentation program that adhered to known digital experimentation practices, such as correct hypothesis development, sample size selection, and controlled exposure mechanisms. This methodical implementation helped ensure that the noted differences in performance could be ascribed to format differences and not confounding factors [6].

The experimentation design followed a sequential testing approach where early discovery experiments involving small exposure discovered successful format configurations before scaling up to validation experiments with bigger user segments. This helped balance innovation pace with risk management, supporting fast learning while safeguarding overall user experience. The analysis indicated predictable performance trends where vertical formats showed better engagement features than larger square formats in a variety of dimensions and user segments [5].

This empirical data drove the strategic depreciation of 2x2 video ads in favor of the higher-performing 1x2 alternative. The rollout followed a thoughtfully designed implementation roadmap that accommodated stakeholder notice, phased traffic migration, and ongoing performance monitoring through the transition window. This deliberate format optimization is an example of how data-driven decision-making can make user experience enhancement and business goals converge to enable sustainable monetization schemes that maintain platform health [6].

Performance Metric	Pipeline Component	Functionality	Application
Real-Time Analysis	Lambda Architecture	Hybrid batch/stream processing	Anomaly detection, experience quality monitoring
Data Integrity	Distributed ingestion with buffers	Prevent data loss during processing fluctuations	Reliable metrics collection under variable loads
Pattern Detection	Window-based aggregations	Multi-timeframe analysis	Identify both sudden and gradual experience changes
User Segmentation	Experimentation framework	Automatic cohort creation	Compare performance across format exposures
Format Evaluation	Sequential testing approach	Progressive experiment scaling	Balance innovation speed with risk management
Decision Support	Statistical significance testing	Validate metric variations	Prevent conclusions from random fluctuations
Format Performance	Comparative engagement analysis	Vertical vs. square format assessment	Identify optimal creative specifications

Table 2: Metrics Pipeline Components for Format Optimization [5, 6]

4. Implementation of Creator Content Surface Revenue Sharing

The creator content monetization module needed advanced revenue attribution systems to enable fair creator payoffs while preserving platform economics. The product deployment called for significant changes to core delivery and attribution infrastructure, impacting various backend elements along the monetization stack. Content monetization frameworks need to balance several concerns, such as user experience, creator incentive, and platform sustainability, to generate sustainable long-term value exchanges among all participants [7].

Journal of Information Systems Engineering and Management
2026, 11(2s) e-ISSN: 2468-4376 https://www.jisem-journal.com/ **Research Article**

Backend delivery logic was drastically rewritten to facilitate creator revenue-sharing capability. The system had a two-path auction process whereby ads could be delivered either via the old platformowned inventory path or the new creator-attributed path based on placement context and qualification rules. This implementation was consistent with industry best practices that focus on open revenue-sharing models as a primary driver for creator retention and content quality enhancement. Technical design accommodated variable revenue share percentages that might be dynamically varied depending on creator tier, content category, and performance factors without the need for system reconfiguration [8].

Eligibility calculation rules for revenue-sharing membership used several evaluation aspects, such as creator account standing, content quality metrics, policy compliance record, and audience attributes. The platform used a layered method with preliminary basic eligibility calculated through a light rule assessment, and complete participation needing more advanced analysis carried out through an async evaluation pipeline. This approach reflects successful monetization tactics that focus on quality standards and brand safety as preconditions for viable content monetization schemes [7].

Revenue splitting calculations were achieved with a dynamic framework that accommodated various attribution models according to engagement behavior and performance metrics. The underlying attribution system combined signals such as view duration, interaction rates, and conversion performance to compute fair revenue allocation percentages. This implemented multi-touch attribution methods that could properly allocate value along sophisticated user paths involving numerous interaction points. The attribution models combined first-touch and last-touch elements with customizable weighting parameters for balanced credit assignment that accounted for discovery and conversion drivers [8].

Technical architecture ensured strict segregation between native platform and creator content revenue streams using isolated tracking and attribution processes. This segregation allowed exact performance measurement and optimization without data contamination between inventory sources. The rollout enabled extensive analytics dashboards for creators, giving visibility into revenue generation trends and performance metrics. Studies on multi-channel attribution illustrate that visible performance reporting considerably boosts platform trust and creator investment, leading to higher quality content creation benefiting the entire platform ecosystem [8].

Component	Function	Technical Approach	Business Impact
Delivery Logic	Ad path determination	Dual-path auction mechanism	Maintains platform economics while enabling creator monetization
Revenue Allocation	Payment distribution	Variable percentage framework	Supports creator tiers and performance-based incentives
Eligibility System	Participation qualification	Tiered evaluation approach	Ensures quality standards and brand safety
Attribution Engine	Value assignment	Multi-touch attribution model	Accurately credits both discovery and conversion influences
Performance Analytics	Creator insights	Segregated tracking mechanisms	Increases platform trust and content quality investment

Financial Isolation	Revenue stream separation	Dedicated tracking architecture	Prevents data contamination between inventory sources
Creator Dashboard	Performance visibility	Comprehensive analytics interface	Enhances creator engagement and retention

Table 3: echnical Components of Creator Content Surface Revenue Sharing System [7, 8]

5. Cross-Functional Technical Integration

The implementation needed harmonized technical integrations in more than one organizational domain, implying complex system interfaces and communication protocols between erstwhile unconnected components. Effective monetization expansion initiatives have to overcome typical integration issues such as the incompatibility of data formats, the variation of API standards, and the variability of transaction processing models [9]. Integration working groups were set up by the implementation team with representation from all impacted domains, working under common technical specifications and implementation schedules.

5.1 Finance Systems Integration

Financial systems had to undergo substantial architectural changes to accommodate the increased monetization surfaces and revenue-sharing features. New placement-specific billing codes were made available through an extensible taxonomy system that facilitated hierarchical classification while preserving backward compatibility with legacy reporting structures. This solution addressed one of the most pervasive enterprise integration problems: ensuring system compatibility while introducing new capability without upsetting existing operations [9].

Creator revenue share revenue attribution models required transaction processing pipelines to be extended to accommodate multi-party payment flows with dynamic split ratios. The adoption utilized standardized data exchange formats between systems to provide uniform transaction processing across organizational boundaries. Integration professionals acknowledged that successful crosssystem financial integration relies on well-documented communication protocols ensuring data integrity throughout complex processing flows [9].

Reconciliation procedures for divided revenue accounting were enforced through automated validation processes that matched anticipated allocations against real distribution records. Such systems had advanced exception handling capabilities that could detect and highlight discrepancies for investigation purposes and permit regular operations to proceed uninterrupted. The rollout was carried out according to best practices for enterprise integration through well-defined data ownership boundaries and reconciliation duties within organizational domains [10].

5.2 Organic Surface Protection Mechanisms

Technical controls were deployed to maintain organic content integrity throughout the newly monetized surfaces. Proper ad-to-organic content ratios were sustained by density control algorithms using dynamic adjustment mechanisms that reacted to scheduled parameters and real-time signals. The architecture adopted robust application design principles such as circuit breaking and graceful degradation to ensure experience quality during system variability [10].

Quality threshold mechanisms avoided low-quality ad intrusion via multi-dimensional appraisal frameworks that monitored both creative quality and contextual appropriateness. The deployment included redundancy patterns that provided uniform quality appraisal even under partial system failure. It was in line with resilient system design procedures that focus on preserving core functionality amid unforeseen operating conditions [10].

User experience protection circuit breakers adaptively modify ad density in response to engagement indicators, applying a proportional tiered response mechanism based on measured experience degradation. The protection mechanisms illustrated the health checking and circuit-breaking patterns required to ensure system reliability in distributed complex systems. Automatic intervention was facilitated by the implementation before problems might multiply into considerable user experience issues [9].

5.3 Infrastructure Scaling Solutions

The infrastructure team applied end-to-end scaling solutions to accommodate the boosted processing loads of the increased monetization surfaces. Capacity planning for higher rendering load applied horizontal scaling methods that spread out processing across multiple instances, allowing linear capacity growth without architectural modifications. This deployment obeyed cloud-native design principles that focus on elasticity and service distribution to manage fluctuating workloads effectively [10].

Fallback mechanisms for technical failures executed an advanced hierarchy of degradation that preserved core functionality with partial system outage. The system used redundancy and replication patterns that removed single points of failure while facilitating automated recovery from typical failure situations. These methods exhibited industry best practices for constructing fault-tolerant distributed systems that remain available even in the face of component failure [10].

Graceful degradation designs for high-load environments integrated load-shedding methods that favored major operations during capacity stress. The system held pre-defined service level targets for every component, with non-critical processing automatically postponed during overload periods. Caching optimizations for new creative types applied multi-level caching techniques that really lowered rendering overhead using effective resource reuse, with context-specific invalidation policies ensuring content freshness while optimizing cache efficiency [10].

Integration Domain	Key Components	Implementation Approach	Technical Benefit
Finance Systems	Placement-specific billing codes	Extensible taxonomy system	Backward compatibility with legacy reporting
	Revenue attribution models	Standardized data exchange formats	Consistent crossboundary transaction processing
	Reconciliation procedures	Automated validation workflows	Exception handling with operational continuity
Organic Protection	Density control algorithms	Dynamic adjustment mechanisms	Experience quality during system fluctuations
	Quality threshold mechanisms	Multi-dimensional evaluation frameworks	Consistent quality evaluation during degradation

Journal of Information Systems Engineering and Management
2026, 11(2s) e-ISSN: 2468-4376 https://www.jisem-journal.com/ **Research Article**

	Experience protection circuit breakers	Tiered response system	Preemptive intervention before cascade failures
Infrastructure Scaling	Capacity planning	Horizontal scaling distribution	Linear growth without architectural changes
	Fallback strategies	Redundancy and replication patterns	Core functionality during partial outages
	Graceful degradation patterns	Load-shedding techniques	Service prioritization during capacity constraints

Table 4: Cross-Functional Integration Components for Monetization Expansion [9, 10]

6. Results and Technical Implications

Technical execution led to effective monetization expansion without adverse effect on core user experience metrics. This result constitutes a notable engineering success in that introducing advertisements generally entails performance tradeoffs. Post-execution analysis proved that important engagement metrics stayed within defined variance ranges while monetization capabilities meaningfully grew. This effective balance showcases effective digital monetization execution, respecting the fundamental principles of sustainable digital transformation strategies [11].

Performance monitoring showed that session depth metrics were consistent with both experimental and control cohorts after full deployment. This consistency demonstrated that adding advertisements to previously unsanctioned surfaces did not interfere with core engagement behavior when added with proper controls and optimization practices in place. The deployment is an example of how monetization strategies can be designed to preserve user experience quality while increasing revenue channels when informed by user-friendliness design principles [11].

Revenue performance outpaced early estimates by about 15% due to greater-than-anticipated adoption by advertisers of the new placements and creative formats. The result confirmed the technical strategy of building flexible creative specifications targeted to the distinct characteristics of each surface. The success is in line with findings that individualized and contextually targeted monetization strategies tend to outperform standard implementations in both the revenue and engagement metrics [11].

Important technical success factors were a modular architecture design that allowed format-specific optimizations. The implementation adopted distributed systems design paradigms that focused on component separation and well-defined interface definitions. Such an architectural style allowed for scalable scaling and optimization of separate system components without asking for system-wide modifications. Modular design is a basic distributed systems pattern that simplifies system complexity at the cost of enabling domain-specific optimizations [12].

Real-time monitoring function allowed fast response when anomalies in behavior patterns occurred. The deployment included end-to-end observability capabilities with fine-grained telemetry, autonomic alerting, and visual dashboards that gave real-time visibility into the system's performance. This is in line with core monitoring patterns for distributed systems, where visibility across component boundaries becomes more important as the complexity of systems increases [12].

Data-informed decision-making informed format choice and optimization across the implementation life cycle. The team developed strict experimentation methods that removed subjective taste from format choice, using instead objective performance metrics to inform implementation decisions. This method is an example of how effective digital transformation programs use data as a strategic resource to inform technical decisions as opposed to using intuition or stakeholder tastes [11].

Cross-cutting system integration with well-defined interfaces made it easy to collaborate across organizational boundaries. Implementation created well-defined API contracts among system parts, making it possible to develop components independently while still assuring compatible integration. This methodology enforces the interface segregation principle that has been found crucial for dealing with complexity in distributed systems while allowing independent evolution of components [12].

Strong fallback mechanisms provided system resilience during the process and after. The design used multiple layers of protection, such as circuit breakers, graceful degradation patterns, and recovery mechanisms provided by automated processes that preserved core functionality in cases of unexpected situations. These implementations are fundamental resilience patterns of distributed systems, such as circuit breaking, bulkheading, and graceful degradation, that, combined, allow systems to preserve core functionality even when individual parts fail [12].

Conclusion

The presented case study illustrates that the process of careful technical architecture, systematic analysis of performance, and cross-functional engineering collaboration allowed the successful expansion of monetization and maintained the quality of user experience. The introduction of the modular architecture design, full operational real-time monitoring, and decision-making that is based on data has come in handy to strike a balance between revenue goals and the satisfaction of users. Through the creation of specialized and creative formats, setting up advanced eligibility systems, and building a strong protection system, the engineering team was able to introduce adverts to monetize hitherto unexploited surfaces without interfering with the fundamental patterns of engagement. The revenue-sharing implementation of Creator Content Surface is another clear example of how technical innovation can establish fair value distribution between creator and platform. Cross-functional interdependence among organizational areas guaranteed the uniform performance of the system, as well as the possibility of special optimization of each of the technical domains. These patterns of resilience that were integrated across the architecture ensured that the localized issues did not impact the overall user experience, which was one of the differences that made the initiative very successful. The implementation offers useful lessons that can be applied in the adoption of other similar digital advertising platforms and the significance of careful technical design implementation in ensuring sustainable monetization plans.

References

[1] Dieter Rappold, "Balancing User Experience and Monetization: The Ultimate Guide for Mobile Apps," ContextSDK Blog, 2024. [Online]. Available: https://contextsdk.com/blogposts/balancinguser-experience-and-monetization-the-ultimate-guide-for-mobile-apps

[2] Jin, "High-Performance Notification Systems Architecture, Design, Maintenance, and Operation," Medium, 2024. [Online]. Available: https://medium.com/tech-x-humanity/high-performancenotification-systems-architecture-design-maintenance-and-operation-e0f906d8119d

[3] Velvetech, "Top Cross-Platform App Development Frameworks,". [Online]. Available:
https://www.velvetech.com/blog/cross-platform-technologies/

[4] Solace, "The Ultimate Guide to Event-Driven Architecture Patterns,". [Online]. Available: https://solace.com/event-driven-architecture-patterns/

[5] Amber Chowdhary, "Understanding Real-time Data Pipelines: Architecture, Implementation, and Technologies," ResearchGate, 2025. [Online]. Available: https://www.researchgate.net/publication/388803146_Understanding_Realtime_Data_Pipelines_Architecture_Implementation_and_Technologies

[6] Personizely, "A Digital Experimentation Guide for Businesses,". [Online]. Available: https://www.personizely.net/blog/digital-experimentation

[7] Colin Burton, "Content Monetization Explained: Models, Platforms, & Strategies," Thinkific Blog, 2024. [Online]. Available: https://www.thinkific.com/blog/how-to-monetize-content/

[8] Yusuf Abiodun et al., "Advances in Multi-Channel Attribution Modeling for Enhancing Marketing ROI in Emerging Economies," ResearchGate, 2025. [Online]. Available: https://www.researchgate.net/publication/392708829_Advances_in_Multi-Channel_Attribution_Modeling_for_Enhancing_Marketing_ROI_in_Emerging_Economies

[9] ThinkAI, "Overcoming Common Challenges in Enterprise Integration: Tips and Solutions," 2024. [Online]. Available: https://thinkaicorp.com/overcoming-common-challenges-in-enterpriseintegration-tips-and-solutions/

[10] Google Cloud, "Patterns for scalable and resilient apps," 2025. [Online]. Available: https://cloud.google.com/architecture/scalable-and-resilient-apps

[11] Vijay Kumar Tiwari Brij, "The Monetization Playbook: Digital Transformation Success in the Digital Economy," ResearchGate, 2025. [Online]. Available: https://www.researchgate.net/publication/391639452_The_Monetization_Playbook_Digital_Transformation_Success_in_the_Digital_Economy

[12] Mahi Mullapudi, "Mastering Distributed Systems: Essential Design Patterns for Scalability and Resilience," Dev.to, 2024. [Online]. Available: https://dev.to/tutorialq/mastering-distributedsystems-essential-design-patterns-for-scalability-and-resilience-35ck