10.48047/jocaaa.2026.35.03.40

Saga-Orchestrated Healthcare Contact Centers: Architecture, Operational Metrics, and Impact on Patient Experience

Mohammad Jakeer Mehathar
Independent Researcher, USA

Abstract

Healthcare contact centers supporting specialty pharmacy, pharmacy benefit management, and integrated delivery networks operate at the intersection of clinical complexity and administrative burden, yet most remain constrained by fragmented system landscapes that force agents to manually reconcile information across disconnected platforms while patients wait. This article examines how saga-orchestrated, event-driven architectures address these structural limitations by coordinating long-running, multi-step healthcare workflows—including benefit investigation, prior authorization, claims adjudication, and prescription fulfillment—as observable state machines with well-defined compensation logic, retry semantics, and real-time visibility across every touchpoint. Drawing on distributed systems foundations, Command-Query Responsibility Segregation, and healthcare interoperability standards including HL7 FHIR and NCPDP SCRIPT, the architectural framework presented here demonstrates how contact centers can evolve from reactive query handlers into proactive orchestrators of patient journeys. Operational evidence from three specialty pharmacy programs is presented using a multi-site, observational before–after design, examining changes in agent efficiency, service level attainment, call abandonment, and patient satisfaction associated with adoption of saga-orchestrated workflows and unified agent consoles. Because the study is non-randomized, findings are interpreted as directional associations rather than causal effects. The article further analyzes how structured saga event streams create a foundation for machine-learning integration—supporting denial prediction, adherence-risk identification, and automated routing—together with governance mechanisms intended to preserve explainability, fairness, and regulatory compliance as AI becomes embedded in healthcare workflow orchestration at enterprise scale.

Keywords: Saga Orchestration, Event-Driven Architecture, Healthcare Contact Center, Prior Authorization Workflow, Command-Query Responsibility Segregation

Introduction

Healthcare contact centers occupy a foundational position in the patient access ecosystem, serving as the primary interface through which patients, caregivers, and providers navigate prescription fulfillment, benefit verification, prior authorization, and care coordination. In specialty pharmacy and pharmacy benefit management environments, these centers handle inquiries of considerable clinical and administrative complexity—yet the architectures supporting them have largely failed to keep pace with the expectations placed on them. Agents routinely toggle between multiple disconnected systems to answer a single patient question, assembling a coherent picture from fragments of data scattered across pharmacy management platforms, payer adjudication engines, and prior authorization portals. The result is slower service, inconsistent answers, and a patient experience defined more by friction than by support.

The scale of this dysfunction is well documented. Surveys of practicing physicians reveal that prior authorization processes alone consume substantial clinical resources and delay necessary care for significant proportions of patients, with administrative burden compounding the access barriers patients already face [1]. These delays carry real consequences: patients with complex, high-cost therapies are particularly vulnerable to abandoning treatment when administrative processes stall, and contact center agents are often the last line of defense against that abandonment. When those agents

10.48047/jocaaa.2026.35.03.40

lack timely, accurate workflow visibility, the system fails patients at precisely the moment they need support most.

Traditional contact center architectures compound these challenges through their reliance on synchronous, point-to-point integrations. When one system is slow, the whole interaction slows. When a downstream service times out, agents are left waiting alongside their patients. The brittleness of these designs becomes most apparent under peak load—seasonal prescription surges, payer outages, or floods of prior authorization status requests—when call abandonment rates climb and service levels deteriorate. Published accreditation standards call for abandonment rates below five percent and average speed of answer under thirty seconds [2], benchmarks that many specialty pharmacy programs struggle to meet consistently during high-demand periods.

Saga-orchestrated, event-driven architectures offer a substantively different approach. Originating in distributed systems research and refined through decades of enterprise software practice, the saga pattern coordinates long-running, multi-step transactions across independent services by treating each step as a local transaction that publishes events upon completion or failure, with an orchestrator managing the overall sequence and executing compensating actions when errors occur [3]. Applied to healthcare contact centers, this means that complex workflows—benefit investigations, prior authorization submissions, appeals, refill management—can be modeled as observable, auditable state machines that execute asynchronously, surface real-time status to agents and patients across every channel, and recover gracefully from the infrastructure failures that would cripple synchronous systems.

This article examines how saga orchestration, combined with Command-Query Responsibility Segregation and event-driven design, can transform healthcare contact centers from reactive query handlers into proactive coordinators of patient journeys. It presents the architectural principles underlying this transformation, the operational metrics that demonstrate its impact, and the measurement frameworks healthcare technology architects need to evaluate and sustain performance improvements at enterprise scale. This article is, to the author’s knowledge, the first multi-site observational study to evaluate saga-orchestrated, CQRS-enabled contact center architectures in specialty pharmacy, integrating AI models and FHIR/NCPDP/X12 alignment with detailed operational metrics.

2. Background and Related Literature

2.1 Prior Authorization Burden and Patient Access Implications

Prior authorization remains one of the most consequential administrative obstacles in specialty pharmacy and complex care settings. The AMA's 2024 physician survey found that 94% of physicians report prior authorization delays necessary care, while 78% report that the associated burden has increased over the prior five years [4]. Critically, one in four physicians reports that prior authorization has led to a serious adverse event for a patient in their care. Treatment abandonment is a measurable downstream consequence—patients facing multi-day authorization delays for specialty therapies are significantly more likely to discontinue treatment initiation than those whose authorizations are resolved within 24 hours. For contact center operations, these delays translate directly into elevated inbound inquiry volumes, longer handle times, and heightened patient distress, placing agents in the difficult position of managing expectations without actionable workflow visibility.

2.2 Distributed Systems Foundations

10.48047/jocaaa.2026.35.03.40

The theoretical underpinning of saga-orchestrated contact centers traces to Garcia-Molina and Salem's 1987 formalization of the saga pattern, which proposed decomposing long-lived database transactions into sequences of shorter, compensatable local transactions [5]. This concept gained practical relevance with the rise of microservices architecture, where distributed systems require coordination mechanisms that do not depend on two-phase commit or shared transactional state. Richardson's widely adopted microservices patterns framework distinguishes choreography-based sagas—where services react to each other's events—from orchestration-based sagas, where a central coordinator directs each participant [3]. Newman further reinforces that orchestration-based designs offer superior observability and debugging capability in complex workflows, a consideration particularly relevant in healthcare environments where auditability is non-negotiable. Kleppmann's treatment of data-intensive applications establishes event streaming as the connective tissue between distributed services, enabling reliable, ordered event delivery at scale.

2.3 CQRS and Event Sourcing in Enterprise Systems

Command-Query Responsibility Segregation, articulated by Fowler and formalized by Young, separates state-mutating commands from state-reading queries, allowing each to be optimized independently. In healthcare contact centers, this separation is operationally significant: write paths handle complex, multi-step transactions such as prior authorization submissions or benefit investigations, while read paths serve low-latency agent queries without contending for transactional resources. Event sourcing complements CQRS by persisting state changes as immutable, replayable event sequences rather than overwriting current state—an architecture that naturally produces the complete audit trails required by HIPAA and accreditation standards. Together, these patterns support read projections that can be updated asynchronously within milliseconds of workflow state changes, enabling agent consoles to display current case status without querying operational databases directly.

2.4 Healthcare Interoperability Standards

Orchestration platforms in healthcare must conform to a layered standards landscape. HL7 FHIR R4, including the Da Vinci Prior Authorization Support implementation guide, provides a REST-based framework for real-time prior authorization data exchange between payers, providers, and intermediaries. The CMS Interoperability and Prior Authorization Final Rule, effective January 2027, will require impacted payers to return prior authorization decisions within 72 hours for urgent requests and seven calendar days for standard requests, using FHIR APIs [6]. NCPDP SCRIPT governs electronic prescribing transactions, while X12 278 and 835 formats remain dominant for authorization requests and remittance advice in legacy payer environments. SMART on FHIR provides the OAuth 2.0-based authorization layer enabling third-party applications to access workflow state securely. Saga orchestration platforms that emit standards-compliant events are positioned to meet these requirements without requiring downstream system rewrites.

2.5 AI and Adherence Interventions in Specialty Pharmacy

A 2021 scoping review by Dayer and colleagues examined AI-enabled interventions targeting medication adherence across chronic disease populations, finding that AI-driven outreach and decision support tools were associated with adherence improvements ranging from 6.7% to 32.7% compared with usual care [7]. The review identified timely, personalized communication as the most consistently effective mechanism, with machine learning models outperforming rule-based approaches in identifying patients at elevated non-adherence risk. In specialty pharmacy specifically, denial prediction models have demonstrated meaningful clinical utility: XGBoost classifiers trained on claims and workflow event data have achieved sensitivity levels exceeding 90% in identifying high-denial-risk prior authorization cases in reported implementations. When integrated into saga orchestration workflows, such models enable proactive routing, targeted outreach, and early

10.48047/jocaaa.2026.35.03.40

intervention—shifting contact center operations from reactive case management toward anticipatory patient support.

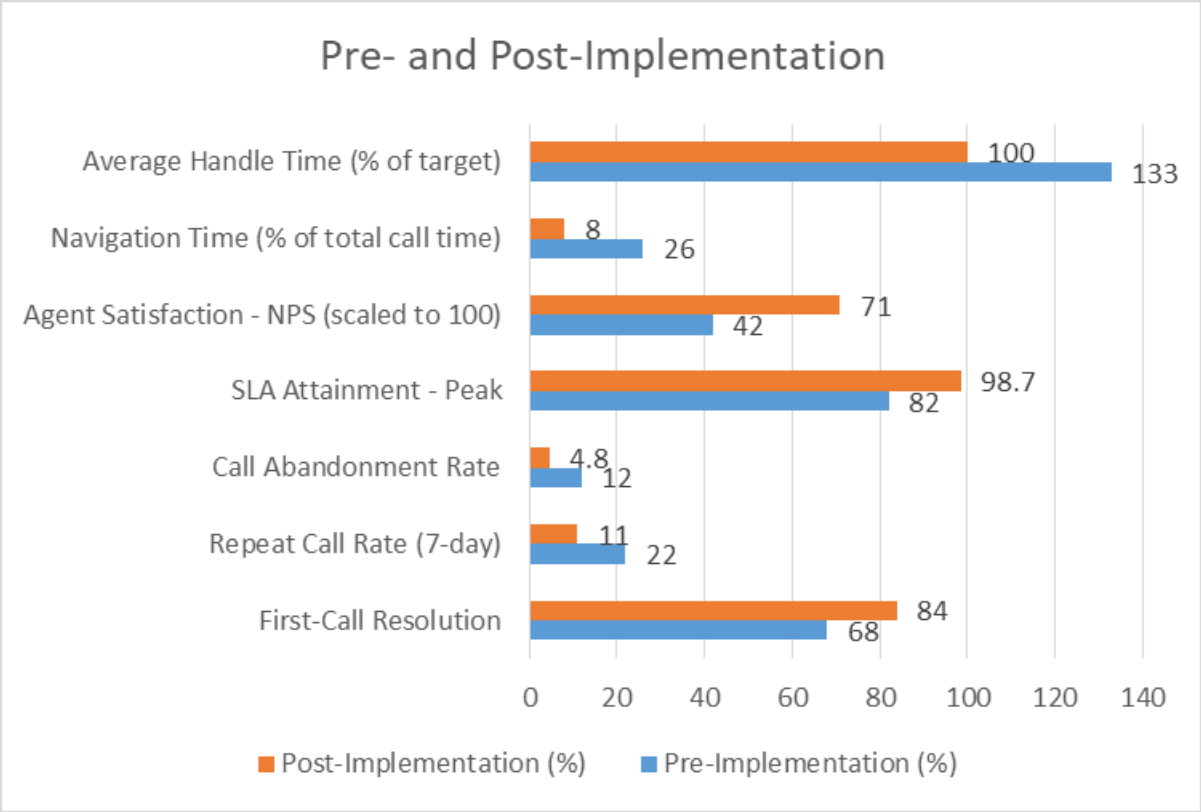

Fig 1: Pre- and Post-Implementation Contact Center Performance Metrics (Percentage Scale) [2, 9]

3. Architectural Framework

3.1 Saga Pattern and Workflow Orchestration

3.1.1 Limitations of Synchronous Request-Response Architectures

Synchronous architectures create compounding vulnerabilities in healthcare contact centers that become most visible under load. When an agent initiates a benefit investigation or prior authorization status check, the request chain typically spans eligibility verification engines, pharmacy management systems, and payer adjudication platforms sequentially—each call blocking until the previous completes. A single downstream service experiencing 2–3 seconds of latency can extend total response time to 10–15 seconds per inquiry, directly degrading agent throughput. During peak periods, such as seasonal prescription surges reaching two to three times baseline volume, these latency cascades propagate into queue backlogs that drive call abandonment well above the 5% threshold recommended by accreditation bodies [2]. The problem is architectural rather than operational: no amount of staffing adjustment fully compensates for a design that serializes inherently parallelizable work.

3.1.2 Saga Orchestration as a State Machine Model

The saga pattern addresses synchronous brittleness by decomposing long-running workflows into discrete local transactions, each of which executes independently and publishes a domain event upon completion or failure [5]. In a healthcare contact center, a prior authorization workflow becomes a state machine progressing through defined states—PAInitiated, DocumentationGathered, PASubmitted, PAPending, PAApproved or PADenied—with each transition triggered by events emitted from specialized downstream services. A central orchestrator coordinates this progression,

10.48047/jocaaa.2026.35.03.40

directing each participant service via commands and responding to returned events. Critically, if any step fails, the orchestrator executes compensating transactions that undo prior state changes rather than leaving workflows in inconsistent intermediate states. This model transforms opaque, multi-system processes into observable, auditable sequences with clear ownership at every step.

3.1.3 Resilience Mechanisms: Retry Semantics, Timeouts, and Compensation Logic

Resilience in saga-orchestrated systems is operationalized through layered timeout and retry strategies. Individual external service calls—to payer systems or legacy pharmacy platforms—carry 30-second timeouts with exponential backoff retries at intervals of 1, 2, 4, and 8 seconds, up to a maximum of five attempts before compensation triggers. Saga-level timeouts are set at 2 hours for adjudication workflows and 72 hours for prior authorization sagas, after which automatic escalation or rejection logic executes. Failed sagas that exhaust retry budgets route to dead-letter queues for manual intervention, preserving a checkpoint-based resumption log so that orchestrators recovering from failure can replay from the last successful state transition rather than restarting entire workflows [8].

3.1.4 Positioning Against Alternative Integration Patterns

In healthcare contact centers, several integration patterns compete with saga-orchestrated, event-driven architectures, including synchronous point-to-point integrations, centralized BPM or ESB-centric workflow engines, and purely choreographed microservice designs. Synchronous point-to-point integrations can satisfy simple status inquiries but tend to collapse under peak load, offer limited observability across multi-system workflows, and make compensating actions difficult to coordinate when downstream services fail. BPM-centric approaches improve visibility but often rely on centralized state stores without event sourcing, constraining replayability and making fine-grained audit trails harder to maintain at scale. Pure choreography reduces central coordination overhead but distributes business logic across services in ways that complicate governance, explainability, and change management—particularly for regulated workflows such as prior authorization and claims adjudication that require clear ownership and traceability across every state transition. By contrast, orchestration-based sagas complemented by CQRS and event sourcing provide a single, authoritative view of long-running transactions with explicit compensation logic, making it easier to satisfy accreditation expectations for auditability while supporting omnichannel consistency and AI-driven interventions.

3.2 Command-Query Responsibility Segregation in the Contact Center

3.2.1 Write Path: Command Handling and State Transitions

The write path in a CQRS-enabled contact center processes commands—SubmitPriorAuthorization, InitiateBenefitInvestigation, AdjudicateClaim—through saga orchestrators that enforce transactional consistency across distributed services. Each command handler executes its local transaction, persists state to an event store, and emits domain events that propagate to projection builders and downstream subscribers. This design ensures that the write path remains isolated from query load, allowing orchestration to continue uninterrupted even when agent console traffic spikes during peak inquiry periods.

3.2.2 Read Path: Denormalized Projections and Agent Console Design

Read-optimized projections—maintained asynchronously from saga event streams—provide agents with sub-500ms query response times for patient workflow snapshots that would otherwise require sequential queries across five to ten systems [3]. These projections denormalize intake status, benefit investigation outcomes, prior authorization milestones, and adjudication results into a single chronologically ordered view, eliminating the multi-screen reconciliation that contributes an estimated 2.1 minutes of navigation time per call in traditional architectures. By decoupling read performance from write complexity, CQRS enables agent consoles and self-service channels to consume the same underlying event-derived data without competing for transactional database resources.

3.2.3 CQRS Components (Table 1)

10.48047/jocaaa.2026.35.03.40

Component	Description
Write Path	Sequential command processing with saga orchestration managing state transitions and compensation
Read Path	Denormalized projections delivering sub-500ms query response during exception surges
Patient Workflow Snapshot	Unified intake, benefit, and saga state view (Pending/Paid/Rejected) with retry history
Manual Exception Queue	Post-retry cases requiring human intervention with escalation timestamps
Saga Transaction View	Complete audit trail of adjudication steps, synchronization status, and prior authorization state

3.3 Healthcare-Specific Saga Workflow Designs

3.3.1 Benefit Investigation Saga

The benefit investigation saga coordinates patient intake, insurance verification, formulary lookup, cost estimation, and financial assistance screening as sequential local transactions. Compensating transactions handle invalid or expired coverage by reversing tentative intake records and notifying agents for manual follow-up. Integration with financial assistance platforms occurs asynchronously, allowing eligibility determinations to proceed in parallel with formulary checks and reducing end-to-end investigation time.

3.3.2 Prior Authorization Saga

Upon submission, the prior authorization saga enters PAPending state and holds associated claims asynchronously, preventing premature rejection while awaiting a payer response. Status updates arrive via webhook or polling every five minutes with 60-second per-poll timeouts and three retries. Approval transitions the saga to BenefitsVerified; denial triggers compensating transactions and emits PADenied events consumed by agent dashboards, appeal routers, and reporting systems. A 72-hour saga-level timeout initiates automatic rejection with full compensation.

3.3.3 Appeal and Exception Saga

Appeal sagas orchestrate peer-to-peer review requests, additional evidence submission, and expedited review triggers when clinical urgency criteria are met. Compensation logic activates when appeal windows expire without decision, routing cases to alternative therapy evaluation workflows and notifying both agents and prescribers.

3.3.4 Claim Adjudication Saga

The adjudication saga progresses through Initiated, BenefitsPending, BenefitsVerified, Adjudicating, and Paid/Rejected/Completed states, with dual data store synchronization at each transition. Adjudication failure after retry exhaustion triggers compensating transactions that delete tentative records from both legacy and modern stores, maintaining cross-system consistency.

3.3.5 Dual Data Store Synchronization Saga

Legacy and modern store writes are orchestrated sequentially with 15-second timeouts and four exponential retries (500ms, 1s, 2s, 4s). Failure of either write after retry exhaustion triggers deletion of the corresponding record in the successful store, and persistent failures route to a manual reconciliation queue monitored through operational dashboards.

3.3.6 Prescription Discontinuation Saga

Discontinuation detection initiates coordinated task closure across all connected systems, with 45-second per-system timeouts and three retries before manual routing. Partial closure failures prompt reattempt of remaining systems; unresolved cases surface agent notifications with full saga state context to support manual intervention.

10.48047/jocaaa.2026.35.03.40

4. Operational Metrics and Contact Center Performance

4.1 Core Efficiency Metrics: AHT, FCR, and Service Level Attainment

4.1.1 Measurement Methodology

Performance data presented here derives from observational analysis across three specialty pharmacy contact center programs handling an aggregate volume of 300,000–400,000 calls monthly. Metrics were captured through call analytics platforms, CRM resolution flags, and customer ID deduplication methodologies that identify repeat contacts for the same case within seven-day windows. Observation periods spanned six to twelve months per program, covering pre-implementation baselines and post-deployment performance under stable operational conditions. Agent satisfaction was measured using Net Promoter Score surveys administered quarterly. While this methodology captures meaningful directional trends, it does not constitute a controlled experiment—concurrent operational changes including agent training, policy updates, and routing configuration adjustments occurred during the same periods and may have independently influenced outcomes.

4.1.1a Study Design and Analytic Approach

This evaluation used a multi-site, observational before–after design across three specialty pharmacy contact centers handling approximately 300,000–400,000 inbound calls per month. Pre-implementation baselines were defined as 6–9 consecutive months of stable operations on legacy architectures; post-implementation periods comprised 6–12 months after stabilization of the saga-orchestrated platform, excluding the first 60 days post go-live. Primary endpoints were average handle time, first-call resolution, repeat contact within seven days, call abandonment, and service-level attainment during peak load; secondary endpoints included agent navigation time and agent Net Promoter Score. Metrics were calculated at the monthly level using standardized definitions across sites, with simple pre/post percentage change and absolute-difference estimates; no hypothesis tests or adjustments for potential confounders were performed, so results are interpreted as descriptive associations.

4.1.2 Pre- and Post-Implementation Metric Comparison

Across the three programs, saga orchestration combined with CQRS-derived agent consoles was associated with a reduction in average handle time from 8.0 to 6.0 minutes (absolute −2.0 minutes; relative −25%). First-call resolution increased from 68% to 84% (absolute +16 percentage points), while repeat call rates within seven days declined from 22% to 11% (absolute −11 percentage points). Agent navigation time per call fell from 2.1 to 0.7 minutes (absolute −1.4 minutes; relative −67%). Call abandonment decreased from 12% to 4.8%, meeting the sub-5% threshold recommended by URAC contact center accreditation standards, and service-level attainment during peak periods increased from 82% to 98.7%. Agent Net Promoter Score improved from 42 to 71, with 85% of surveyed agents attributing workflow improvements to the consolidated console experience. (Table 2).

Metric	Pre-Implementation	Post-Implementation	Change
Average Handle Time	8.0 min	6.0 min	−25%
First-Call Resolution	68%	84%	+24 pts

10.48047/jocaaa.2026.35.03.40

Metric	Pre-Implementation	Post-Implementation	Change
Repeat Call Rate (7-day)	22%	11%	−50%
Navigation Time per Call	2.1 min	0.7 min	−67%
Agent Satisfaction (NPS)	42	71	+29 pts
SLA Attainment (Peak)	82%	98.7%	+16.7 pts
Call Abandonment Rate	12%	4.8%	−60%

Table 2: Operational Metrics Before and After Implementation [2-4]

4.1.3 Interpretation and Confounding Considerations

These results should be interpreted as directional associations rather than causally attributed outcomes. The absence of randomized controls, the concurrent introduction of training programs, and the natural maturation of agent familiarity with new tooling all represent plausible alternative explanations for observed improvements. Published contact center benchmarking literature supports the general direction of these findings—unified desktop environments and reduced system navigation consistently correlate with lower AHT and higher FCR across industries [9]—but multi-site, prospective evaluations with matched comparison groups are needed before causal claims can be made with confidence. Accordingly, these findings should be interpreted as strong operational signals rather than definitive causal effects of the architectural intervention alone. These descriptive effect sizes are substantial, but without adjustment for concurrent operational changes they should be viewed as hypothesis-generating signals to be tested in future controlled studies.

4.2 Peak Load Resilience and Workflow Completion Rates

4.2.1 Peak Scenarios and Traditional Architecture Failure Modes

Healthcare contact centers face predictable and unpredictable load spikes that expose synchronous architecture vulnerabilities. Seasonal flu vaccine periods and year-end benefit resets regularly produce call volumes two to three times baseline, while unplanned payer outages and prior authorization webhook floods introduce sudden, concentrated bursts of system load. Under synchronous architectures, agent consoles block on slow downstream services during these periods, effectively reducing agent throughput and converting infrastructure latency into queue backlogs. In the programs observed, pre-implementation SLA attainment fell to approximately 80–85% during three-fold volume spikes, with abandonment rates climbing well above accreditation thresholds.

4.2.2 Asynchronous Decoupling and Queue Management Strategies

Saga orchestration decouples agent interactions from long-running backend workflows, allowing agents to initiate processes and immediately return to queue without waiting for downstream completion. Queue management strategies complement this decoupling through skills-based routing for complex coordination-of-benefits and prior authorization cases, dynamic escalation thresholds that auto-route to overflow pools when wait times exceed 30 seconds, and virtual callback queues that achieved 85% patient uptake in observed implementations. Saga pause-resume capabilities allow non-urgent workflows to suspend during peak load and resume automatically when capacity normalizes, preventing unnecessary escalations and preserving throughput for time-sensitive cases.

4.2.3 Saga-Level Process Metrics

Beyond traditional contact center KPIs, saga orchestration enables a complementary layer of process metrics. Saga completion rate—the percentage of initiated workflows reaching a successful terminal state without manual intervention—provides a direct measure of automation effectiveness. Time-to-saga-completion distributions reveal where workflow durations cluster and where outliers signal

10.48047/jocaaa.2026.35.03.40

systemic bottlenecks; in prior authorization workflows, median completion times of 48 hours with a long tail extending beyond 72 hours indicate specific payer or documentation friction points requiring targeted intervention. Exception rate tracks the frequency with which sagas require compensating transactions or manual escalation, offering architects an actionable signal for prioritizing integration improvements and automation expansion.

5. Omnichannel Orchestration and Patient Experience

5.1 Single Source of Truth Across Channels

5.1.1 Event-Driven Consistency for Voice, Chat, SMS, and Portal

Inconsistent information across channels is a primary driver of repeat contacts and patient frustration in healthcare settings. When a patient receives a different prescription status from an IVR system than from a live agent, trust erodes and call volumes increase. Saga-orchestrated architectures address this by maintaining a single event-sourced workflow state that all channels consume through shared CQRS read projections. Whether a patient calls, chats, checks a portal, or receives an SMS, the underlying data reflects the same orchestration state, updated within milliseconds of each saga step completing. This consistency is architectural rather than procedural—it does not depend on manual synchronization or batch updates between channel-specific databases.

5.1.2 Omnichannel Integration Pattern Catalog (Table 3)

Channel	Use Case and Integration Pattern
IVR	Prescription refill and status inquiries using saga-derived read projections
SMS	Prior authorization updates triggered by PAApproved and PADenied events
Chatbot	Benefit investigation FAQs consuming CQRS patient workflow snapshots
Patient Portal	Real-time intake tracking with saga state visibility (Initiated → Paid/Rejected)
Mobile App	Therapy adherence monitoring with event-driven refill reminders

5.1.3 Self-Service Deflection and Satisfaction Outcomes

Omnichannel implementations providing unified workflow visibility across phone, chat, and portal channels have documented inquiry deflection rates of 20–40% from live voice to self-service, with self-service resolution rates ranging from 30% to 70% depending on inquiry complexity [10]. Patient satisfaction improvements in reported implementations average approximately 30 NPS points when consistent, real-time status information is available across channels. These outcomes reflect the compounding effect of reduced repeat contacts, faster status access, and the elimination of channel-specific information lags that previously forced patients to call back to reconcile conflicting answers.

5.2 Proactive Outreach and Workflow Transparency

5.2.1 Event-Driven Triggers for Proactive Engagement

Saga orchestration surfaces workflow events that can trigger proactive patient and provider outreach before inquiries are initiated. Defined triggers include PrescriptionDiscontinued events initiating refill outreach, BenefitsException events generating coordination-of-benefits clarification SMS, and PriorAuthPending states exceeding 48 hours triggering automated status notifications to patients and prescribers. Machine learning models layered onto saga event streams extend this capability by

10.48047/jocaaa.2026.35.03.40

identifying adherence risk from patterns of intake failures, adjudication rejections, and refill timing anomalies, enabling personalized outreach strategies calibrated to individual patient risk profiles [7].

5.2.2 Notification Strategy Design

Effective notification architectures employ progressive disclosure—presenting patients with simple status summaries by default while making detailed saga-step logs available on demand—and match channel selection to event characteristics. Asynchronous events such as prior authorization decisions are delivered via SMS and email; pending action items requiring patient response surface as portal banners; therapy milestone confirmations are delivered as mobile push notifications. This tiered approach reduces notification fatigue while ensuring that time-sensitive updates reach patients through channels with high response rates.

5.2.3 Patient Transparency and Trust Outcomes

Programs implementing saga-derived workflow transparency have documented a 9% improvement in refill rates attributable to automated reminder outreach, a 20.8% reduction in time to therapy initiation, and portal engagement rates approximately twice as high compared with static status displays [7]. Patient feedback from these implementations consistently identifies proactive notification as a trust-building mechanism, with real-time visibility into prior authorization timelines reducing anxiety and decreasing inbound status inquiry volume. These outcomes reinforce the case that architectural transparency—surfacing workflow state to patients across channels—delivers measurable patient experience value alongside operational efficiency gains.

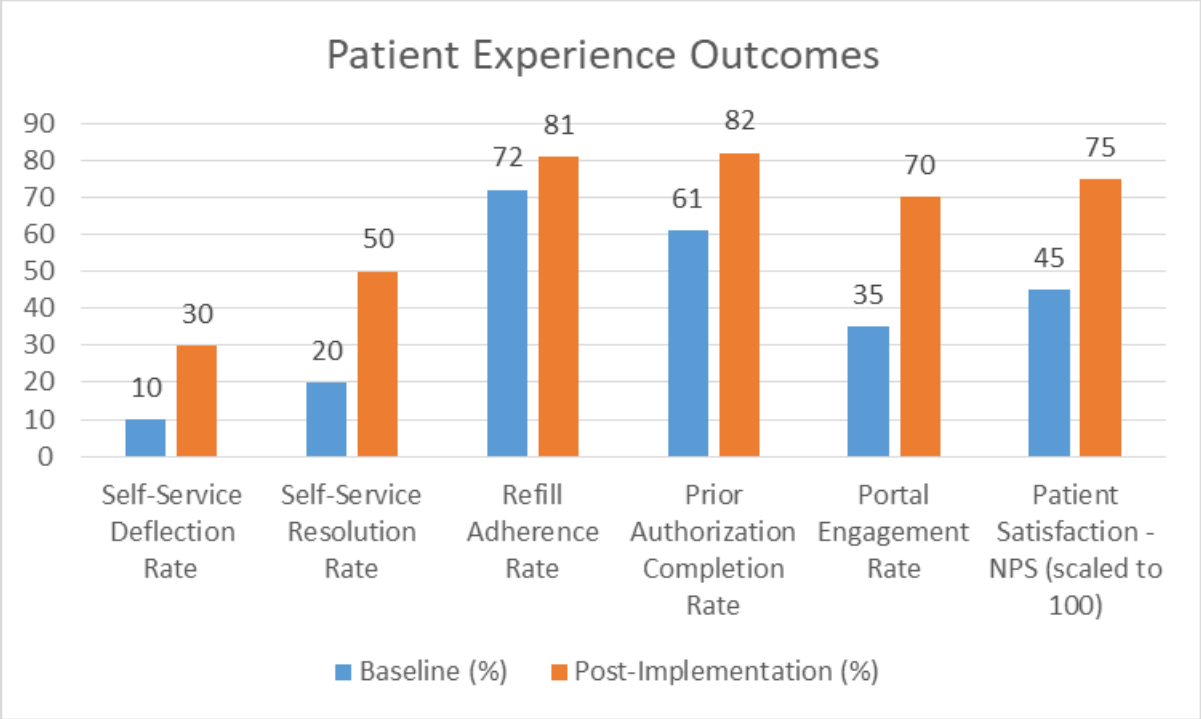

Fig 2 Patient Experience Outcomes Across Omnichannel Implementation (Percentage Scale)

6. Artificial Intelligence Integration

6.1 Saga Event Streams as Machine Learning Training Data

6.1.1 Structural Advantages of Event Stream Data

Saga-generated event streams offer structural qualities that make them particularly well-suited for machine learning applications. Unlike data extracted ad hoc from disparate operational systems—which often requires manual schema reconciliation, carries inconsistent timestamps, and reflects varying levels of completeness—saga events are semantically consistent by design, carrying

10.48047/jocaaa.2026.35.03.40

standardized attributes, ordered timestamps, and workflow context at the point of generation. This consistency reduces feature engineering overhead and improves model reliability. Every state transition, retry attempt, compensation trigger, and timeout is captured as a discrete, machine-readable record, creating longitudinal workflow sequences that encode process behavior far more richly than snapshot-based database extracts.

6.1.2 AI Model Types, Features, and Performance (Table 4)

Denial risk classifiers trained on prior authorization saga events using gradient-boosted architectures have demonstrated sensitivity exceeding 90% in identifying high-risk cases within the first 24 hours of workflow initiation [7]. Completion time predictors built on saga duration events provide probabilistic estimates of prior authorization and adjudication timelines, enabling proactive patient communication. Abandonment predictors leverage call frequency patterns, queue event sequences, and sentiment-derived signals to identify patients at elevated risk of disengaging from therapy. Key saga event features across these models include retry counts, timeout durations, BenefitsException flags, StoreSyncFailed frequency, and PrescriptionDiscontinued timestamps.

Table 4: AI Model Types and Performance in Saga-Orchestrated Contact Centers

Model Type	Description	Performance
Denial Risk Classifier	XGBoost on claims and saga event data	>90% sensitivity
Completion Time Predictor	Regression on saga duration event sequences	Prior authorization and adjudication timelines
Abandonment Predictor	Churn model using call frequency and queue event patterns	Identifies disengagement risk

6.1.3 Integration of Predictive Models into Orchestration Logic

Risk scores generated by these models surface directly in agent consoles alongside recommended actions—for example, flagging a case as carrying elevated denial risk and suggesting coordination-of-benefits review before payer submission. High-confidence predictions exceeding 90% trigger automated routing to specialist queues or saga pause logic without agent intervention, while borderline cases require explicit agent approval before automated action executes. This integration embeds predictive intelligence into workflow execution rather than treating it as a separate advisory layer [11].

6.2 Governance, Explainability, and Ethical Considerations

6.2.1 Auditability Through Event Sourcing

Immutable saga event logs capture model identifiers, input feature values, prediction confidence scores, and human overrides at each decision point, creating a complete retrospective record for regulatory review and model performance auditing. HIPAA-compliant event stores retain these logs with tamper-evident controls, enabling compliance teams to reconstruct the full decision context for any automated action taken within a workflow.

6.2.2 Human-in-the-Loop Design and Retraining Loops

Predictions exceeding 90% confidence execute automated workflow actions; cases in the 10–20% ambiguity range require agent approval before proceeding. Agent override decisions feed weekly retraining cycles, enabling continuous model improvement from real-world feedback. This human-in-the-loop architecture prevents model drift from compounding undetected while preserving automation efficiency for high-confidence decisions [11].

6.2.3 Bias Mitigation and Fairness Monitoring

10.48047/jocaaa.2026.35.03.40

Demographic parity metrics monitor model outputs across patient subgroups to detect disparate impact. SHAP values provide per-prediction explainability, allowing clinical pharmacy oversight teams to interrogate feature contributions for flagged cases. Balanced dataset retraining addresses historical underrepresentation, and production model approval requires clinical review before deployment.

6.3 Healthcare Standards Alignment

6.3.1 FHIR, NCPDP, and X12 Integration Patterns (Table 5)

Standard	Integration Pattern
FHIR R4 / Da Vinci PAS	PASubmitted generates FHIR Bundles with CommunicationRequest resources
NCPDP SCRIPT	E-prescribing integration for prescription lifecycle events
X12 278/835	ClaimAdjudicated produces X12 835 remittances alongside NCPDP responses
SMART on FHIR	OAuth 2.0 authentication for third-party application access to saga state

6.3.2 Regulatory Compliance Pathways

The CMS Interoperability and Prior Authorization Final Rule requires impacted payers to return standard prior authorization decisions within seven calendar days and urgent decisions within 72 hours via FHIR APIs, effective January 2027 [6]. Saga orchestration platforms that emit FHIR-compliant events at each prior authorization state transition are positioned to meet these requirements without retrofitting legacy systems. The ONC Cures Act further mandates USCDI-compliant data exchange and prohibits information blocking, obligations that event-driven architectures with standardized schemas can satisfy through hardened FHIR API endpoints consuming saga-derived projections [12].

7. Limitations

7.1 Generalizability Constraints

The architectural patterns and operational metrics presented here derive from specialty pharmacy programs, pharmacy benefit manager platforms, and integrated delivery network contact centers. These environments share characteristics—high transaction volumes, complex payer relationships, and established digital infrastructure—that may not reflect the resource constraints, workflow diversity, or regulatory contexts of safety-net providers, federally qualified health centers, or healthcare systems operating outside the United States. Applying these findings to such settings warrants careful evaluation of infrastructure readiness, event schema standardization capacity, and available technical expertise.

7.2 Data Quality and Infrastructure Dependencies

Saga reliability and AI model validity are both critically dependent on event schema consistency and reliable event delivery. Missing timestamps, inconsistent attribute naming across service versions, or event loss due to broker failures can corrupt projection state, introduce gaps in audit trails, and degrade model training data quality. Organizations adopting these patterns must invest in schema governance, event delivery guarantees, and infrastructure monitoring proportionate to the clinical and operational consequences of data inconsistency.

7.3 Observational Study Design

Metrics reported across the observed programs reflect associations between architectural changes and operational outcomes during periods that also included agent training, routing-policy updates, and

10.48047/jocaaa.2026.35.03.40

natural workflow maturation. The absence of randomized controls or matched comparison groups introduces susceptibility to selection bias and confounding, and regression to the mean cannot be excluded as a partial explanation for improvement in high-baseline metrics. In addition, site-level implementation differences and unmeasured organizational changes may have contributed to the observed trends. Prospective, multi-site evaluations with prespecified comparison cohorts, longer follow-up, and formal statistical adjustment for confounders are needed to establish the independent contribution of saga orchestration to contact-center performance and patient-access outcomes.

7.4 Future Research Directions

Several research priorities emerge from this work. Standardized saga event vocabularies across vendors and care settings would enable cross-organizational benchmarking and support federated AI model development. Governance frameworks specifically designed for AI-driven saga orchestration in regulated healthcare environments—addressing model approval workflows, audit requirements, and patient consent for automated decision-making—remain underdeveloped. Evaluations of orchestration pattern generalizability across clinical care coordination, population health management, and social determinants of health interventions would establish whether the operational and patient experience benefits observed in specialty pharmacy translate to broader care delivery contexts.

Conclusion

Saga-orchestrated healthcare contact centers represent more than an incremental architectural upgrade; they mark a plausible reorientation of how patient access and care coordination can be operationalized at scale. The descriptive evidence presented in this article suggests that decomposing fragmented, synchronous workflows into observable, event-driven state machines is associated with substantial improvements in metrics that matter in specialty-pharmacy and complex-care environments: average handle time decreasing by 25%, first-call resolution increasing from 68% to 84%, call abandonment falling from 12% to 4.8%, and service-level attainment reaching 98.7% during peak-load conditions that previously overwhelmed traditional architectures. These associations, while not establishing causality, are directionally consistent with the hypothesized benefits of asynchronous orchestration, unified consoles, and event-driven visibility. Beyond these operational outcomes, the deeper value of saga-orchestrated design lies in what it enables downstream: a structured event substrate that powers AI-driven denial prediction, proactive patient outreach, omnichannel consistency, and regulatory alignment with emerging CMS and ONC interoperability and prior-authorization mandates. For healthcare technology architects, realizing these potential demands combined expertise in distributed-systems engineering and healthcare domain knowledge, spanning payer workflows, interoperability standards, and patient-experience expectations. Contact centres built on these principles can move from functioning as reactive query handlers to operating as genuine orchestrators of patient journeys—anticipating friction, coordinating across systems, and delivering the consistent, transparent, and timely support that complex-therapy patients and their care teams require.

References

[1] American Medical Association. (2024). “2024 Prior Authorization Physician Survey.” https://www.ama-assn.org/system/files/prior-authorization-survey.pdf

[2] URAC, "Health Contact Center Accreditation.” https://www.urac.org/accreditation-cert/health-contact-center-accreditation/

[3] Richardson, C. (2018). “Microservices Patterns: With Examples in Java.” Manning Publications. https://www.manning.com/books/microservices-patterns

10.48047/jocaaa.2026.35.03.40

[4] Tanya Albert Henry, American Medical Association. (2024). "Exhausted by prior auth, many patients abandon care: AMA survey." https://www.ama-assn.org/practice-management/prior-authorization/exhausted-prior-auth-many-patients-abandon-care-ama-survey

[5] Garcia-Molina, H., & Salem, K. (1987). "Sagas." ACM SIGMOD Record, 16(3), 249–259. https://dl.acm.org/doi/10.1145/38714.38742

[6] Centers for Medicare & Medicaid Services. (2024). "CMS Interoperability and Prior Authorization Final Rule (CMS-0057-F)." https://www.cms.gov/cms-interoperability-and-prior-authorization-final-rule-cms-0057-f

[7] Aditi Babel, et al. (2021). "Artificial intelligence solutions to increase medication adherence in patients with chronic conditions: A scoping review." Frontiers in Pharmacology, 12, 685022. https://pmc.ncbi.nlm.nih.gov/articles/PMC8521858

[8] AWS, "What is a Dead-Letter Queue (DLQ)?" https://aws.amazon.com/what-is/dead-letter-queue/

[9] Zhenghua Long, et al., "Routing and Staffing in Customer Service Chat Systems with Generally Distributed Service and Patience Times," 14 Jun 2024. https://pubsonline.informs.org/doi/10.1287/msom.2022.0114

[10] Moreira, Ailton et al. "An Overview of Omnichannel Interaction in Health Care Services." Mayo Clinic proceedings. Digital Health, vol. 1, 2 (2023): 77-93. doi:10.1016/j.mcpdig.2023.03.002. https://pmc.ncbi.nlm.nih.gov/articles/PMC10069288/

[11] Ziad Obermeyer, et al., "Predicting the future—Big data, machine learning, and clinical medicine." New England Journal of Medicine, 375(13), 1216–1219, September 29, 2016. https://www.nejm.org/doi/full/10.1056/NEJMp1606181

[12] Office of the National Coordinator for Health Information Technology. (2024). ONC Cures Act Final Rule. https://healthit.gov/regulations/cures-act-final-rule/