International Journal of Computational and Experimental Science and ENgineering (IJCESEN)

Vol. 11-No.4 (2025) pp. 9988-9994
http://www.ijcesen.com

ISSN: 2149-9144

Research Article

Reference Architecture for Artificial Intelligence-Driven Real-Time Orchestration in Cloud-Native Advertising Systems

Sujoy Datta Choudhury*

Independent Researcher, USA
* **Corresponding Author Email:** sujo2y@gmail.com - **ORCID:** 0000-0002-0047-0050

Article Info:

DOI: 10.22399/ijcesen.4588
Received : 15 December 2025
Revised : 21 December 2025
Accepted : 24 December 2025

Keywords

Cloud-Native Architecture,
Real-Time Orchestration,
Microservices,
Artificial Intelligence,
Advertising Systems

Abstract:

Contemporary cloud-native advertising platforms confront a challenging dichotomy between ultra-low latency demands and computational intensity associated with sophisticated artificial intelligence mechanisms. The architectural framework delineated herein positions orchestration as a principal design element rather than auxiliary implementation minutiae. System components are organized across discrete strata encompassing ingress management, stateless coordination, persistent services, and delivery mechanisms, interconnected via contract-driven design methodology. Coordination strata manage feature provisioning systems, model inference endpoints, and marketplace computation engines while preserving stringent latency thresholds and facilitating autonomous component maturation. Resilience methodologies incorporating operation repeatability, failure isolation circuits, and stateless construction establish continuous availability alongside graceful performance reduction. Visibility integration emerges as a fundamental design characteristic through distributed request tracking, structured event logging, and defined service performance criteria. Deployment governance tools support controlled releases via capability toggles, versioned configurations, and incremental exposure strategies. The framework accommodates burgeoning demands concerning power consumption optimization, decisional transparency, and sophisticated machine learning integration spanning reinforcement algorithms and expansive language models. Though centered on advertising infrastructure, these design approaches extend applicability toward latency-constrained, intelligence-augmented cloud-native solutions demanding multi-dependency synchronization under temporal restrictions.

1. Introduction

1.1 Coordination Complexity Within Modern Advertising Platforms

Contemporary advertising infrastructure operating on cloud-native principles experiences pronounced conflict between operational velocity requirements and computational sophistication [1]. Modern implementations must execute decisional processes under severely compressed temporal windows while managing intricate auction methodologies, reinforcement-based bidding algorithms, and sustained experimental protocols [6]. Traditional resolution approaches employed monolithic architectures wherein coordination mechanisms became dispersed as ancillary code fragments distributed throughout system boundaries. Such methodologies generate brittleness during scalability progression and evolutionary cycles, complicating autonomous component modification and holistic behavioral comprehension.

1.2 Elevating Coordination to Architectural Priority

The architectural framework advances coordination from implementation detail toward foundational design consideration. Rather than positioning model synchronization, service interaction, and policy management as subordinate to commercial logic, designated coordination strata become established as cornerstones for resilient, scalable infrastructure. Design patterns derive from proven microservice methodologies while confronting distinct real-time advertising constraints, including traffic surge patterns, degradation requirements, and perpetual model refinement [1][2]. Coverage

spans complete request trajectories initiating from ingress points, progressing through attribute extraction, model computation, and marketplace logic, culminating in response transmission.

1.3 Contemporary Architectural Demands

Modern advertising infrastructure encounters escalating demands, reshaping design prerequisites. Power consumption considerations intensify examination of datacenter operations [3], mandating that coordination mechanisms incorporate environmental considerations within resource distribution determinations. Algorithmic transparency mandates necessitate exhaustive audit documentation elucidating decision-making sequences. Sophisticated machine learning methodologies incorporating reinforcement algorithms generate novel integration complexities [7][8]. Contributions herein furnish architectural patterns and design tenets absorbing evolutionary demands while sustaining fundamental characteristics, including minimal latency, sustained availability, and operational comprehensibility.

2. Requirements and Constraints for Real-Time Advertising Orchestration

2.1 Request Structure and Decisional Intricacy

Individual advertisement requests transport multifaceted contextual intelligence spanning user characteristics, positioning attributes, publisher limitations, regulatory stipulations, and marketplace conditions [5]. Coordination mechanisms must amalgamate these information streams while honoring contractual commitments toward demand and supply ecosystem participants. Decisional outputs transcend creative selection, encompassing pacing modifications, budget application, qualification screening, and multi-position distribution when circumstances warrant. Such intricacy demands architectural constructs capable of synthesizing heterogeneous inputs into unified decisions without introducing synchronization impediments.

2.2 Temporal Constraints and Traffic Fluctuation

Advertising infrastructure operates under rigorous temporal allocations permitting minimal computational or transmission overhead. Concurrently, these mechanisms must accommodate traffic variations spanning multiple magnitude orders within abbreviated intervals, propelled by occurrences, cyclical patterns, or content virality [1][6]. Such volatility necessitates an adaptable infrastructure supporting lateral expansion without decisional quality deterioration. Coordination strata must impose temporal allocations across dependencies, executing timeout and isolation mechanisms, preventing delayed components from propagating failures systemwide.

2.3 Resilience and Autonomous Progression

Sustained availability constitutes a non-negotiable prerequisite given the revenue ramifications of operational interruptions. Architecture must facilitate graceful performance reduction when dependencies encounter partial disruptions, ensuring advertisement decisional continuity even when specific features or models experience temporary unavailability. Such resilience must harmonize with autonomous component progression wherein model groups update training sequences, feature specialists modify extraction algorithms, and marketplace groups adjust bidding regulations without synchronized deployments [1]. Contract-based component interfaces enable such autonomy while sustaining system-wide coherence.

2.4 Visibility and Responsibility

Individual decisions require traceability across system boundaries with adequate context elucidating outcomes [2]. Such traceability supports operational troubleshooting, policy verification, regulatory adherence, and retrospective decisional quality examination. Visibility infrastructure must document not simply decisional occurrence but comprehensive provenance of inputs, intermediate calculations, and constraints influencing outcomes. Service performance criteria surrounding temporal allocation, error frequency, and decisional quality require explicit measurability, with coordination mechanisms responsible for threshold enforcement and violation exposure.

2.5 Power Efficiency Considerations

Extensive advertising computational loads execute within datacenters whose cumulative power consumption attracts heightened examination [3]. Architectural selections concerning service positioning, request direction, and resource distribution directly affect power efficiency. Coordination strata, functioning as computational work coordinators across platforms, assume responsibility for power-conscious determinations regarding inference execution locations and feature computation methodologies.

3. Layered Reference Architecture

3.1 Ingress and Protocol Standardization

Ingress strata manage protocol conclusion, authentication procedures, and preliminary request verification [1]. These components transform heterogeneous external protocols and request structures into standardized internal representations, protecting downstream components from client diversity. Through stable internal contract establishment at ingress perimeters, architecture permits external interface evolution without cascading modifications across system boundaries. Ingress strata execute fundamental request verification and augmentation, appending correlation identifiers enabling complete tracing through subsequent processing phases.

3.2 Stateless Coordination Foundation

Coordination strata function as intentionally stateless coordinators concentrating exclusively on compositional logic [7]. These components distribute requests toward feature provisioning systems, configuration endpoints, model inference mechanisms, and marketplace computation engines, subsequently aggregating outputs into unified decisions while imposing consistency constraints and temporal allocations. Statelessness facilitates lateral scalability and simplifies failure remediation since individual coordinator instances can handle arbitrary requests without session binding or state replication. Coordinators maintain exclusively correlation identifiers and minimal ephemeral context, delegating persistent state to explicit repository constructs.

3.3 Persistent Service Stratum

Persistent services furnish durability for configuration data, pacing conditions, budget monitoring, and frequency limitation information. These components frequently utilize hybrid deployment configurations merging traditional microservice frameworks with serverless or function-oriented approaches, managing traffic surges effectively [4]. Separation between stateless coordination and persistent services establishes distinct boundaries for consistency reasoning, durability characteristics, and expansion properties. State alterations adhere to explicit repository constructs encapsulating persistence considerations and furnishing defined concurrency characteristics.

3.4 Feature Extraction and Model Inference

Feature extraction alongside model inference mechanisms exposes functionalities through versioned service agreements. Feature provisioning systems calculate user characteristics, contextual indicators, and derived measurements necessary for downstream decisional processes. Model inference endpoints furnish predictions for trained machine learning constructs spanning elementary regression through sophisticated reinforcement algorithms [8]. These services possess internal implementation autonomy, maintaining the liberty to modify algorithms, training datasets, or infrastructure while preserving agreement compatibility with coordination strata.

3.5 Delivery and Event Broadcasting

Delivery strata convert coordinated decisions into structures anticipated by external bidders, advertisement servers, and integration associates. These components govern structured log and event broadcasting, feeding attribution mechanisms, reporting conduits, and offline learning workflows. Through delivery concern isolation from coordination logic, the architecture accommodates varied integration configurations without core decisional path complications. Event broadcasting furnishes a visibility infrastructure foundation, documenting decisional context and outcomes for downstream examination.

3.6 Contract-Driven Design Methodology

Interlayer interfaces receive designation as stable, versioned constructs subject to explicit governance [1]. Coordinators depend exclusively on these agreements, never accessing dependency implementation particulars. Such discipline permits autonomous evolution wherein model groups reconstruct inference infrastructure, feature specialists refactor extraction conduits, and marketplace groups revise auction algorithms without coordination code alterations. Resulting loose interconnection establishes foundation for autonomous deployability and group independence within extensive engineering organizations

4. Robustness, Observability, and Change Management Patterns

4.1 Operation Repeatability and Retry Protection

Operations must yield identical outcomes under repeated execution because transient disruptions and automatic retries constitute inevitable distributed system characteristics [1]. Coordinators

achieve repeatability through meticulous state mutation design, correlation identifier management, and duplication detection logic. Request identifiers traverse all processing phases, enabling services to identify and manage duplicate operations securely. Such property establishes resilience foundation permitting aggressive retry strategies without duplicate billing risks, impression enumeration, or budget depletion.

4.2 Timeout and Isolation Circuit Mechanisms

Timeouts alongside isolation circuits forestall cascading disruptions when dependencies decelerate or malfunction, executing compartmentalization patterns isolating malfunctions toward specific subsystems [1][2]. Coordinators establish suitable timeouts for individual dependencies based on service performance agreements and monitor threshold transgressions. When error frequencies exceed limits, isolation circuits activate to expedite failure rather than accumulating requests behind unresponsive services. These mechanisms sustain overall system availability even when individual components encounter performance reduction.

4.3 Distributed Request Tracking and Correlation

Coordination strata serve as inherent roots for distributed traces monitoring requests from ingress across all downstream dependencies toward final delivery [2]. Correlation identifiers propagate through service invocations, enabling complete request path reconstruction. Structured logs tagged with these identifiers, service-tier measurements, and complete traces combine generating comprehensive system behavioral representations. Such visibility infrastructure empowers engineers to answer queries concerning specific request outcomes and comprehend system-wide configurations.

4.4 Service Performance Criteria

Service performance criteria for temporal allocation, error frequency, and decisional quality receive explicit and quantifiable definitions [2]. Coordinators impose temporal allocations through suitable timeout establishment and transgression monitoring. Error allocations quantify permissible failure frequencies for individual service tiers, with monitoring mechanisms alerting when consumption surpasses limits. Decisional quality measurements document business-pertinent outcomes, empowering groups to identify performance regressions or policy enforcement deterioration.

4.5 Capability Toggles and Progressive Exposure

Capability toggles furnish granular control over functionality introduction, permitting new models or policies exposure toward regulated traffic segments [1]. Configuration modifications receive designation as versioned constructs with deployment lifecycles incorporating validation phases, progressive exposure, and automated reversion upon error identification. Because coordinators process complete traffic, positioning exists for consistent experimentation policy enforcement, ensuring proper isolation and statistical legitimacy in evaluations.

4.6 Canary Evaluation and Automated Reversion

Incremental exposure mechanisms enable secure deployment of modifications carrying commercial risk. Novel machine learning constructs might initially receive limited traffic percentages with performance continuously benchmarked against baseline policies [8]. Automated canary evaluation identifies performance deterioration and initiates reversion before substantial traffic exposure to problematic modifications. Such a systematic change governance approach diminishes the failure impact radius and strengthens organizational assurance in frequent production updates.

5. Future Directions and Scalability Considerations

5.1 Sophisticated AI Model Incorporation

Artificial intelligence trajectory within advertising indicates increasingly advanced model architectures [7][8]. Expansive language models initiate participation in creative comprehension and generation while reinforcement learning mechanisms optimize dynamic bidding approaches across extended temporal horizons. These sophisticated constructs introduce novel coordination complexities incorporating expanded computational requirements, streaming indicators from external mechanisms, and decisional policies evolving continuously rather than through discrete deployments.

5.2 Architectural Constructs for Technological Advancement

Presented patterns receive construction to absorb technological progression [1]. Distinct stratification ensures novel model categories integrate as supplementary dependencies without coordination logic restructuring. Robust agreements permit groups experimenting with innovative feature engineering or auction design approaches while sustaining system compatibility. Stateless coordination enables lateral expansion, accommodating heightened computational demands, while primary visibility furnishes transparency into increasingly opaque machine learning mechanisms.

5.3 Transparency and Regulatory Adherence

Transparency alongside regulatory adherence prerequisites continue to intensify surrounding algorithmic decisional processes and user privacy [2]. Coordination strata must generate auditable documentation of decisional formulation sequences beyond merely executing correct decisions. Tracing and logging infrastructure furnishes accountability foundations, yet prospective mechanisms will necessitate enriched semantic metadata documenting commercial logic and policy constraints affecting individual decisions.

5.4 Power-Conscious Coordination

Power efficiency will probably transition from operational consideration toward primary architectural constraint as datacenter power consumption attracts regulatory attention and organizations establish sustainability commitments [3]. Coordination mechanisms will necessitate power-conscious determination execution concerning workload positioning and resource distribution. This might materialize as routing policies favoring less power-intensive model alternatives when quality compromises prove acceptable, or scheduling approaches transferring non-latency-critical calculations toward intervals when renewable power abundance exists.

5.5 Organizational and Operational Ramifications

Ramifications extend beyond technical architecture toward group organization and operational methodologies [1]. Independence enabled through contract-based design permits organizations to structure groups around specific competencies, incorporating feature engineering, model development, and marketplace logic with distinct ownership perimeters. Visibility infrastructure supports such autonomy through dependency clarification and objective measurement provision for service performance agreements between groups. Change governance patterns diminish coordination overhead through autonomous deployment enablement while sustaining system-wide stability.

Table 1. *Core Requirements and Architectural Constraints for Real-Time Advertising Orchestration [1][5][6]*

Latency Budget	Millisecond-scale response windows	Minimal computational overhead tolerance	Timeout enforcement, circuit breakers
Observability	Complete decision traceability	End-to-end correlation requirements	Distributed tracing, structured logging
Energy Efficiency	Datacenter power consumption scrutiny	Resource-aware execution decisions	Power-conscious routing, workload scheduling

Table 2*. Layered Architecture Components and Responsibilities [1][4][7]*

Architectural Layer	Primary Responsibility	Key Components	Interface Type	Deployment Pattern
Ingress Stratum	Protocol normalization, authentication	Protocol terminators, validators, enrichers	External-to-internal contract	Edge-deployed services
Stateless Coordination	Request composition, dependency fanout	Orchestrators, timeout managers, aggregators	Service-to-service contracts	Horizontally scaled pods
Stateful Services	Persistence, state management	Configuration stores, budget trackers, pacing engines	Repository abstractions	Hybrid microservice-serverless
Feature and Model Serving	Attribute computation, inference	Feature extractors, ML model endpoints	Versioned service contracts	Dedicated compute clusters

Delivery Stratum	Response formatting, event emission	Format transformers, log publishers, attribution feeds	Internal-to-external contract	Edge-deployed services

Table 3. *Robustness Patterns and Implementation Mechanisms [1][2][8]*

Pattern Category	Mechanism	Implementation Approach	Failure Protection	Recovery Strategy
Idempotency	Correlation identifiers, deduplication logic	Request ID propagation across all stages	Duplicate billing prevention	Safe retry execution
Timeout Management	Deadline enforcement per dependency	SLA-based timeout configuration	Slow dependency isolation	Fail-fast behavior
Circuit Breakers	Error threshold monitoring	Automated circuit tripping at error limits	Cascading failure prevention	Gradual recovery ramp
Stateless Design	Minimal transient context retention	Correlation IDs only, no session state	Simplified failure recovery	Any-instance processing
Bulkhead Isolation	Subsystem fault containment	Independent resource pools per service	Fault propagation blocking	Partial system availability

Table 4. *Future Evolution Considerations and Architectural Adaptations [3][7][8]*

Evolution Domain	Emerging Challenge	Architectural Requirement	Proposed Adaptation	Expected Benefit
Advanced AI Models	Large language models, complex RL agents	Expanded computational footprints	Clear layering for model integration	Seamless model type additions
Streaming Signals	Real-time external data feeds	Continuous policy evolution	Strong contract adherence	Independent signal source evolution
Transparency Demands	Algorithmic accountability requirements	Enriched decision metadata	Enhanced semantic logging	Regulatory compliance support
Energy Constraints	Datacenter power consumption limits	Power-aware orchestration	Energy-conscious routing policies	Sustainability goal achievement
Team Autonomy	Independent capability evolution	Contract-based boundaries	Objective SLA metrics between teams	Reduced coordination overhead

6. Conclusions

Orchestration within cloud-native advertising platforms undergoes reframing from auxiliary implementation particulars toward principal architectural consideration. The reference framework rests upon interconnected constructs incorporating distinct stratification separating coordination from commercial logic, robust agreements enabling autonomous evolution, stateless coordination simplifying expansion and remediation, and primary visibility rendering system behavior comprehensible to operators and stakeholders. The conceptual model furnished toward practitioners centers upon comprehending how individual requests traverse systems, rationale whereby architecture sustains both minimal latency and resilience, and positions where novel models and policies integrate without fragility introduction. Through orchestration treatment as an intentional design challenge rather than an emergent microservice composition property, groups can construct advertising platforms sustaining adaptability, confronting evolving machine learning methodologies, shifting commercial prerequisites, and escalating operational intricacy. Though this work concentrates specifically on advertising infrastructure, the constructs and tenets demonstrate extensive applicability toward any latency-constrained, intelligence-augmented cloud-native solution. The fundamental challenges of synchronizing multiple dependencies under stringent temporal allocations, sustaining resilience under partial disruptions, enabling perpetual evolution, and furnishing operational visibility manifest across domains spanning financial trading through real-time personalization. The reference framework serves as both a conceptual foundation and a practical initiation point for engineering groups confronting these challenges within their respective mechanisms.

Author Statements:

- **Ethical approval:** The conducted research is not related to either human or animal use.
- **Conflict of interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper
- **Acknowledgement:** The authors declare that they have nobody or no-company to acknowledge.
- **Author contributions:** The authors declare that they have equal right on this paper.
- **Funding information:** The authors declare that there is no funding to be acknowledged.
- **Data availability statement:** The data that support the findings of this study are available on request from the corresponding author. The data are not publicly available due to privacy or ethical restrictions.

References

[1] Oyekunle Claudius Oyeniran, et al., "Microservices architecture in cloud-native applications: Design patterns and scalability," Computer Science & IT Research Journal, vol. 5, no. 9, pp. 2107–2124, September 2024. Available: https://doi.org/10.51594/csitrj.v5i9.1554

[2] Bowen Li et al., "Enjoy your observability: an industrial survey of microservice tracing and analysis," Empirical Software Engineering, vol. 27, article 25, 2022. Available: https://link.springer.com/article/10.1007/s10664-021-10063-9

[3] Eric Masanet, et al., "Recalibrating global data center energy-use estimates," Science, vol. 367, no. 6481, pp. 984–986, 28 February 2020. Available: https://www.science.org/doi/10.1126/science.aba3758

[4] Hassan B. Hassan, et al., "Survey on serverless computing," Journal of Cloud Computing: Advances, Systems and Applications, vol. 10, article 39, 2021. Available: https://journalofcloudcomputing.springeropen.com/articles/10.1186/s13677-021-00253-7

[5] Luis Miralles-Pechuán, et al., "Real-time bidding campaigns optimization using user profile settings," Electronic Commerce Research, vol. 23, no. 2, pp. 1297–1322, June 2023. Available: https://link.springer.com/article/10.1007/s10660-021-09513-9

[6] Ummay Faseeha, et al., "Observability in Microservices: An In-Depth Exploration of Frameworks, Challenges, and Deployment Paradigms," IEEE Access, early access, 2025. Available: https://www.researchgate.net/publication/390903567

[7] Xiangyu Zhao, et al., "DEAR: Deep Reinforcement Learning for Online Advertising Impression in Recommender Systems," in Proc. AAAI Conf. on Artificial Intelligence, vol. 35, no. 1, pp. 750–758, 2021. Available: https://ojs.aaai.org/index.php/AAAI/article/view/16156

[8] Xiangyu Zhao, et al., "Deep reinforcement learning for search, recommendation, and online advertising: a survey," ACM SIGWEB Newsletter, Spring 2019, Article 4, pp. 1–15, 2019. Available: https://dl.acm.org/doi/10.1145/3320496.3320500